JOURNAL OF
EMERGING INVESTIGATORS

Article

Open Source RNN designed for text generation is capable of composing music similar to Baroque composers

Zoya Goel and Cristi Lebron
[1] The Walker School, Marietta, Georgia

SUMMARY

Recurrent neural networks (RNNs) are useful for text generation since they can generate outputs in the context of previous ones. Baroque music and language are similar, as every word or note exists in context with others, and they both follow strict rules. We hypothesized that if we represent music in a text format, an RNN designed to generate language could train on it and create music structurally similar to Bach's. We used an RNN designed for text generation and trained it on 107 Bach pieces. Unlike previously implemented algorithms, such as Bachbot, this algorithm trained on all types of Bach's music-both his instrumental/orchestral pieces and chorales. Since the dataset used in this study is more diverse than that of previous algorithms, we predicted that the present algorithm's output will have a structure that is more reflective of all of Bach's music. We compared the $\frac{Attacks}{Spaces}$ and the $\frac{Spaces}{Attacks+Spaces}$ ratios of both Bachbot's and this experiment's outputs and found that Bachbot's $\frac{Attacks}{Spaces}$ ratios were significantly lower than this experiment's outputs ($p < 0.001$), and their $\frac{Spaces}{Attacks+Spaces}$ ratios were significantly higher ($p < 0.001$). Overall, we found that the music generated by our RNN shared a similar structure with Bach's music in the input dataset, while Bachbot's outputs are significantly different from this experiment's outputs and thus are less similar to Bach's repertoire compared to our algorithm. Therefore, it is plausible that an RNN designed for text generation could create Baroque music.

INTRODUCTION

There are many practical applications for artificial intelligence (AI) to generate music. One reason for doing this is to provide royalty free music to other content creators. Another is that it might provide inspiration to aspiring composers. However, the most relevant application would be in the classical music world. There are many compositions that have been left unfinished by their composers, such as *Mozart's Requiem in D minor* (1). Many people have attempted to complete this and other works, and AI might be able to help musicologists and composers to create effective completions.

Recurrent Neural Networks (RNNs) are a type of machine learning model that may be appropriate for composing music. RNNs are able to generate outputs in the context of previous ones (2,3), which is an important skill in composing music. When composing a section in a piece of music, one must consider the notes that come before it, as it provides context for that section. However, one must also consider what will come after that section as well in order to create a cohesive piece of music. Bidirectional RNNs are helpful for this, as they are capable of considering future outputs when generating an output (2). Long Short-Term Memory (LSTM) networks are a type of Bidirectional RNN which can consider even earlier outputs than a traditional RNN. We used an LSTM, as their enhanced capability to consider earlier outputs made it more likely to create a cohesive piece of music (2).

There are a few characteristics of a piece that are important when analyzing music. One such characteristic is a key signature. Key signatures denote the set of notes that are used in a piece of music (4). For example, a piece in the key of C major will have the notes C, D, E, F, G, A, and B. However, some pieces utilize accidentals, which are notes in a piece that are not in the key signature. A time signature is an indicator of rhythmic structure (5). For example, in a piece with a 4/4 time signature, there are 4 beats in a measure, and a quarter note gets the beat. In a piece in 2/2, there are two beats in a measure, and a half note gets the beat. Polyphony is another important characteristic of music, especially Baroque music. A polyphonic piece has multiple ascending and descending lines that occur simultaneously (6). An example is a piano piece in which a musician plays two different melodies with each hand at the same time. A concept used specifically for digital compositions is a MIDI (Musical Instrument Digital Interface) attack, which is when a series of notes are played over the same interval in a MIDI file (i.e. they are played at the same time).

