Song Generator

Andrea Keiper | keiper.a@northeastern.edu
Kaydence Lin | lin.kay@northeastern.edu
Jeremiah Payeur | payeru.j@northeaster.edu
Samantha Sobhian | sobhian.s@northeastern.edu

Northeastern University, Boston, MA, USA

Abstract

For this final project, a song lyric generator was created that would take in songs from a single artist, and produce a new song that stayed true to the artist's style. Analyzing different areas of the music of four artists, the goal was to use weighted n-grams and sentiment analysis to generate a new song for each artist, and then use cosine similarity to visualize comparisons.

Introduction

While creating a song lyric generator, there were a few goals in mind. The goal was to create a song that exuded the authenticity of the artist being mimicked while maintaining a certain degree of uniqueness when compared to the other songs in the database. A combination of n-grams, probability, sentiment analysis, and rhyme scheme development was used in order to produce such songs. With this song creation, the group sought to discover: how far could natural language processing go? If one compiles a database of songs for a single artist and generates a song based on this collective database, what is the similarity between each one of these songs and the generated song? Knowing this, how does this generated song compare in similarity to other artists' songs, and songs generated by the algorithm for those other artists?

For the four different artists, songs from different areas of their music were pulled. What kind of song might be produced when given all the songs from just one Taylor Swift album? What might be the outcome of looking at Beyonce's biggest hits only? What song would result from some of Steve Lacy's most popular songs and some of his least-known ones? And Lauryn Hill's solo music vs her career with the Fugees? The motivation behind this includes both the group's interest in music and NLP. Can one use their knowledge of natural language processing to mimic the songs they enjoy the most? It would be interesting to dissect the lyrics and sentiment of these generated songs and analyze whether they have the same, or even a similar effect, that the artists' true songs have on the listener.

Data Sources and Methods

Data sources for this project include the lyrics of the artists' songs that were chosen[4]. A collection of songs were specifically chosen from Taylor Swift, Beyoncé, Steve Lacy, and Lauryn Hill. Data was acquired by Google searching the lyrics for each chosen song and copying them into separate txt files, which were read into the code.

The first step was data munging; after reading in the data, the lyrics are made usable by splitting each line by word and removing punctuation and other aspects of the lyrics that may interfere with our algorithm. Using a dictionary of songs for each artist, a list of n-grams (tri-grams), start words, and end words were created for each artist. Start words are held in a dictionary, with the start word as the key and the probability of it being selected as the value. In order to make the lyric generator more accurate, there is a counter function that creates a counter

of the prevalence of each n-gram in the actual songs, as well as a function that applies a probability to the random selection of each n-gram in song creation based on the n-gram counter. This creates a dictionary of weighted tri-grams. For each song in our database, the last two words of every line are appended to a list as a tuple, creating a list of end-grams.

When the song lyrics are generated, the start words and the tri-grams are associated with their weighted values so that in random selection, the more prevalent start words and tri-grams have a higher probability of being chosen in the generated songs, while still offering a sense of randomness to the lyric selection. As for the end words, the list of end-grams are the only words that a line may end with to be considered complete.

In terms of sentiment, a list of sentiments over time was created for each artist. Each song in each artist's database was split into parts based on the length of the song. The sentiment for each part of the split-up song is stored in a list, and the sentiment for each part of the song was averaged across each artist. This generated four lists, an average sentiment over time for each artist. When the song lyrics are generated, the sentiment of the line being generated must fall in line with that artist's average sentiment in that section of their true songs. This creates a natural flow of sentiment throughout the generated song that should match the artist's style.

In order to generate songs with a rhyme scheme, a rhyme scheme function was implemented. The user is able to input a rhyme scheme of their choice as a string of letters, such as 'ABAB.' One function returns a tuple containing the length of the rhyme scheme and the order of the rhyme scheme. Order sorts the rhyme scheme so that the last words in rhyming lines

are associated. For example, in an ‘ABAB’ rhyme scheme, lines 1 and 3 should be associated with A, and lines 2 and 4 should be associated with B.

To generate the song, a line-generating function was created which would take a random start word and add words to it based on the weighted tri-grams, and stop adding words once the last two words are in the list of end-grams, and it had reached the minimum length for words per line OR it had hit the maximum length for words per line. The song would only accept the line if the line fell within the specified sentiment range for that section of the song. For each line in the generated song, the line number determines which part of the rhyme scheme the end word should follow. It specifies that if it is the first word of the rhyme scheme, the end word can be anything. However, if it is a repeated instance of a specific rhyme (i.e. the second ‘A’ in ‘ABAB’), Jame Wenzel’s GitHub library, phyme[1], was used to decipher whether or not the associated lines are true rhymes. The word will only be accepted if it is an acceptable rhyme OR if it is over the 1000th line generated (suggesting that it may be very difficult to find a true rhyme to the prior word).

