Musenet : Music Generation using Abstractive and Generative Methods

Abhilash Pal, Saurav Saha, R. Anita

Abstract: ***Humans have been entertained by music for millennia. For ages it has been treated as an art form which requires a lot of imagination, creativity and accumulation of feelings and emotions. Recent trends in the field of Artificial Intelligence have been getting traction and Researchers have been developing and generating rudimentary forms of music through the use of AI. Our goal is to generate novel music, which will be non-repetitive and enjoyable. We aim to utilize a couple of Machine Learning models for the same. Given a seed bar of music, our first Discriminatory network consisting of Support Vector Machines and Neural Nets will choose a note/chord to direct the next bar. Based on this chord or note another network, a Generative Net consisting of Generative Pretrained Transformers(GPT2) and LSTMs will generate the entire bar of music. Our two fold method is novel and our aim is to make the generation method as similar to music composition in reality as possible. This in turn results in better concordant music. Machine generated music will be copyright free and can be generated conditioned on a few parameters for a given use.The paper presents several use cases and while the utilization will be for a niche audience, if a easy to use application can be built, almost anyone will be able to use deep learning to generate concordant music based on their needs.***

Keywords: ***AI in Art, Deep learning, Music Generation, Music Theory, Natural language processing, Predictive models, Tokenization***

I. INTRODUCTION

Music is an art form organized in time. It involves a free interplay of various sounds, pitches and forms which yield coherent hearing experiences. Because of its complexity, generating music using computing techniques is a bit difficult. However, with the advent of deep learning, many new avenues of research have opened up. These involve learning from data and replicating the patterns in music.

Revised Manuscript Received on April 30, 2020.
***** Correspondence Author

Abhilash Pal*, Department of Computer Science, SRM Institute of Science and Technology, Chennai, India. Email: abhilashpal8@gmail.com

Sourav Saha, Department of Computer Science, SRM Institute of Science and Technology, Chennai, India. Email: sauravsaha48@gmail.com

Anita, Department of Computer Science, SRM Institute of Science and Technology, Chennai, India. Email: anitar@srmist.edu.in

Code and Models : https://github.com/AbhilashPal/MuseNet

The application in our base paper by Dong et al. in MuseGan[3] used a different approach. They trained three different models. The models were GANs which could generate different instruments playing in unison using a set of GANs or a single GAN conditioned on an input vector.

This input vector acted as the seed in the network and was used by the different GANs to generate music with the same intent and alignment.

This architecture formed state of the art in music generation and utilized the Lakh Piano-Roll Dataset. They also formulated NLP based metrics to evaluate their models and validate them against a baseline.

Our methodology includes a novel approach wherein we shall generate music bar by bar, and the contents of each bar will be conditioned on a chord or set of notes. This chord will be chosen from a prefixed set of chords a song can use, using the discriminator. The generator can use the same chord as an intent vector and generate a bar of music. This shall make the music generated coherent and help it hopefully to have long term dependencies and structures which are inherent in today's music.

The expected results of the project will include novel generated music in a single scale, or raag (as present in Indian Classical Music). Basing our generative model on coherent sources like a scale or chord will yield results which sound better and appeal more.

The paper is divided into five sections. Section II explains the current trends in research, and how we build our own work on top of that. Section III explains our model, it's various parts and the training parameters, The next section IV concludes the paper with added anecdotes about how successive research may be handled.

II. STATE OF THE ART

Music Generation requires an innate amount of insight over a particular piece of music, similar to text which has been used widely in Natural Language based applications. Music Generation can be of two major types – extractive and abstractive. Extractive Music Generation provides a short musical phrase on the basis of music already present in the input files and merely uses an algorithm to chose parts of music on the basis of their importance to be used in the generated phrase. In case of Abstractive Music Generators, they capture the information of the whole piece of musical input and create an entirely new piece of music containing new phrases. Phrases are not extracted but generated in this case.