An important piece of research that helped inform this study is Bachbot (7). Bachbot uses an RNN with LSTM cells, similar to this study. Bachbot is trained on a dataset consisting of only Bach's chorales in MusicXML format. A chorale is a piece of music typically performed by a chorus in church services (8). Congregation members were expected to sing along, so these pieces utilized simple melodies. Instead, composers put more effort in making impressive harmonies. As a result of their simple melodies, chorales tend to consist of a lesser amount of attacks which are held for long periods of time. The creators transposed each of the chorales into the key of C major and quantized the time intervals in the pieces to sixteenth notes. Also, they avoided encoding other complex musical concepts, like motifs and chords. Rather, they only focused on notes, relying on the model to learn and display these more

complex traits after having been trained. However, they did put some focus on harmonization, and harmonization is an important part of composing chorales. In order to measure the success of their algorithm, they created a public website where people could take a musical discrimination test. In this test, participants listened to audio recordings of a piece composed by Bach and a piece generated by Bachbot, and participants guessed which one they believed was composed by Bach.

In order to see if an RNN designed for text generation could create music structurally similar to Bach's, we generated a dataset consisting of compositions by Johann Sebastian Bach, a Baroque composer, to train an open-source RNN designed for text generation. We hypothesized that this text-generation RNN is capable of imitating Bach's compositions. Our results aligned with our hypothesis, as all of the metrics used to analyze the output compositions were within the standard deviations of the metrics for the input compositions. AI has been used to effectively create Baroque music in the style of Bach before, as evidenced by Bachbot. However, Bachbot only uses Bach's chorales as training data, rather than his orchestral/instrumental music (7). In this study, we developed a new dataset using both orchestral/instrumental pieces and chorales from Bach's repertoire to train the RNN. Because this dataset is more comprehensive than that used by Bachbot, we predicted that our algorithm would generate music more representative of Bach's compositions than the music generated by Bachbot. To test this, we compared two rhythm-related metrics used to analyze the outputs of our RNN and the Bachbot output: The $\frac{Attacks}{Spaces}$ ratios and the $\frac{Spaces}{Attacks+Spaces}$ ratios. Since Bachbot was trained only on chorales, we hypothesized that our model's outputs' $\frac{Attacks}{Spaces}$ ratios will be significantly higher than Bachbot's and the $\frac{Spaces}{Attacks+Spaces}$ ratios will be significantly lower. Our results also followed this hypothesis, as the $\frac{Attacks}{Spaces}$ ratios were significantly higher than Bachbot's ($p < 0.001$) and the $\frac{Spaces}{Attacks+Spaces}$ ratios were significantly lower($p<0.001$.)

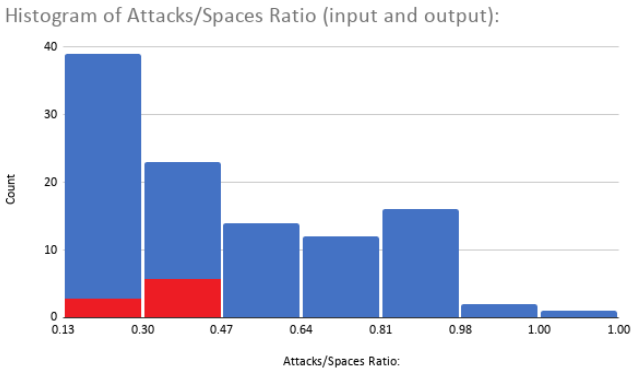

Figure 1: Superimposed histograms of the $\frac{Attacks}{Spaces}$ ratios for the input (blue) and output (red) datasets. Each bin has a size of 0.17. The output set is mostly in the 0.30-0.47 bin, but is also present in the 0.13-0.30 bin, which are the two most populated bins.

RESULTS

Showing the Output's Similarity to Bach's Music

It is customary for similar experiments to use similar metrics for analyzing their algorithm's performance. However, in their study, Bachbot's creators evaluated their model's performance by analyzing user feedback. They created a website, Bachbot.com, and promoted it via social media. Users indicate their age range (under 18, 18-25, 26-45, 46-40, and over 60), and their level of music knowledge (novice, intermediate, advanced, and expert). Then, they take a musical discrimination test with 5 questions. In each question, users listen to two audio recordings (a piece composed by Bach and a piece generated by Bachbot), and they must guess which piece was composed by Bach. This method of measuring the model's effectiveness relies on a participant's subjective perception of the music.