In order to create visuals to analyze the data within the project, cosine similarity graphs were used. This involves a filtering function to remove common words[9], vectorize, magnitude, dot product, and cosine similarity functions in order to find the cosine similarity value between all songs in each artist’s dictionary as well as the program’s generated song.

Analysis

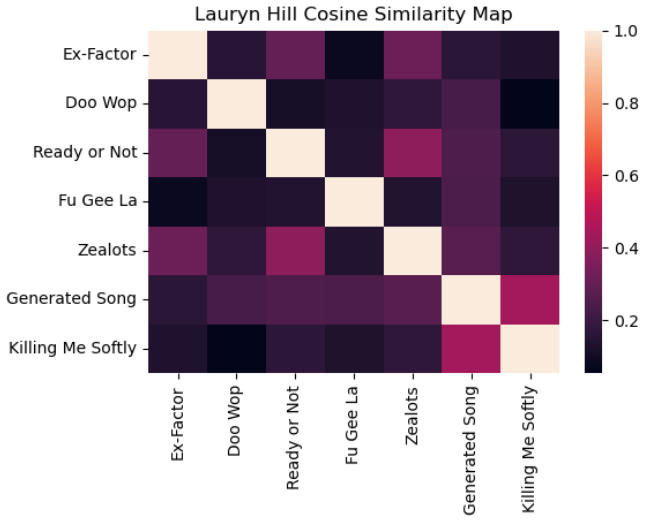


Figure 1. A cosine similarity graph depicting the closeness in lyrics of all songs in the Lauryn Hill database, which includes songs from her solo career and her work with Fugees, as well as the song generated by the algorithm. The second to last row/column shows the generated song, with a range of similarity scores similar to that of the other songs to each other.

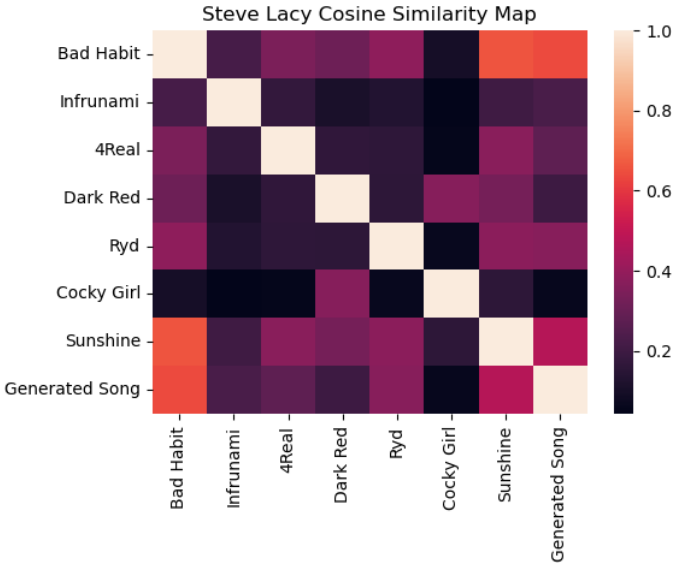


Figure 2. A cosine similarity graph depicting the closeness in lyrics of all songs in the Steve Lacy database, as well as the song generated by the algorithm. Note that Bad Habit, Dark Red, Ryd, and Sunshine are popular songs, while Infrunami, 4Real, and Cocky Girl are underground.

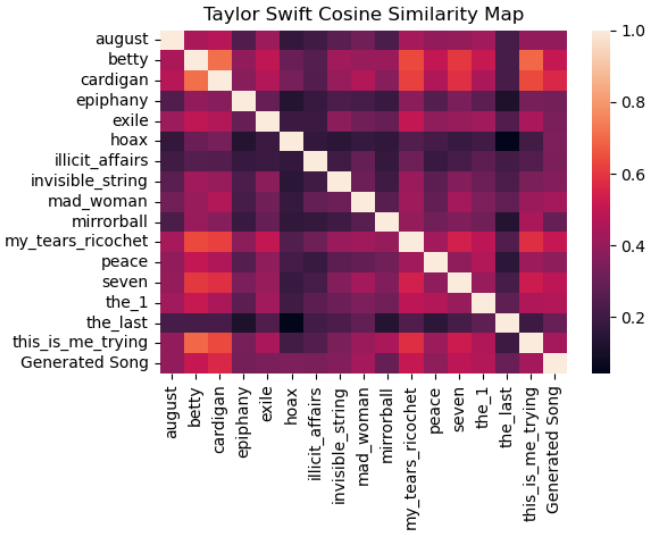


Figure 3. A cosine similarity graph depicting the closeness in lyrics of all songs in the Taylor Swift database, all songs chosen from the album *Folklore*, as well as the song generated by the algorithm. The generated song displays similarity scores similar to if not higher than that of the true songs compared to one another.

