Transformer Bard: Music and Poem Generation Using Transformer Models

Dustin Palea
Computational Media
University of California, Santa Cruz
dpalea@ucsc.edu

Hongwei (Henry) Zhou
Computational Media
University of California, Santa Cruz
hzhou55@ucsc.edu

Kapil Gupta
Computational Media
University of California, Santa Cruz
kgupta15@ucsc.edu

ABSTRACT

Machine Learning has shown promising results in media generation. Generating Text and Music are among the various areas of interest because of their complex nature as well as the implicit requirement of a certain aesthetic. We fine-tuned a GPT-2 model for generating poems and used a Self-Attention Transformer model to generate Low-Fidelity (Lo-Fi) music. We then provided an intuitive User Interface for easy access to these models.

CCS CONCEPTS

• **Applied computing → Fine arts**.

KEYWORDS

Transformer, GPT-2, Music Transformer, Poem, LoFi

ACM Reference Format:
Dustin Palea, Hongwei (Henry) Zhou, and Kapil Gupta. 2020. Transformer Bard: Music and Poem Generation Using Transformer Models. In *Conference '20: Proceedings of ACM Conference, August 32nd, 2020, Forest Hills, NY*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Content generation has been a major application in the Artificial Intelligence research. Procedurally generated content includes a variety of mediums including text[8], images[7] and video game levels[4]. These AI-driven creations are not only benchmarks for artificial intelligence research, but also tools for new possibilities in computer art.

We built our project based on two AI architectures for generation: GPT-2[3] for text and music transformer[1] for music. Both architectures utilize transformer layers to perform their generative tasks. GPT-2 allows the user to perform fine-tuning so its output text is similar in style to the given samples.

We combined these two architectures to build a generative poem reader. The aesthetic inspiration is the genre of Lo-Fi music that projects a mostly impressionist, atmospheric and calm mood using mostly electronic sound samples. We situated our music transformer so that it generates the style of Lo-Fi music. In addition, we also fine-tuned our GPT-2 model using poems by Emily Dickinson and Robert Frost. To interact with the generative poem reader, the user is asked for provide a prompt. This prompt is then fed for poem generation. After receiving the output of poem generator and music generator, the program will start reading the poems using text-to-speech while the generative music is played.

2 RELATED WORK

Transformer has its root in Natural Language Processing research[5]. Since Transformer's self-attention mechanism takes a global approach to its sequential data process, the usual setbacks of Recurrent Neural Network such as difficulties with parallelization and long-term dependencies are largely resolved for Transformers. Language model GPT-2, developed by OpenAI, utilizes transformers for text processing[3]. It was trained by using Transformer-decoders to summarize Wikipedia articles. Similarly, Music transformer, developed by Google Brain, also utilizes Transformer to model musical structure [1].

2.1 Interactive Stories

The most notable creative work based on Transformer architectures is *AI Dungeon 2*[6]. *AI Dungeon 2* uses GPT-2 to build a text adventure game that takes any player-entered text as input. Comparing to traditional text adventure games that offer options hard-coded by the game writers, *AI Dungeon 2* showcases the potential of expanding player agency with powerful language models. The play experience of *AI Dungeon 2* does leave a lot to be desired. Specifically, the game doesn't seem to remember the state of the world - dead characters can become alive after several choices, the location of the player can shift from choices to choices. But it is undeniable that *AI Dungeon 2* is a clear indication of the artistic potential that transformers enables.

2.2 Compelling Images

Another notable application of the Transformer Model is its use in image generation. One recent study uses this model in an Image Transformer [2], capable of creating super-resolution images (generatively upsampling low resolution images) and completing incomplete images (by "filling in the blanks"). These tasks are typically done using other network architectures such as RNNs and CNNs. However, as previously mentioned, a major bottleneck is the computational complexity of performing these tasks. RNNs, for example, predict each new pixel value by using all previous pixel values as input. This is why for these and similar tasks, CNNs are commonly chosen due to their parallelizability. Yet still, while at a lower level of computational complexity, these CNNs lack the ability to accurately reproduce more macro level image features (e.g. symmetry, occlusion). One way to address this is to simply grow the receptive field in order to recognize these larger scale patterns, but then we again run into the same efficiency problems as RNNs. The Image Transformer utilizes the self-attention mechanism in the transformer model to train more efficiently and to produce more accurate results, in particular, through replication of larger scale features. This new model offers a few creative affordances. First, through more efficient operation, applications can be deployed on less powerful hardware and are able to offer semi real-time operation. Second, the accuracy of this new model opens creative spaces with more compelling artwork. Prior to the transformer model achieving these characteristics was impractical, especially for the average creative.

3 METHODS

This section is divided into three subsections: poem generation, music generation and web interface. Each subsection was developed independently and then integrated.

3.1 POEM GENERATION

GPT-2 355M-hyper-parameter version is used for the poem generator. To match the aesthetic style of LoFi music as well as to ease the difficulty of data gathering, we collected all of Emily Dickinson's and Robert Frost's poems as fine-tuning data. A primer, or a given prompt, is provided to the generator. The generated text is continuation of the primer. A generated example is as follows, the primer is omitted:

XVI.