In this study, we analyzed the RNN-generated compositions using four objective metrics. All of these metrics rely on a "cleaned" CSV format (see Methodology section), which represents the music in a more efficient way. The "attacks" refer to a series of notes played simultaneously, and spaces or space characters refer to the time between one attack and the next. One metric was the ratio of accidentals to total notes in the piece, $\frac{Accidentals}{Total\ Notes}$. The second metric was the average number of notes per attack in a piece, $\frac{Notes}{Attacks}$. This was done to quantify polyphony, a defining trait of Bach's music. A large ratio of notes per attack throughout a piece indicates that each instrument in the piece has a substantial part, which is what occurs in polyphonic pieces. The ratio of space characters to total attacks plus space characters ($\frac{Spaces}{Attacks+Spaces}$) was also used, as well as the ratio of attacks to space characters ($\frac{Attacks}{Spaces}$). Both of these metrics are somewhat similar, as they consider the rhythmic structure and give a sense of how "dense" the music is. For example, a chorale represented in the cleaned CSV format will have fewer attacks and more space characters, since notes are held for longer periods of time. Compared to a piece that is not a chorale, these metrics will look different.

We calculated these metrics for each piece in the input dataset, and the averages and medians for each metric as well(**Table 1**). We also calculated the standard deviation for each metric (**Table 1**). The values and ratios for these metrics for each of the eight output pieces generated by our RNN are summarized in **Table 2** and **Figures 1-4** . The $\frac{Attacks}{Spaces}$ ratio, the $\frac{Spaces}{Attacks+Spaces}$ ratio, and the $\frac{Notes}{Attacks}$ ratio were within the standard deviation for the input dataset. **Figure 1** demonstrates how the $\frac{Attacks}{Spaces}$ ratios for the outputs fall within the distribution for the inputs, and **Figures 2-3** do the same for the $\frac{Spaces}{Attacks+Spaces}$ ratio, and the $\frac{Notes}{Attacks}$ ratio, respectively. The same was true for the $\frac{Accidentals}{Total\ Notes}$ ratio, but using each data point's z-score, we found one outlier ($z > 3$) in the input dataset that increased the standard deviation (**Figure 5**). Without that outlier, the standard deviation was reduced from 0.0830 to 0.0636 (**Table 1**). Despite this, all of the accidental ratios for the outputs were within the standard deviation for the input set. **Figure 4** demonstrates how the $\frac{Accidentals}{Total\ Notes}$ ratios for the outputs fall within the distribution for the inputs when the outlier is removed. Thus, it can be concluded that this model produced music

that was structurally similar to Bach's.

Table 1.

Ratio	Median	Average	Standard Deviation	Standard Deviation Excluding Outlier
Accidentals/Total notes	0.0579	0.0746	0.0830	0.0636
Notes/Attack	2.3727	2.6588	1.1915	(N/A)
Spaces/(Attacks+Spaces)	0.7375	0.6983	0.1130	(N/A)
Attacks/Spaces	0.3559	0.4732	0.2584	(N/A)

Table 1: The averages, medians, and standard deviations for each of the four metrics for the input dataset.

Table 2.

Piece Number	Accidentals/ Total Notes	Notes/Attacks	Spaces/(Attacks+Spaces)	Attacks/Spaces
1	0.0682	2.1290	0.7646	0.3079
2	0.0611	2.2200	0.7693	0.2998
3	0.0314	2.6267	0.7926	0.2617
4	0.0482	2.2937	0.7372	0.3564
5	0.0524	2.8612	0.7880	0.269
6	0.0534	2.6822	0.7457	0.3410
7	0.0444	3.1096	0.7467	0.3392
8	0.0319	3.2910	0.7654	0.3065

Table 2: The total values and ratios for accidentals, notes, attacks, and spaces for each of the eight pieces generated by the RNN.