Recent work in Music Generation have either relied on GANs, or an encoder decoder architecture that generates audio either as Midi[2] or as Raw Waveform[4,5].

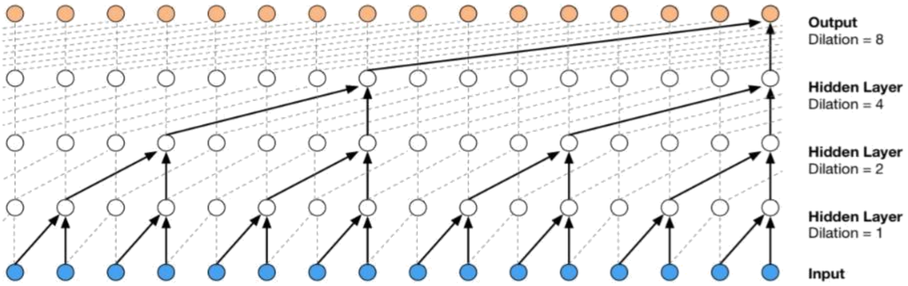


Fig. 1. Convolutional Dilation[5]

GANs[3] have recently performed exceedingly well in generative tasks. WaveNet and it's variants[5] were the first to utilize an end to end seq2seq architecture. The encoder was used to formulate a thought vector based on the past at a point in time. According to this thought vector, future speech or music was formulated with the decoder.

Although this approach is simple and relies on a big network running with high compute power(it took Google's Deepmind 1 hour to output 1 second of data), it is not feasible for music generation because it utilizes low level features which might not have a bearing on the actual musical output.

Dong et al. in MuseGan[3] used a different approach. They trained three different models. The models were GANs which could generate different instruments playing in unison using a set of GANs or a single GAN conditioned on an input vector. This input vector acted as the seed in the network and was used by the different GANs to generate music with the same intent and alignment.

This architecture formed state of the art in music generation and utilized the Lakh Piano-Roll Dataset. They also formulated NLP based metrics to evaluate their models and validate them against a baseline.

Radford et al. introduced a model called GPT-2, a transformer based model in their 2019 paper titled Better Language Models and their Implications[19] which can be easily retrained and utilized for recreating any language sequences with novelty. GPT-2 was trained on WebText, a set of crawled webpages and is a 1.5B parameter model. It outperformed state of the art models in 3 out of 4 language modeling tasks.

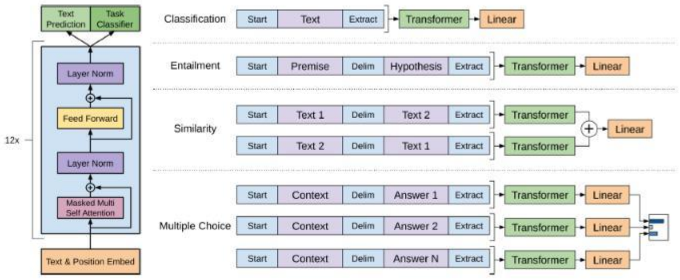


Fig. 2. GPT2 Architecture [19].

The models which have been till now utilized in the space of music generation have been either statistical or neural models. Neural models like the base paper by Dong et al. in MuseGan[3] utilizes a big GAN based neural network trained on a lot of data. However, our model tries to take advantage of the inherent structures of music by interplaying two modules for the harmony and melody together.

III. PROPOSED WORK

We introduce a new model for abstractive generation - MuseNet. It features two main portions, the discriminator for generating the first chord of a bar conditioned on the notes of the bar which came previously and the generator, which generates the notes of a given bar based on the chord as seed note. Needless to say, this approach maintains a good amount of coherent and gives concordant sounds. Furthermore, since it does not involve training a GAN or heavy modules like CTC Loss layers, training is faster.

