MusicAI: Music Generation for Multiple Genres and Novel Sampling Techniques with Transformer-XL

Danielle Kutner, Martin Lim, Kenneth Lee, Natan Vidra, Bar Kadosh, Ben Kadosh

Abstract

Our research built upon the state of the art "Pop Music Transformer: Generating Music with Rhythm and Harmony" research done by Yu-Siang Huang and Yi-Hsuan Yang of National Taiwan University.[1] In particular, we focused on two key aspects. First, training the model on additional music of different genres (classical, jazz, contemporary pop, and contemporary rock). We used our generated samples as inputs into a genre classifier to measure whether our model was generating tunes in the genre we specified. Second, we experimented with different sampling techniques when generating a tune. While previous research used the first 4 bars of a reference tune as a starting sample, we experimented with using 2 and 8 bar samples, as well as sampling from both the middle and the beginning of the song. Our classifier predicted the correct genre of AI generated songs with an accuracy of 60.1%. In addition, when surveying others, only 13% of people were able to identify the AI-generated song out of a group of human-generated and AI-generated songs. Approximately 70% of respondents also signified that their favorite generated song was generated using the 8 bar-middle sampling technique.

1 Introduction

AI generated music has been around for several years and has seen many interesting and innovative changes in its evolution. Architectures featuring RNNs, LSTMs, GRUs, and CycleGANs have been built upon to create the Transformer and Transformer-XL neural network architectures. Introduced by Google AI in 2017 and 2019 respectively, these architectures have significantly improved the quality of music generated and addressed one of the major issues with music generation: the ability to model long-term dependency.

Work over the years has also focused on better understanding music interpretation by humans to better assess the quality of AI generated music and making it sound natural and less robotic to humans. In efforts to tackle the robotic sound of AI generated music, Malik (2017) developed a network that was heavily trained on note velocity data.[2] Velocity in the context of music can best be defined as: “the force with which a note is played, and it is vitally important in making MIDI performances sound human - or if you use a fixed velocity, making them sound mechanical.”[3] In order to be “consumable” for humans, models have advanced to change the velocity of the generated notes in a way that makes the song sound more fluid and less robotic. These advances have heavily improved AI-generated music, making it much more relevant today.

Using Neural Networks to generate new music in different genres can have numerous applications in the real world:

- Inspire musicians to broaden their styles and produce more creative content
- Enable musicians to appeal to broader fan-bases and genres they may not be familiar with
- Help singers create melodies based on their style (through a 4-8 bar sample of their work)
- Reduce costs by providing businesses with royalty-free music, unrestricted by copyright
- Create new songs from bands of the past for fans in the present
- Provide free alternative music for local businesses

2 Background

The improvements seen in the field of music generation have been a byproduct of the advances in natural language understanding and translation and the introduction of new neural network architectures. Like language, music has patterns and long-term dependencies that influence its structure and sound. The state of the art today has built upon the Transformer Architecture introduced by Google AI in August of 2017 with their submission of the research paper "Attention Is All You Need" to NeurIPS.[4] In a Google AI blog post discussing the report, Jakob Uszkoreit (a member of the research team) explains some of the limitations of RNNs and why the team introduced a novel architecture,

> RNNs have in recent years become the typical network architecture for translation, processing language sequentially in a left-to-right or right-to-left fashion. Reading one word at a time, this forces RNNs to perform multiple steps to make decisions that depend on words far away from each other. The sequential nature of RNNs also makes it more difficult to fully take advantage of modern fast computing devices such as TPUs and GPUs, which excel at parallel and not sequential processing.[5]

The research team identified two major limitations of RNNs: the sequential nature of processing data and the resulting struggle of models to retain long term information, as well as the performance limitations resulting from RNNs struggling to fully utilize modern hardware. And so, the team introduced the Transformer architecture which "only performs a small, constant number of steps (chosen empirically). In each step, it applies a self-attention mechanism which directly models relationships between all words in a sentence, regardless of their respective position."[5]

The key here is the self-attention mechanism, which has two major benefits. The first is that it allows us to focus more attention on certain words or notes; it allows us to better identify the essence or driving force behind a sentence or bar of music. The second is that it allows for modeling the relationship between all words or all notes in a sentence or bar, regardless of position. The Transformer overcomes the limitations of a sequential model "by generating initial representations, or embeddings, for each word. Then, using self-attention, it aggregates information from all of the other words, generating a new representation per word informed by the entire context. This step is then repeated multiple times in **parallel** for all words, successively generating new representations."[5] Modeling a direct relationship irrespective of position for each word or note addresses many of the limitations of RNNs.