Comparing the Outputs to Bachbot's Outputs

Since we've shown that this experiment's output is structurally similar to Bach's repertoire, we compared Bachbot's ratios for each of these metrics to this experiment's output ratios to determine whether our method produces compositions that more effectively represent Bach's repertoire than those produced by Bachbot . The metrics that are related to rhythmic structure ($\frac{Attacks}{Spaces}$ ratio and the $\frac{Spaces}{Attacks+Spaces}$ ratio) are important when discussing how the results of this algorithm are more comprehensive than Bachbot's. This is because Bachbot only trained on chorales, whereas this RNN trained on both chorales and instrumental pieces. Bach's chorales have differing rhythmic structure than normal instrumental pieces (8), so Bachbot's output would likely reflect that. This makes the rhythm-related metrics necessary for comparing these algorithms' outputs. Since the chorales used in Bachbot's training tend to consist of a lesser amount of attacks, we hypothesized that Bachbot's output would tend to have a lower $\frac{Attacks}{Spaces}$ ratio and a higher $\frac{Spaces}{Attacks+Spaces}$ ratio compared to the output pieces of our RNN. In order to verify this, the five original compositions made by Bachbot and this experiment's outputs were compared using the $\frac{Attacks}{Spaces}$ and $\frac{Spaces}{Attacks+Spaces}$ ratios for all the outputs. In accordance with our hypothesis, the $\frac{Attacks}{Spaces}$ ratios from this experiment are significantly higher than Bachbot's ($p < 0.001$), and the $\frac{Spaces}{Attacks+Spaces}$ ratios are significantly lower than Bachbot's ($p < 0.001$). This shows that because of the strict use of chorales in Bachbot's dataset, their outputs are more similar to chorales, whereas this experiment's outputs are more representative of all of Bach's repertoire.

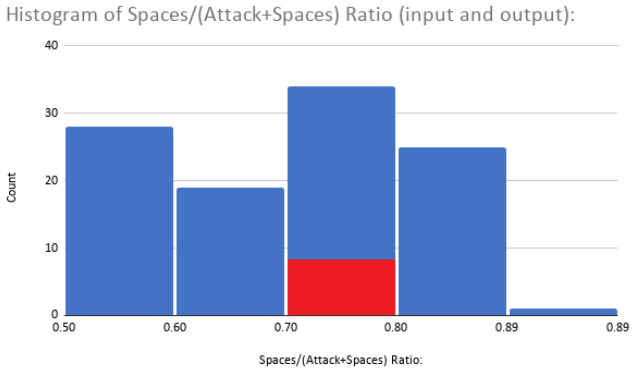

Figure 2: A histogram of the $\frac{Spaces}{Attacks+Spaces}$ ratios calculated for the input and output datasets. The datasets are bucketed into five bins, each with a size of 0.10. The output set histogram (red) is superimposed on the input set histogram (blue). All of the output set is present in the 0.70-0.80 bucket, which is the largest bucket.

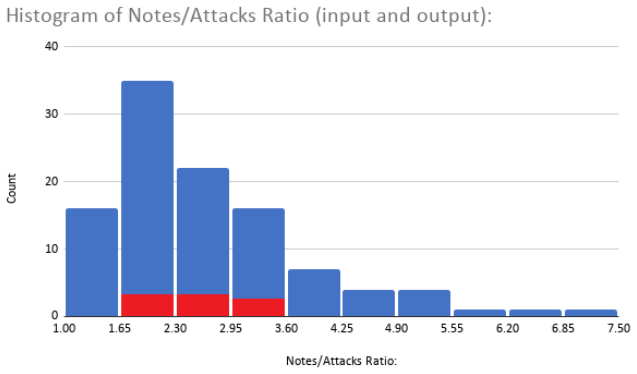

Figure 3: A histogram of the $\frac{Notes}{Attacks}$ ratios calculated for all the generated pieces. The datasets are bucketed into ten bins, with a size of 0.65. The output set histogram (red) is superimposed on the input set histogram (blue). The output set is present in the three largest buckets - the 1.65-2.30 bucket, the 2.30-2.95 bucket, and the 2.95-3.60 bucket.