A. Basic Architecture

The basic architecture consists of two parts, the discriminator and the generator. Our methodology includes a novel approach wherein we shall generate music bar by bar, and the contents of each bar will be conditioned on a chord or set of notes. This chord will be chosen from a prefixed set of chords a song can use, using the discriminator. The generator can use the same chord as an intent vector and generate a bar of music. This shall make the music generated coherent and help it hopefully to have long term dependencies and structures which are inherent in today's music.

The expected results of the project will include novel generated music in a single scale, or raag (as present in Indian Classical Music). Basing our generative model on coherent sources like a scale or chord will yield results which sound better and appeal more.

However, since many bars will be conditioned on the same chords, some continuity and repetition issues may creep in since our model is not a full fledged GAN model. Yet, the advantages it provides, including easier training and less memory cannot be fully ignored.

B. Modules

- **Discriminator**

The discriminator is made up of a MultiLayerPerceptron Network with 5 layers and 64-32-16-8-8 units in each layer.

The output layer has 8 units since that is the number of chords it is predicting for a given song. The same can be changed for different datasets. The model can also be retrained by only replacing the last two or so layers for a different dataset. What essentially this means is that the discriminator can predict with enough certainty which chord follows a certain bar of music. Based on this the next part of the model takes over.

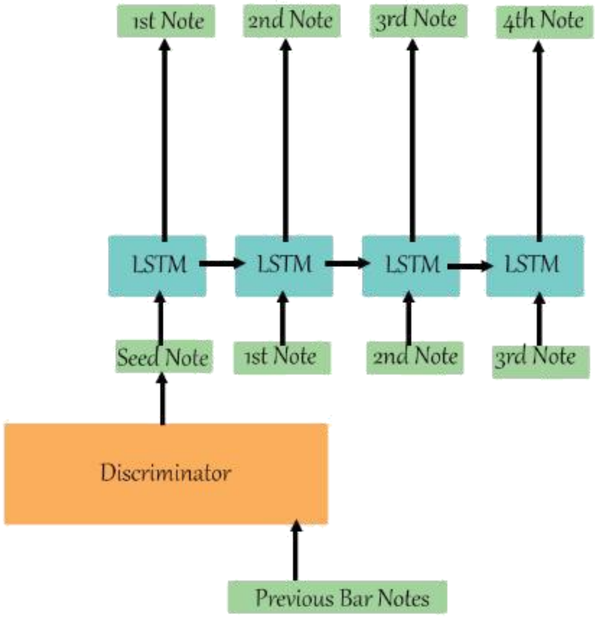


Fig. 3. Main Model Architecture.

- **Generator**

The Generator is a component which uses an LSTM architecture. It receives the entire hidden vector of the discriminator output and using it, generates the output note by note at each an every instance or a timestep. The hidden vector that the decoder uses as input is first injected with a START token at the beginning of the sequence and the an END token is added at the end of the sequence. While decoding this vector, the target sequence is unestablished. The Decoder starts predicting the target sequence by first parsing the START token which signifies the beginning and the end is denoted by the END token, after which it understands that there are no more notes to be processed. Each recurrent unit of the LSTM architecture accepts an element from the hidden vector also known as a hidden state from the previous unit and also produces a new hidden state as well as its own hidden state. The architecture is presented in Fig. 3. We also utilize the GPT2 architecture as shown in Fig 2. as the Generator. This in turn generates more coherent music than the vanilla LSTM since the GPT2 decoder can extremely easily learn long term dependencies, having a large number of parameters in itself.

C. LSTM UNIT OVERVIEW

Our model uses three layers of stacked LSTM(Long Short Term Memory) units as the Recurrent Neural Unit. LSTMs have gained a lot of traction in recent years due to the way they handle the vanishing gradient problem with ease. Our models were trained for about ten epochs each with RMSProp as the optimizer. Using RMSProp instead of traditional optimizers like Stochastic Gradient Descent improves training efficiency and helps the model converge better. The LSTM layer consists of three distinct gates - the Input gate, Forget gate and Output gate.The Input gate informs what new information we’re going to store in the cell state, the Forget gate tells us about the information to throw away from the cell state. The output gate is used to provide the activation to the final output of the LSTM block at a given timestamp.