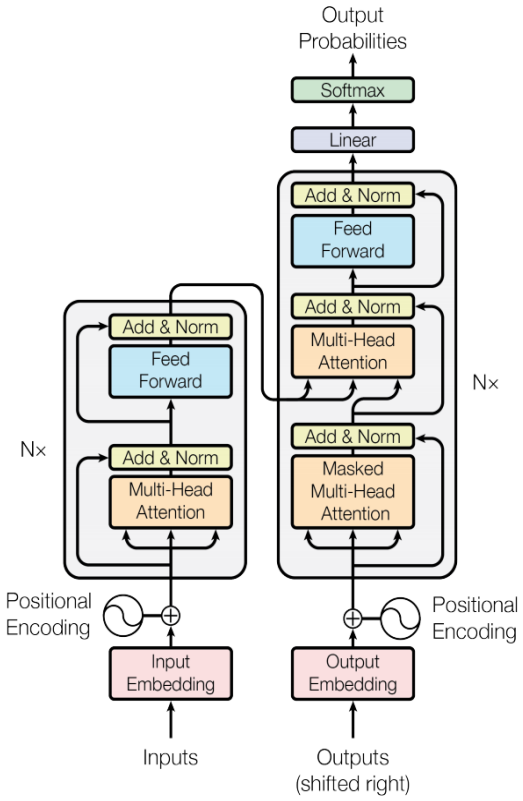

Figure 1: Transformer Architecture [4]

And while the introduction of the novel Transformer architecture redefined the state of the art, it still had its own limitations resulting from the fixed-length nature of the Transformer. By converting sequences into segments of fixed length, the Transformer struggled to model dependencies that were longer than the predefined fixed length of the model and resulted in fragmentation issues. And so, in January 2019, Google AI built on its previous work and introduced Transformer-XL. To overcome these limitations, the team proposed an architecture that would "consist of two techniques: a segment-level recurrence mechanism and a relative positional encoding scheme."[6]

For the segment-level recurrence mechanism, "representations computed for the previous segment are fixed and cached to be reused as an extended context when the model processes the next new segment. This additional connection increases the largest possible dependency length by N times, where N is the depth of the network. Moreover, this recurrence mechanism also resolves the context fragmentation issue, providing necessary context for tokens in the front of a new segment."[6] However, a segment-level recurrence mechanism on its own did not suffice, as positional encodings did not flow properly from the prior segments to the current segment. To address that issue, the team introduced a new relative positional encoding approach where "[the] formulation uses fixed embeddings with learnable transformations instead of learnable embeddings, and thus is more generalizable to longer sequences at test time."[6] What's particularly interesting here is that while both of these techniques do not work well alone, they work incredibly well when combined. That is to say, the combination of the two techniques has a multiplier effect such that the new architecture is significantly more effective at modeling long-term dependencies. For context, the research found that "Transformer-XL learns [a] dependency that is about 80% longer than RNNs and 450% longer than vanilla Transformers, and Transformer-XL is up to 1,800+ times faster than a vanilla Transformer during evaluation on language modeling tasks, because no re-computation is needed."[6]