DISCUSSION

Effective algorithms to generate music similar to Bach's music exist, but they only cover specific aspects of his music. An example of this is Bachbot (7), as it only uses Bach's chorales as training data. However, we have shown that an RNN designed for text generation given a comprehensive dataset consisting of all types of music from Bach's repertoire

can produce music more representative of Bach's work. Four metrics, the $\frac{Accidentals}{Total\ Notes}$ ratio, the $\frac{Attacks}{Spaces}$ ratio, the $\frac{Spaces}{Attacks+Spaces}$ ratio, and $\frac{Notes}{Attacks}$ ratio, were calculated for each piece in the input dataset (**Table 1**) as well as for each of the RNN-generated pieces (**Table 2, Figures 1-4**).

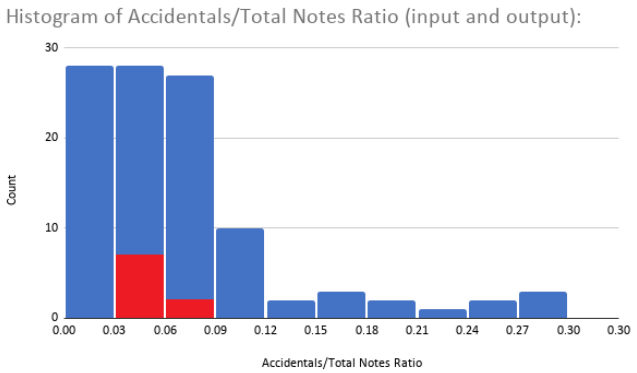

Figure 4: A histogram of the $\frac{Accidentals}{Total\ Notes}$ ratios calculated for the input and output datasets. In this data, the outlier in the input set is not present. The datasets are bucketed into ten bins, with a size of 0.03. The output set histogram (red) is superimposed on the input set histogram (blue). Most of the output dataset is present in the 0.03-0.06 bucket, but is also present in the 0.06-0.09 bucket, which are both one of the largest buckets.

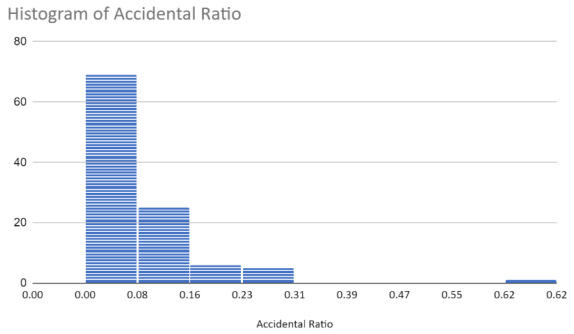

Figure 5: A histogram of the $\frac{Accidentals}{Total\ Notes}$ ratios for the input datasets. All but one of the data points is within the range of 0.00 to 0.31. Thus, that piece is an outlier in regards to this metric.

The results here showed that the outputs from this experiment are more representative of the breadth of Bach's repertoire compared to Bachbot's outputs. We compared Bachbot's outputs against our model's outputs in order to gauge how similar Bachbot's outputs are to all of Bach's repertoire. Bachbot's Attacks/Spaces ratios were significantly lower than this experiment's outputs ($p < 0.001$), and their $\frac{Spaces}{Attacks+Spaces}$ ratios were significantly higher ($p < 0.001$), showing that Bachbot's outputs are not as similar to Bach's repertoire compared to this study's model. This is shown in **Figure 6**, where Bachbot's $\frac{Attacks}{Spaces}$ ratios were all grouped lower than this project's outputs, and where their $\frac{Spaces}{Attacks+Spaces}$ ratios were all grouped higher.