FRIENDS.

Is Heaven a lodging?
Could anybody stay
Arrangement;
Who stayed there could stay,

Nor visitors?
Could anybody leave
Without my consent, –
Who stayed there could stay,
Nor visitors?

XVII.

THE BLUE JAY.

No brigadier throughout the year
Would dare to cross my turf;
But when the Colonel
Forth from field to field,

With artillery and drivers,
This imperceptibly cabined creature

Becomes unmannerly all the year,
Till January, when there comes a
Bliss from the prairie.

XVIII.

The devil has a Blue Jay –
He knows no other way;
He is as much blotted out
As if he had signified.

The sod has a blue jay;
He knows no other way;
He is as much blotted out
As if he had signified.

The sun has a blue habit;
He self-consciously adorns
With scarlet thread such ways
I have not been.

The sod has a demand of thee;
Signatory to everything
Exists in heaven.
Where? In heaven?

That trespasses so wordlessly There,
I will not pass.

The wicked sauntered in before,
The law remained still;
Now, I will make my appearance
Before thee, –
Thou shalt have power to obey,
Or death.

XIX.

THE WOOD POTATO.

This is a common bean, –
A shoot equal parts hog and gator;
Not all bean types the same.
Butterflies would stretch
Their hock like fools did the hog,
And bugle boys and girls
Would play at their stools, –
Butterflies, not of this world,
Dissolve without a fee.

Since the generated output can cut off mid-poem, the output is processed so it contains complete poems. The generated output is sent to a text-to-speech server hosted by Streamlabs. Streamlabs is a software mainly used to support livestreaming on *Twitch.tv*. It features a rather robust and high-quality text-to-speech feature used for voicing donation messages. A request containing the generated text is sent to the Streamlab server. The audio file is returned, ready to be played on the web interface.

3.2 MUSIC GENERATION

The Self-Attention Transformer used for Music Generation is trained on Piano Roll dataset. We used Erik Satie's pieces with the pre-trained network to generate music with the Low-Fi aesthetic. We used parts of Eric Satie's pieces as primer that the music transformer used as base to generate the rest of the piece in continuation. We then removed the original piece to just use the model's output for our user interface.

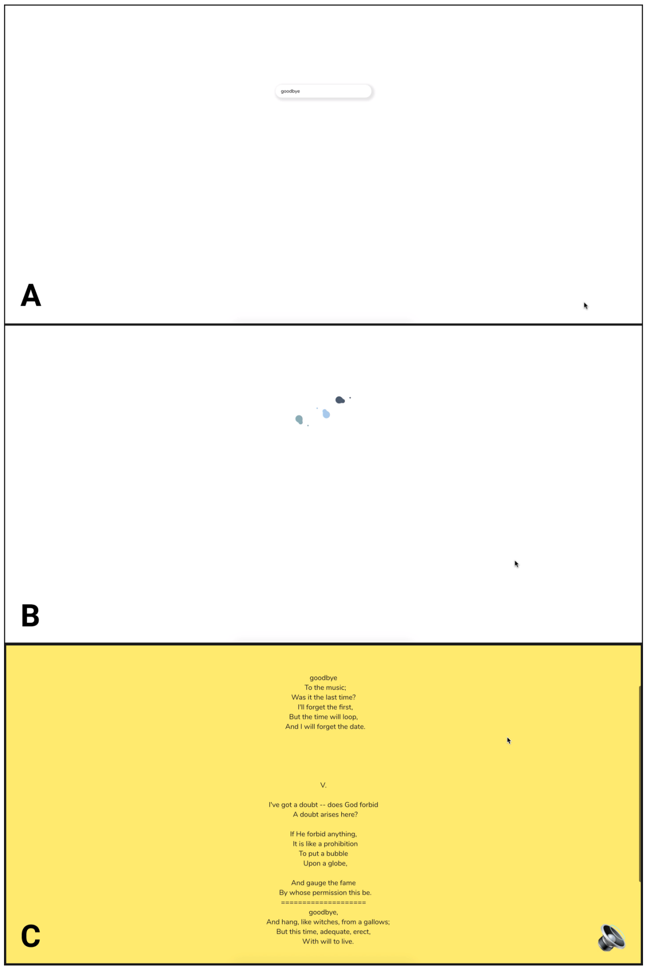
3.3 INTEGRATION THROUGH A WEB INTERFACE

Our user interface combines the results of the previous two subsections into one coherent piece. We host a web server locally so that we may bridge browser interactions to the machine learning models and allow users to interact with them. On the client side, our website allows the user to input some text prompt which is sent to the web server API using the POST method. On the server side, we extract the text data from the incoming request and fork a new child process. In this child process, we execute a simple script that takes in the text input as a parameter and feeds it to a Python program that implements GPT-2. This program generates the poem and writes it out to a text file. First, we read in this file and send its contents through a sentiment analysis package (npm sentiment) that outputs a sentiment score, an integer with positive values indicating a positive sentiment and vice versa. Second, we send the file's contents to a TTS API as described above which returns a source URL to a rendered audio file. Meanwhile, while we are generating the poem and fetching the speech audio, we also generate a song using music transformer and output a rendered WAV file. Finally, we return the poem text, sentiment score, and speech source URL to the client, and host the generated song on our server for the client to fetch.