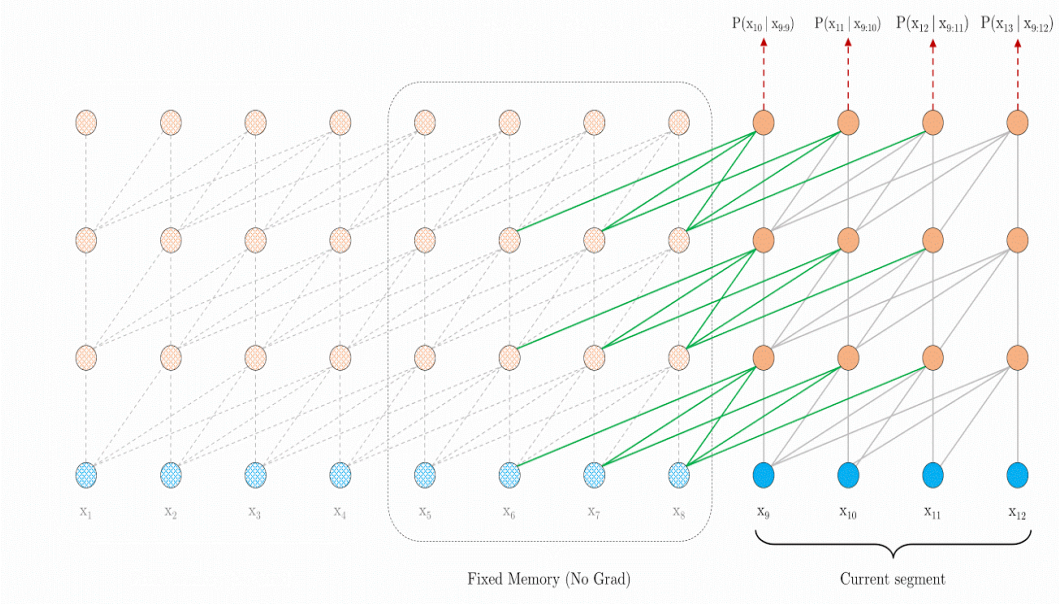

Figure 2: Segment-Level Recurrence for Transformer-XL [6]

While machine learning models have advanced significantly in this space, they still struggle with understanding the rhythm and structure of the music they generate. Recent research has started focusing more on the process of converting music into a more effective input format to address these limitations. As the age old saying goes, garbage in garbage out. In their paper, Huang and Yang look to improve the quality of the generated music by altering the way the input data is handled and understood. Instead of using MIDI files, the standard in this space, they develop what is known as a "REMI" file.[1]

MIDI is short for Musical Instrument Digital Interface. It is a protocol that allows computers, musical instruments and other hardware to communicate. MIDI never transmits an actual audio signal (it includes information only) and MIDI messages are sent as a time sequence of one or more bytes. The first byte is a "Status" byte, often followed by "Data" bytes with additional parameters. MIDI files are heavily based on "events," such as "Note-On" (start of the note) and "Note-Off" (end of the note); the "Note-On" message is sent when the performer hits a key of the music keyboard. It contains parameters to specify the pitch of the note (the number of the note played), as well as the note's velocity (intensity of the note when it is hit). When a synthesizer receives this message, it plays that note with the correct pitch and force level (volume). When the "Note-Off" message is received, the corresponding note is switched off by the synthesizer. When processing MIDI files, machine learning models ultimately have to learn how to interpret these events to develop a structure and rhythm.

Although MIDI files are the standard, and though not many have strayed from this standard, Huang and Yang are not the first to explore alterations to this standard. In his paper, Spitael (2017) proposes a novel method for vectorizing MIDI files; instead of a vector representing the complete state of the keyboard, one pitch (and its velocity and lapsed time) will be encoded at a time.[7] However, the "REMI" files Huang and Yang develop specifically attempt to address the difficulties that language-based models have with musical structure.[1] These files build upon MIDI files, but add additional markers/fields to make it easier for models to understand an explicit musical structure. While MIDI files use the "Note-On" – "Note-Off" structure to understand when and how long to play notes, "Note-On" or "Note-Off" events can get paired incorrectly. The REMI files explicitly state the duration of a note rather than letting the model interpret it. MIDI files also discard "Bar" and "Position" events; Huang and Yang believe this leads to a lack of structure and rhythm and to melodies overflowing into bars they do not belong in. "Position" explicitly states the note's incremental location within a bar

and the "Bar" event creates a unit structure, such that a group of positions are grouped into bars.[1] These events make it easier for the model to understand the structure of music.

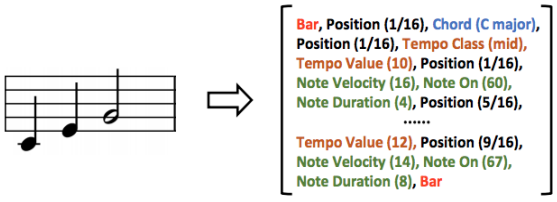

Figure 3: REMI Musical Structure Representation [1]

As can be seen, each grouping of notes is separated by bars. In addition to the "Position", "Note-On", and "Duration" events discussed, each node also has "Tempo" and "Velocity" events. The "Tempo" value makes it easier to naturally change tempo on a more granular level. The REMI file also includes a "Chord" event that enables easy handling of chords.