There were several limitations in this study. There was an outlier present in the input dataset with the $\frac{Accidentals}{Total\ Notes}$ ratios, and although the metrics weren't significantly impacted with or without the outlier, it might have affected the music generated by our RNN. This may be an internal source of error, as the piece that was an outlier might not have been in the key of C, G, or D major. It also is possible that it could be an external source of error instead, as the MIDI file of the piece could have been incorrect since each of the MIDI files we analyzed was crowdsourced (9). Additionally, this projected was limited by the computational resources available, which resulted in a less-than ideal, but still adequate amount of output pieces. This resulted in the output distributions for the histograms (**Figures 1-4**) not obviously matching the input ones.

Further research could be done using more computing

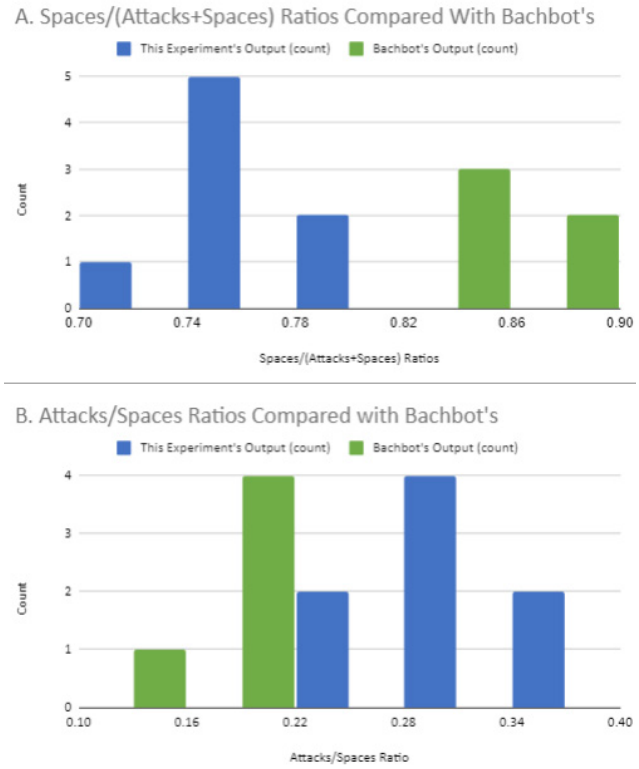

Figure 6: Comparison of the output of this study's RNN with Bachbot. a) In the $\frac{Spaces}{Attacks+Spaces}$ ratio histogram, Bachbot's ratios are consistently in the higher bins compared to this experiment's outputs. b) In the $\frac{Attacks}{Spaces}$ ratio histogram, Bachbot's ratios are consistently in the lower bins compared to this experiment's outputs.

resources and a different RNN architecture, perhaps specifically designed for this kind of project. Different data collection methods might be used as well, such as using an algorithm that converts sheet music documents into text instead of using MIDI files . Such a data collection method would make it easier to find more training data, which would improve the RNN's performance. Also, in order to increase the size of the dataset, there could be transposed versions of a piece, so that there could be a version of the piece for

every key allowed in the dataset. Utilizing a different means of data collection might allow more pieces of music to be found, and more data will generally help a neural network train more effectively.

While this project faced many challenges in terms of acquiring the dataset and computational resources, it resulted in an algorithm that outperformed a more prominent one. Further work in this topic could be extremely impactful for many, from composers trying to reconstruct lost symphonies, to those who simply enjoy music.