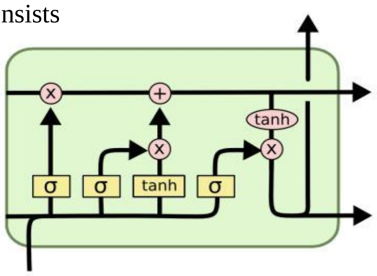


Fig. 4. LSTM Unit [8].

D. ATTENTION LAYER

The attention layer sits on top of the encoder and decoder LSTM layers.

Through training it learns to concentrate more on parts of the output of the previous layer which is mostly required for generating the notes in the next layer using the dense output layer. Without the attention layer, the encoder would have to pass on the entire information about the input sequence using the thought vector, leading to information loss and ambiguity.

The attention layer, as shown in Fig. 2 solves this issue by taking information from the encoder at each step and weighing them in accordance to the decoder's needs

E. THE DATASET

We train and test our LSTM model on the Lakh Piano roll Dataset [9]. It consists of 174,154 multitrack piano rolls derived from the Lakh MIDI Dataset (LMD). Apart from this, we also have to cleanup the dataset, after which we end up with lpd-cleansed containing 21,425 multitrack piano rolls collected from lpd-matched with the following rules. Note that lpd-cleansed contains songs from ALL genres, which is different from the description on the paper. It follows the following rules:

- Remove those having more than one time signature change events
- Remove those having a time signature other than 4/4
- Remove those whose first beat not starting from time zero -Keep only one file that has the highest confidence score in matching for each song

The GPT2 model is trained and tested on the Nottingham Music Dataset[20]. It is an open source dataset of about 1000 British folk songs. The dataset was cleaned using similar means and keywords were generated using the gpt2-simple-keyword library. Afterwards, transfer learning was performed on GPT2 to train it to generate music similar in style to the Nottingham dataset, but original in composition

IV. EVALUATION AND RESULTS

The final model relies on two parts, the Discriminator and the Generator. For the Generator we utilized both LSTMs and GPT2, while the discriminator is a simple LSTM.

The following tables showcase the loss values and outputs for both the models, with simply LSTMs or coupled with GPT2.

A set of output is given as follows in Fig 7, the same was run through a ABC to Midi online software and the following sheet music was obtained, as in Fig 8. These outputs correspond to novel music being generated.

V. CONCLUSION AND FUTURE WORK

Our paper provides a valuable framework for training sim-ple music generation models on text based music notations. Utilization of transformers in the form of GPT2 is novel in case of music generation, alongside the modular Discriminator Generator architecture.

Future work can focus on training bigger networks, capable of outputting longer musical phrases. A weight factor maybe introduced to penalize simpler phrases, thus forcing the model towards longer and more coherent outputs. Models with alter-native recurrent units like GRUs may also be used.