After training their model, Huang and Yang produce musical outputs by sampling the first 4 bars of an existing MIDI file and feeding that input into the model.[1] The model then produces the rest of the bars of the tune (number of bars to be produced is specified by the user). Selecting the first 4 bars from the beginning of the song seems to be an arbitrary decision and we believe alterations can be made to enhance the quality of tunes produced. We have worked to expand upon Huang and Yang's work, particularly around extending the results to additional data sets and to sampling the initial input into the model differently.

3 Process and Work

As Huang and Yang state in their paper, "A total number of 775 pieces of piano music played by different people is collected...they are covers of various Japanese anime, Korean popular and Western popular songs, playing only with the piano."[1] Their novel work with REMI files is successful at generating music with the piano data they collect. However, one area we explored was whether using REMI files (with their model) also successfully generated music when trained with other data sets and genres. We trained the model with the REMI structure by collecting data for the following musical genres: pop, classical, jazz, and rock. We then evaluated whether the model successfully generates music for all of these genres.

While conducting our research, we found that it was hard to locate MIDI files that are strictly piano, especially for contemporary music. We leveraged our knowledge of musical tools such as Musescore, Garageband, and GB2MIDI to access and alter MIDI files. To expand our access to data, we downloaded MIDI files that had many tracks. We used Musescore and Garageband to manually remove unnecessary tracks and convert non-piano tracks into piano tracks. We then used GB2MIDI to covert these files back into the proper MIDI input format. This allowed us to access new and unique data that Huang and Yang may not have had access to if they only pulled piano pop files.

The second research question we explored was related to generating music with a starting sample. To do this, Huang and Yang use the first 4 bars of a sample song as a prompt when generating music. However, we propose that this may be an arbitrary choice and that other choices may yield better results. We propose trying 2, 4, and 8 bar samples, both from the beginning and middle of a song, when generating a song with a starting sample. We will compare the 6 generated

songs (2 bar beginning, 4 bar beginning, 8 bar beginning, 2 bar middle, 4 bar middle, and 8 bar middle) to determine if major differences arise. We hypothesize that providing more bars when sampling will give the model more context of the song and generate a better tune. Additionally, we believe that the middle of a song might provide more variety and produce a better tune as well. However, it is possible that sampling the middle of a song will make it harder to generate a song with a traditional "intro" and that people might find this off-putting. Regardless, we believe that there is value in exploring this research question and identifying if there is a better sampling procedure.

As part of our implementation, we used the following softwares/libraries/tools: TensorFlow, MIDI toolkit, Python, Garageband, Musescore, and GB2MIDI.

4 Data Sets

In addition to the baseline training dataset, we used several other datasets to train and test different genres. We used the MAESTRO dataset, which is a dataset composed of over 200 hours of classical piano performances provided by the Magenta project.[8] We also accessed classical and jazz MIDI files from two Kaggle competitions.[9,10] The datasets include music from artists such as Beethoven, Bach, Mozart, and Chopin, and Duke Ellington, Freddie Hubard, and Dave Brubek, respectively.

In addition, we used a subset of the Million Song Dataset to access contemporary music.[11] Using this subset, we created additional sets of contemporary pop and rock music. To access additional songs that we believed would be good for training, we used files from UC Irvine's School of Information and Computer Sciences, which offers MIDI files free for download.[12] We also used freemidi.org.[13] For the pop and rock genres, we sampled music from artists such as Coldplay, Vanessa Carlton, Sara Bareilles, and Sam Smith, and Oasis, Journey, The Beatles, and Guns N' Roses, respectively.

5 Evaluation Method

For a quantitative evaluation, we generated new songs using our model trained on different genre datasets. We then used a classifier to evaluate whether our AI-generated songs were predicted to be of the genre they were intended to be. This gave us a better indication of whether we succeeded in creating songs that are genre-based; it also helped us validate whether the model can be used to create different genres of music and not only pop songs.

Second, we utilized human evaluation to assess the quality of the music produced. Ultimately, our goal was to produce music that is both pleasing to listen to, as well as not discernible from real music. Regarding the sampling research question, we also hoped to measure which form of sampling people enjoyed most. To do this, we sampled 25 individuals and played several songs for them, some original songs and some generated songs.