MATERIALS AND METHODS

Preparation of Musical Data

Before any experimentation was done, the training data had to be formatted in a way that would make it as comprehensible to the RNN as possible. To collect the data, all of Bach's music that was in a MIDI format and freely available was collected using the International Music Score Library Project (9). Music was chosen in the keys of C major, G major, and D major. These keys were chosen because Bach composed in these keys more than any other (10), and because of their close relation to one another (C has no notes that are sharp, G has one sharp note, and D has two sharp notes). All pieces selected were either in the time signatures of 4/4, as it is the most common time signature in music. Pieces in 2/2 were used as well, as the time intervals between notes are identical to 4/4. If the RNN trained on pieces both in 4 and in other meters, the time intervals would be different, causing the RNN to create irregular, inconsistent rhythms throughout the piece. Solo pieces were excluded, as achieving Baroque-style polyphony is impossible with only one instrument. However, organ/harpsichord pieces still achieve polyphony due to how different parts can be played with both hands simultaneously, so these are still in the dataset. Ultimately, 107 suitable pieces were found.

A set of programs written by Fourmilab that convert MIDI files to CSV files and vice versa were used to convert MIDI files into text files in CSV format (11). Once the files were converted into text files, a we made a Java program to strip each text file down to the bare minimum the RNN needed to compose music: the pitch and timing of each note. A MIDI composition consists of tracks, which are the different "parts" of a piece, such as the different instrumental parts in an orchestral piece. Each track was stripped one at a time. When stripping a track, the important parts for composition are the tick number, which indicates the timing of the note in MIDI ticks, and the pitch, which is a value in the range of 0 to 128 (12). A "map" was created to correlate each pitch value with a single text character that was used for the RNN array. The Note_On/Off_c is also important, as it indicates when the notes turn on or off. Velocity is only important when v=0, as it causes the note to act like it has a "Note_Off_c." All of the lines with "Note_Offs" and a velocity of 0 were deleted. After that, the track number, the channel number, the Note_On_c information, and the rest of the velocity information could be deleted. There were also other metadata that were necessary for playback, but unimportant for composition that were deleted. The "number of MIDI ticks per quarter note" from the file was divided by 8 to represent the number of MIDI ticks per 32nd note in the piece, and then all of the tick values for each note were divided by that number to yield the amount of 32nd note intervals between the next note. An array that was the size of the new tick number in the last note of the track was generated, and the single-character pitch value representations were added to the array at positions corresponding to their new tick number. This process was repeated for each track in the MIDI composition. If multiple pitch values were in the same index, they were concatenated together which shows that these values were being played at the same time, otherwise known as an "attack."

The cleaned composition was then written into a new text file. The entire array was printed, and the indices with null values were represented as spaces. The number of spaces between two strings of characters indicated the number of 32nd note intervals between the playing of those notes. After the RNN trained, the output text was converted back into MIDI format for listening purposes to ensure that the music was still represented correctly after it had been converted. When converting a cleaned MIDI file back to a CSV text file, a similar procedure was used as that described above. However, when multiplying the 32nd note intervals by the tick value per quarter note, this tick value can be any number. Also, one can take liberties when adding the metadata back to the original file, which is in the header of the CSV file. The metadata values will not affect the composition – it only exists so that a MIDI player can compile the music. As a result, the metadata values were chosen from another MIDI file that was successfully played by a MIDI player. However, the number of MIDI ticks per quarter note must be kept the same as the number that was multiplied with the tick values. After this, the text file was imported into Microsoft Excel, saved as a CSV, and re-converted into a MIDI file using the Fourmilab programs (11).

RNN Model and Generation of Music

The RNN used in this work was provided by Max Woolf, a data scientist at Buzzfeed in San Francisco (13). The RNN model comprised 3 layers, each with 128 bidirectional LSTM cells, as LSTM is better than regular RNNs at retaining information from previous layers (3). All of the cells used the sigmoid activation function (13), which was represented as $s(z) = \frac{1}{1+e^{-z}}$. In this function, z is the input vector for the node, which is the output vector of a previous node, or the original input (14). The loss function used was Categorical Cross-Entropy loss (13), which is represented as $\sum_{c=1}^{M} Y_{o,c} \ log(P_{o,c})$. M is the number of classes that a model can categorize outputs into. Y is a binary indicator, indicating whether a class, c, is the correct class, based on an observation, o. P is the predicted probability that o is actually of class c (15).