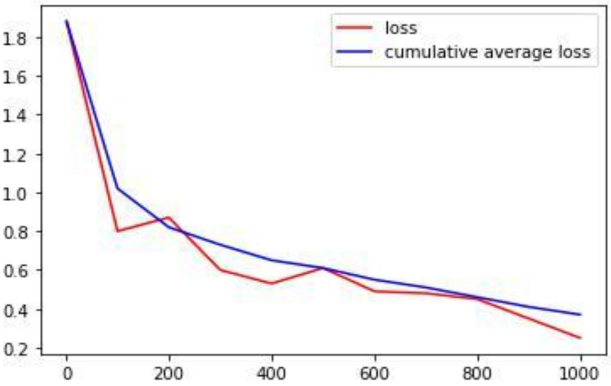


Fig. 5. GPT2 Results

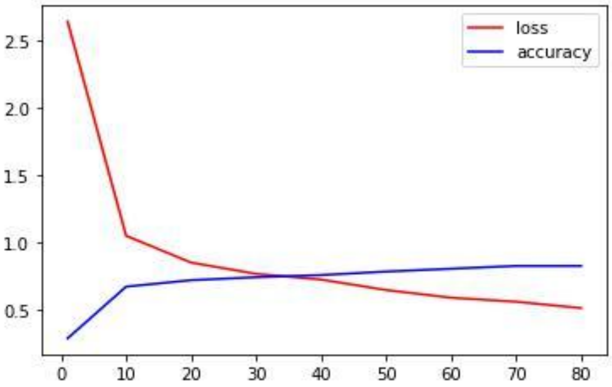
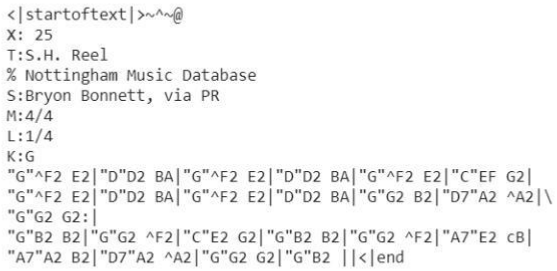


Fig. 6. LSTM Results



```
<|startoftext|>~^~@
X: 25
T:S.H. Reel
% Nottingham Music Database
S:Bryon Bonnett, via PR
M:4/4
L:1/4
K:G
"G"^F2 E2|"D"D2 BA|"G"^F2 E2|"D"D2 BA|"G"^F2 E2|"C"EF G2|
"G"^F2 E2|"D"D2 BA|"G"^F2 E2|"D"D2 BA|"G"G2 B2|"D7"A2 ^A2|\
"G"G2 G2:|
"G"B2 B2|"G"G2 ^F2|"C"E2 G2|"G"B2 B2|"G"G2 ^F2|"A7"E2 cB|
"A7"A2 B2|"D7"A2 ^A2|"G"G2 G2|"G"B2 ||<|end
```

Fig. 7. Sample Output

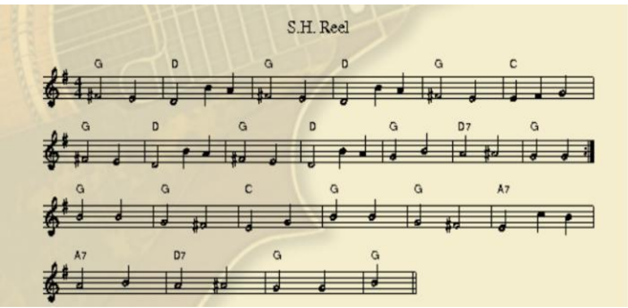


Fig. 8. Sheet Music Representation of Generated Music

Research can also focus on better extractive methods, using a neural approach, which trades runtime for better extractive results. An alternative dataset may also be used.

We may also think about using pretrained embeddings if possible and in the absense of them, train our own music based pretrained embeddings which can be immensely helpful for further work and research.

ACKNOWLEDGMENT

The authors thank Dr Subalalitha C.N , Dept of CSE, SRM Institute of Science and Technology for her support and guidance during this project. We would also like to acknowledge the help we have received from Christopher Olah's Blog on LSTMs[8] during this project.