Our qualitative evaluations included two separate aspects:

1. Ability to correctly identify if a song is an original or a generated song and ability to identify the genre of a generated song

2. Preferences for generated songs using different sampling methods

6 Results

As a first step, we hoped to quantitatively measure whether we could successfully train the model to generate music by genre. To do this, we trained a classifier on a training set of real pop, jazz, classical, and rock songs. We then provided both real and generated samples for each of the four categories as test inputs into the trained classifier and classified them by genre. The real songs achieved a mean per class accuracy of 59.89% and the generated songs achieved a mean per class accuracy of 60.16%. The classifier seemed to better classify jazz and rock songs, and struggled more with classical songs. Given that we classified 4 different genres of music, a test accuracy of 60% is significantly better than random classification, which highlights that the classifier is able to detect differences by genre of music. This means that our model can be trained on different genres.

While an accuracy of 60% is reasonable, we recognize that there is room for improvement. First, we had to spend many hours processing and converting online MIDI files into piano-only format; this made it difficult to have a large training set for each genre. Additionally, a lot of genres are defined by other instruments (guitar for rock, trumpet/saxophone for jazz, etc.); isolating only the piano tracks of songs, or even converting melodies from other instruments to piano tracks, may have made it difficult to truly recognize the differences between genres of music. Given that the classifier produced similar results for real inputs and generated inputs, we feel confident that the lower accuracies may be due to the effects of the issues discussed above.

Ultimately, music is a very subjective medium. We distributed surveys to 25 respondents to measure several factors: ability to differentiate between real music and AI-generated music, ability to identify the intended genre of an AI-generated sample, and preferred sampling method for generated songs (beginning vs. middle, 2 vs. 4 vs. 8 bars).

Our respondents were friends, family, and people from different Facebook groups. Some of them are music experts; however, most of them are average music fans (not experts) that enjoy listening to music in their spare time.

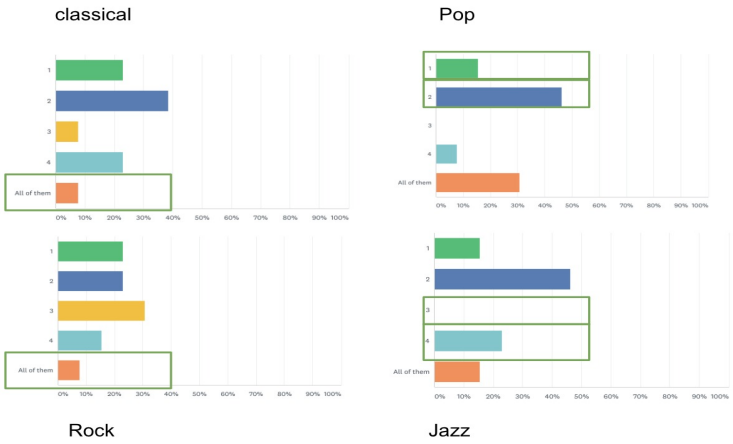

Figure 4: Identifying Human-Generated Music vs. AI-Generated Music Results

As seen in figure 4, we asked respondents to listen to different songs and identify the AI-generated song(s). The figure highlights the correct answer with a green box (can have multiple correct answers) and the bars show the frequency of responses provided. In almost all cases, individuals greatly struggled with identifying the AI-generated songs. In the pop case, individuals successfully identified one of the AI-generated songs, but struggled with identifying the other AI-generated song. This signals to us that our model is generating music that is indiscernible from real music, even when we train it with new genres.

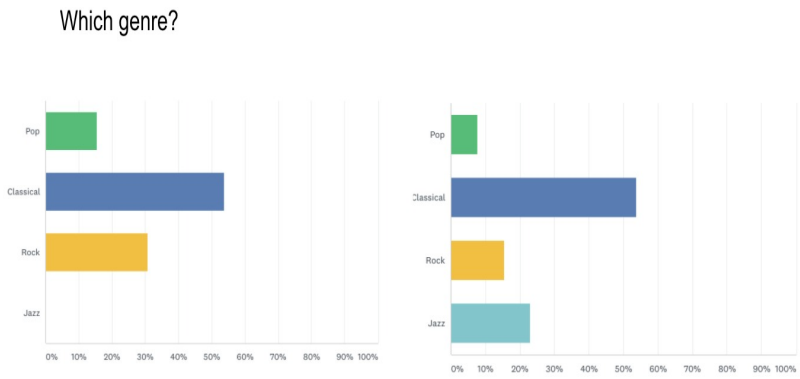

Figure 5: Identifying Genre of AI-Generated Music Results

Figure 5 highlights the results of asking respondents to identify the genre of a generated sample. As can be seen, respondents tended to identify both songs as classical, even though they were Rock and Pop, respectively. One potential reason for this is that classical music is generally associated with piano, while the other genres are not. Since our generated samples were piano-only, it is possible that peoples' perception of the genre they were listening to was skewed by the piano's tone, rather than the pitches, rhythms, melodies, and other factors.