For each of eight trials, the RNN was trained on the

dataset described above for 20 epochs. 90% of the data was used for the model to train on while the other 10% was used for validation. After this, the trial with the smallest amount of loss was chosen. The weights from that trial were exported, and eight samples of music were generated from them.

The five publicly available original compositions made by Bachbot were in .WAV format (16). In order to compare them to this experiment's outputs, the files were converted from .WAV into MIDI using Melodyne (17). Afterwards, the MIDI files were converted to the cleaned MIDI format using the same method as the input dataset for this study.

ACKNOWLEDGEMENTS

I would like to sincerely thank Ms. Cristi Lebron at The Walker School for supervising my work. I would also like to thank The Walker School for providing me with the opportunity to pursue this project and the skills necessary to do so.

Received: May 25, 2020
Acccepted: February 9, 2021
Published: May 5, 2021

REFERENCES

1.Schwarm, Betsy. "Requiem in D Minor, K 626." *Encyclopædia Britannica*, Encyclopædia Britannica, Inc., 30 Mar. 2016, www.britannica.com/topic/Requiem-in-D-Minor.
2.Sutskever, Ilya, et al. "Generating Text with Recurrent Neural Networks." University of Toronto, 2011. Accessed 22 Nov. 2019.
3.Mahendran Venkatachalam. "Recurrent Neural Networks." *Medium*, Towards Data Science, Mar. 2019,towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce. Accessed 22 Nov. 2019.
4.Rios, Nicki. "Key Signatures." *Key Signatures | Music Theory Tutorials*,musictheoryfundamentals.com/MusicTheory/keySignatures.php.
5. Aichele, Michele, et al. "Understanding Time Signatures and Meters: A Musical Guide." *Liberty Park Music*, 22 Sept. 2020, www.libertyparkmusic.com/musical-time-signatures/.
6. "Polyphony." Encyclopædia Britannica, *Encyclopædia Britannica*, Inc., www.britannica.com/art/polyphony-music.
7. Liang, Feynman T. et al. "Automatic Stylistic Composition of Bach Chorales with Deep LSTM." *18th International Society for Music Information Retrieval Conference*, 2017.
8.The Editors of Encyclopaedia Britannica. "Chorale." Encyclopædia Britannica, *Encyclopædia Britannica*, Inc., 20 Nov. 2011, www.britannica.com/art/chorale.
9. IMSLP: Free Sheet Music PDF Download." *Imslp.Org*, 2019, imslp.org/wiki/Main_Page.
10. Marzban, Ethan Paik, and Caren Marzban. "On the Usage of Musical Keys: A Descriptive Statistical Perspective." *The Journal of Experimental Secondary Science*.
11. Walker, John. "MIDICSV: Convert MIDI File to and from CSV." *Fourmilab.Ch*, 2019, www.fourmilab.ch/webtools/MIDIcsv/. Accessed 22 Nov. 2019.
12. "Note Pitch Values." Musescore.Org, 2019, musescore.org/en/plugin-development/note-pitch-values.
13. Woolf, Max. "Minimaxir - Overview." GitHub, *GitHub,* 2012, github.com/minimaxir. Accessed 22 Nov. 2019.
14. Fortuner, Brendan, and Prashant Piprotar. "Activation Functions." *Activation Functions-ML Glossary Documentation*, 2017, ml- cheatsheet.readthedocs.io/en/latest/activation_functions.html#sigmoid.
15. Fortuner, Brendan, and Prashat Piprotar. "Loss Functions." *Loss Functions - ML Glossary Documentation*, 2017, ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html.
16. "BachBot." *SoundCloud*, 2016, soundcloud.com/Bachbot
17. "What Is Melodyne?" *Celemony*, www.celemony.com/en/melodyne/what-is-melodyne.