REFERENCES

1. Deep Learning Techniques for Music Generation - A Survey (Jean-Pierre Briot, Gaetan¨ Hadjeres, Franc¸ois-David Pachet)
2. Convolutional Generative Adversarial Networks with Binary Neurons for Polyphonic Music Generation - ISMIR 2018 (Yang et al.)
3. MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment - AAAI 2018 (Yang et al.)
4. WaveNet: A Generative Model for Raw Audio (Aaron van den Oord, Sander Dieleman et al.)
5. LP-WaveNet: Linear Prediction-based WaveNet Speech Synthesis - ICASSP 2019 (Hwang et al.)
6. Chopra Sumit , Auli Michael , and Rush Alexander M . 2016. Abstrac-tive sentence summarization with attentive recurrent neural networks. In North American Chapter of the Association for Computational Linguistics.
7. Mihalcea, R., Tarau, P. 2004. Textrank: Bringing order into texts. In Lin, D., Wu D. (Eds.), Proceedings of EMNLP 2004, pp. 404–411 Barcelona, Spain. Association for Computational Linguistics
8. Christopher Olah, 'Understanding LSTM Networks', 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Accessed: 20- August- 2019].
9. Music and AI Lab, Academia Sinica , 'Lakh Pianoroll Dataset', 2016. [Online]. Available: https://salu133445.github.io/lakh-pianoroll-dataset/ [Accessed: 20-August- 2019].
10. Pennington Jeffrey, Socher Richard, Manning Christopher D. 2014. 'GloVe: Global Vectors for Word Representation', [Online]. Available: https://nlp.stanford.edu/projects/glove/ [Accessed: 20-August- 2019].
11. [11] Joshi Prateek. 2018. 'An Introduction to Text Summariza-tion using the TextRank Algorithm' [Online]. Available:
12. https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/ [Accessed: 20- August-2019].
13. Devlin Jacob, Chang Ming-Wei, Lee Kenton, Toutanova Kristina. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In NAACL-HLT.
14. Yang Liu. Fine-tune BERT for Extractive Summarization. 2019 in arXiv:1903.10318 [cs.CL]
15. See Abigail. CNN-dailymail dataset. [Online]. Available: https://github.com/abisee/cnn-dailymail [Accessed: 20-August-2019].
16. Sandhaus Evan. The New York Times Annotated Corpus. 2008 [On-line]. Available: https://catalog.ldc.upenn.edu/LDC2008T19 [Accessed: 20- August- 2019].
17. See Abigail,Liu Peter J., Manning Christopher D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. 2017 in arXiv:1704.04368 [cs.CL]
18. Narayan Shashi , Cohen Shay B., Lapata Mirella. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning in Proceedings of NAACL-HLT 2018, pages 1747–1759.
19. Narayan Shashi, B. Cohen Shay , Lapata Mirella. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1797–1807
20. Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario and Sutskever, Ilya. "Language Models are Unsupervised Multitask Learners." (2018): .
21. Eric Foxley , 'Nottingham Dataset', 1997. [Online]. Available: http://abc.sourceforge.net/NMD/ [Accessed: 20- Jan- 2020].

AUTHORS PROFILE

Abhilash Pal, final year student pursuing B.Tech, Computer Science and Engineering from SRM institute of science and technology, has been a member of Institution of Engineering and Technology (IET) and also Association for Computing Machinery (ACM). Deeply involved in computer vision, machine learning and natural language processing. Presented a paper titled "Concisenet : An End to End Model for Topic Generation" at IEEE Madras Section SPC 2019. Currently researching in Reinforcement Learning and Text Generation Methods.

Saurav Saha, final year student pursuing B.Tech, Computer Science and Engineering from SRM institute of science and technology, has been a member of Institution of Engineering and Technology (IET) and also Marine Technology Society (MTS). Significantly involved in underwater robotics and computer vision, machine learning and natural language processing.Presented a paper titled "Concisenet : An End to End Model for Topic Generation" at IEEE Madras Section SPC 2019. Currently involved in research of design and manufacturing of Remotely Operated Underwater Water Vehicles (ROVs)

R. Anita, currently working as an Assistant Professor in Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai. And, completed her PG degree in Sathyabama University, Chennai and UG degree in Manonmaniam Sundaranar University, Tirunelveli. Published a paper titled, "Interpretation of Short Text Using Semantic Knowledge" in the International Journal of Applied Engineering Research, ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 4855-4858. Current research interests is in Natural Language Processing.