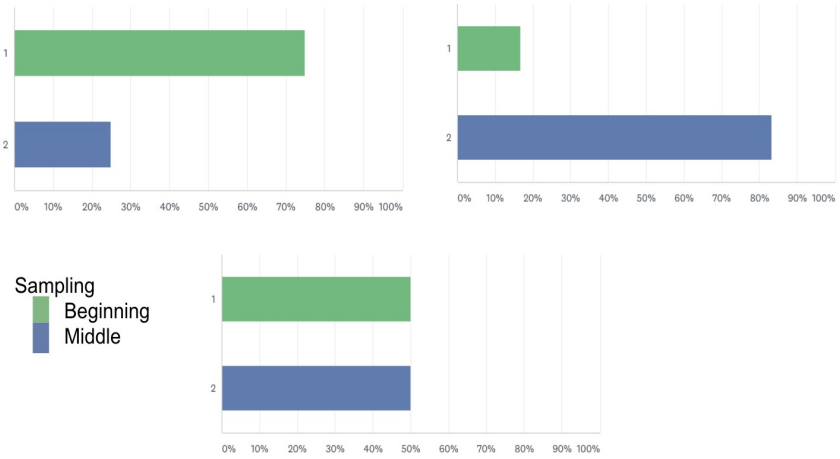

Figure 6: Preferred Sampling Beginning vs. Middle Results

Figure 6 highlights the results of measuring whether individuals preferred music that was generated with a sample from the beginning of a song or from the middle of a song. Ultimately, the results seem to be fairly inconclusive, as preferences seem to be fairly split. This may truly be due to preference: sampling from the beginning of songs seems to generate songs with more traditional "intros", whereas sampling from the middle of songs seems to generate songs with more variety and excitement. Generated songs may also heavily depend on the specific song being used to sample, as the middle and beginning sections may vary from song to song.

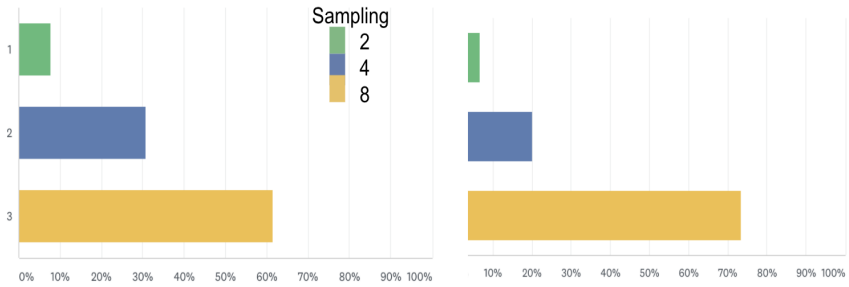

Figure 7: Preferred Sampling 2 vs. 4 vs. 8 Bars Results

Figure 7 displays users' preferences for songs generated using 2 bar vs. 4 bar vs. 8 bar samples. As can be seen, respondents seem to heavily favor samples generated using an 8 bar sample, and generally do not prefer the 2 bar samples. We suspect that providing an 8 bar sample gives the generator more context of the song, allowing it to generate a more realistic and interesting song. This is especially relevant for the intro of a song; 4 bars may not be enough to capture the essence of a song, as songs usually take time to develop.

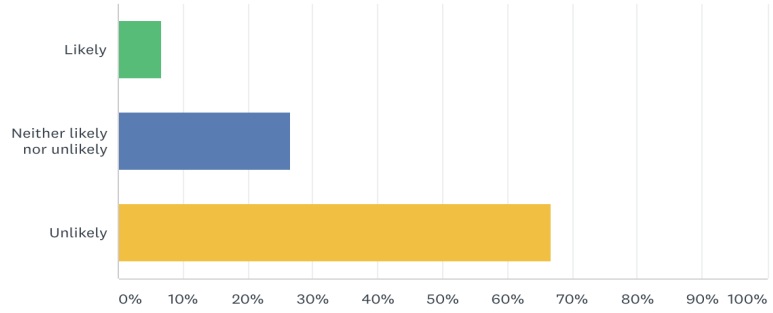

Figure 8: Paying for AI-Generated Music Results

In order to answer the question of whether AI-generated music will eventually replace human-generated music, we asked respondents if they would be willing to pay for AI-generated music. Although AI generating music models have made great strides, figure 8 suggests that further improvements in generation, coupled with a change in the way humans grasp music, need to be made.