In our project there are a total of four audio files: 1) the poem speech, 2) the generated song, 3) happy bird atmosphere, and 4) sad storm atmosphere. After the above process returns, we appropriately play either the happy or sad atmosphere based on the sentiment score. After a few seconds we begin playing the song, and then the speech while the poem text is displayed on screen and the background color is changed to match the mood. The above process produces the speech and song audio, while the atmosphere sounds are static files. When a file begins playing, it is faded-in by linearly increasing the volume attribute on the audio element over some transition time (the opposite is done when sound files are stopped). On each run of the program, we dynamically replace the TTS source URL and overwrite the song file. We disallow browser caching by changing cache headers on any file responses in order to ensure that the latest audio files are always used.

4 RESULTS

The resulting UI presents the user with a text input field into which they may enter a poem prompt (*Fig. 1, A*). The user presses the enter key to submit the text and the program begins processing (*Fig. 1, B*). When loading is complete, the background color changes either



goodbye
To the music;
Was it the last time?
I'll forget the first,
But the time will loop,
And I will forget the date.

V.

I've got a doubt -- does God forbid
A doubt arises here?

If He forbid anything,
It is like a prohibition
To put a bubble
Upon a globe,

And gauge the fame
By whose permission this be.
====================
goodbye,
And hang, like witches, from a gallows;
But this time, adequate, erect,
With will to live.

Figure 1: A) user entering poem prompt "goodbye", B) real-time generation of poem and music in progress, C) generated poem shown over a positive background while the poem speech, music, and atmosphere audio plays

to a light yellow or a dark blue. According to the mood, either atmosphere audio fades in and soon the generated song begins playing. The poem is presented on screen in a color that contrasts with the background and the poem is read in a child's voice (*Fig. 1, C*). At any point, the user may stop all audio and begin a new generation process by pressing the enter key. This resulting UI brings together multiple forms of computer generated artifacts for human enjoyment or creative augmentation.

5 DISCUSSION

5.1 Poem Generator

The generator seems to plagiarize the original works a lot when temperature is below 0.8. Only when the temperature is raised as high as 0.9 the plagiarism stops. This might be an indication that there's not enough training data for the generator to generalize.

5.2 Music Transformer

The current method is capable of generating only single-instrument based pieces. There is no api available to retrain the pre-trained Music Transformer model on our own dataset. The pretrained network also requires a long and coherent primer. In experiments with short but coherent primers with a distinct aesthetic, the model was able to maintain the coherency for most of the piece. While long primers without a coherent structure didn't allow good results.

6 FUTURE WORK

Given the tight time frame for this project, the final product leaves a lot to be desired. There were many technical and artistic ideas that can improve the use experience but were not implemented due to time and technical constraint.

On the technical side, it was originally envisioned that the generators would run indefinitely. The original inspiration of this work is Lo-Fi music station on YouTube, which viewers can simply leave on similar to a radio station. One possible implementation is to allow the Poem Generator and Music Transformer to generate a block of audios. And while those audios are playing, the generators will generate the output for the next time block indefinitely. To keep the output somewhat coherent, parts of the text and music output can be used as primers for the generators.

On the artistic side, the program can be better contextualized. One idea is inspired by the video game *Kind Words*. In *Kind Words*, the player receives letters from other unknown players and writes reply back to them. The game is served as a platform for players to talk about their daily struggles and help others to get through them. The idea is to frame the program similarly, only that the letters the user receives are randomly selected among pre-written ones by the developers. The player will have to provide a primer to address the letters. This interaction scheme is designed to set the mood for the player to provide primers appropriate to the Lo-Fi style of the generated poems and music.

REFERENCES

[1] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, and Douglas Eck. 2018. An Improved Relative Self-Attention Mechanism for Transformer with Application to Music Generation. *CoRR* abs/1809.04281 (2018). arXiv:1809.04281 http://arxiv.org/abs/1809.04281

[2] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image transformer. *arXiv preprint arXiv:1802.05751* (2018).

[3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[4] Julian Togelius, Georgios N Yannakakis, Kenneth O Stanley, and Cameron Browne. 2011. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3, 3 (2011), 172–186.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[6] Nick Walton. [n.d.]. AI Dungeon 2. https://aidungeon.io/

[7] Zhengwei Wang, Qi She, and Tomas E Ward. 2019. Generative adversarial networks: A survey and taxonomy. *arXiv preprint arXiv:1906.01529* (2019).

[8] Ziang Xie. 2017. Neural text generation: A practical guide. *arXiv preprint arXiv:1711.09534* (2017).