7 Conclusion

As our research results suggest, the Transformer-XL architecture (in combination with REMI files) generates quality music that is indescribable from real music for additional genres outside of pop, specifically jazz, rock, and classical. Regarding sampling methods, the results indicate that respondents did not have a strong preference between songs generated using beginning samples and middle samples. However, respondents did clearly prefer songs generated using 8-bar samples, as opposed to 4-bar and 2-bar samples. We believe this is a big step forward in improving upon Huang and Yang's work to generate better quality music from many different genres. For future work, we recommend expanding upon Huang and Yang's research by enabling input files with more than one instrument. As previously noted, we believe that using only one instrument to generate music limits the ability of the model to truly capture the essence of a genre and generate music accordingly. We believe that expanding the functionality of the model to accept MIDI files with multiple instruments will significantly improve the quality of the music generated, especially music of different genres.

8 References

[1] Yu-Siang Huang, Yi-Hsuan Yang, Pop Music Transformer: Generating Music with Rhythm and Harmony, Graduate Institute of Networking and Multimedia, 1 Feb 2020. National Taiwan University https://arxiv.org/pdf/2002.00212.pdf

[2] I. Malik, C. H. Ek, "Neural translation of musical style", CoRR, vol. abs/1708.03535, 2017.

[3] Jones, Hollin. "MIDI Explained: What Is Velocity?" Ask.Audio, Ask.Audio, 5 Dec. 2017, ask.audio/articles/midi-velocity-what-it-is-how-it-works?fbclid=IwAR1i0JZ4ZDk33TmSRGV3_eGhVuSzILXcPd80nKM7fOo6OleqM0226uaYIWo.

[4] Ashish Vaswani,Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Attention is All you Need, 12 June 2017. https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[5] Uszkoreit, Jakob. "Transformer: A Novel Neural Network Architecture for Language Understanding." Google AI Blog, 31 Aug. 2017, ai.googleblog.com/2017/08/transformer-novel-neural-network.html.

[6] Yang, Zhilin, and Quoc Le. "Transformer-XL: Unleashing the Potential of Attention Models." Google AI Blog, 29 Jan. 2019, ai.googleblog.com/2019/01/transformer-xl-unleashing-potential-of.html.

[7] Mathieu De Coster, "Polyphonic music generation with style transitions using recurrent neural networks," A UGent masters dissertation, 2017

[8] "The MAESTRO Dataset." Magenta, 29 Oct. 2018, magenta.tensorflow.org/datasets/maestro.

[9] Sasank, Sai. "Jazz ML Ready MIDI." Kaggle, 27 Nov. 2018, www.kaggle.com/saikayala/jazz-ml-ready-midi/discussion/79251.

[10] Rakshit, Soumik. "Classical Music MIDI." Kaggle, 17 May 2019, www.kaggle.com/soumikrakshit/classical-music-midi.

[11] "Million Song Dataset." Welcome! | Million Song Dataset, 2011, millionsongdataset.com/.

[12] Hirschberg, Dan. "Best Classical Rock Midi." Best Classical Rock Midi, 2009, www.ics.uci.edu/~dan/midi/rock/index.html.

[13] "Free Midi Files Download." Free Midi - Best Free High Quality Midi Site, freemidi.org/.

Appendix

Github Repo with Code: `https://github.com/kenneth-id/band`

Google Drive (contains data, output midis, models, etc.): `https://drive.google.com/drive/folders/1SLdfsR2znIbWBme9XkFDyus4v_oWiaZQ?usp=sharing`

*To access all of our output samples, please access the "output_midis" folder

Soundcloud with Generated Music: `https://soundcloud.com/daniellekutner/sets/band`

Cornell Tech Deep Learning Team Tribute: `https://www.youtube.com/watch?v=mAckKbwgYq4`