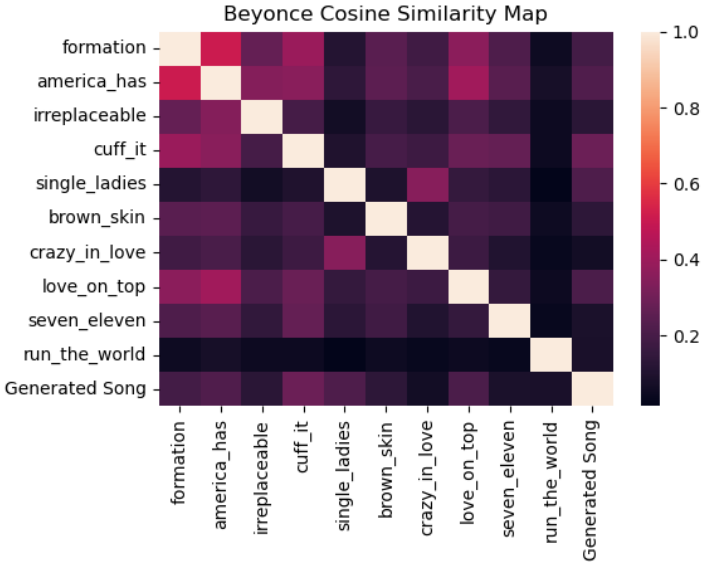


Figure 4. A cosine similarity graph depicting the closeness in lyrics of all songs in the Beyonce database, her most streamed songs, as well as the song generated by the algorithm.

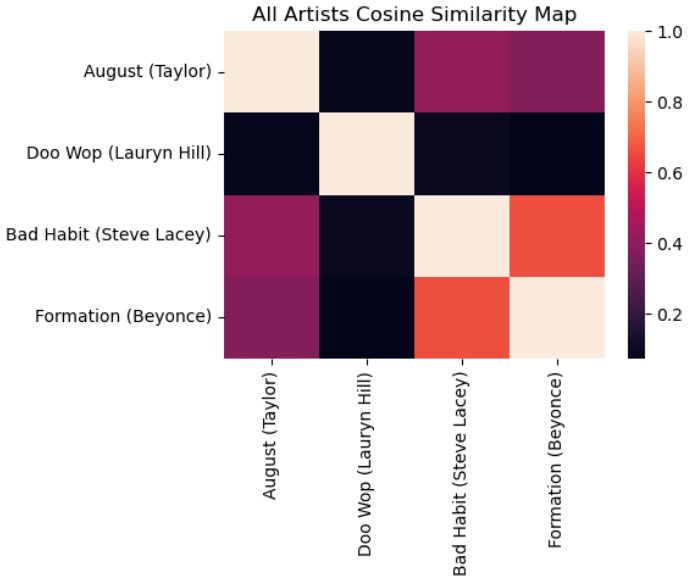


Figure 5. A cosine similarity graph depicting the closeness in lyrics of one song from each of the artists' databases. This graph shows Lauryn Hill to be the most different from all artists, with Steve Lacy and Beyonce being the closest. Taylor Swift's lyrics are slightly closer to Steve Lacy's than Beyonce's.

Taylor Swift	Beyoncé	Lauryn Hill	Steve Lacy
What about that But if I came back Roaring 20s tossing pennies in the summer They show their truth one single glance I was an old cardigan Ive never been a natural Who knows if I never showed up at your yacht Tell yourself you can always stop And if you never gave a warning sign So who am I offending now But we were still changin for the longest time You showed me colors She stole his dog and dyed it key lime green Betty one time I was young All along there was some Youll be flushed when you return Chase two girls lose the one Every time you call when youre back at school I passed your house Dont want no other In the weeds Leave the perfume on the High Line Cause you werent mine to lose Just thinking of you all summer long	Bring the beat that my heart skips when Im with you Got me tighter in my bag double Gs on my hips Get whats mine Im a seasoned professional And keep talking that mess thats fine Ooh wee BB freaky deaky think me see she pink bikini All the single ladies Bet you youll go far night Melanin too dark to throw her shade Young Hov yall know when the cameras close in Man I know Its ya boy Young Dont you ever had fun like this Have you ever for a second Whole slab I kill for All the single ladies Bet you youll see far Get whats mine Im a seasoned professional Young Hov yall know when the cameras close in Man I know Its ya boy Young Dont you ever had fun like this Have you ever for a second To every single curve your body natural What it really feels like to miss me Finally you put me first To every single curve	Well show you how the refugees bring Killing me softly with his fingers Plus when you really are a gem Its against the laws of physics And when I needed you Gonna I beg that you cant hide Killing me softly with his song Why wont you live for me Im bringin on Haitian Sicilians Yo remember back on the side of Babylon trying to front like hes down with That can only get down with No matter how I think not Ill send a letter to my eyes As she cry mi amor the phantom dies in the bag banker looked like a drag Yo remember back on the first episode A born again hooligan only to be Why wont you live for me Im bringin on Haitian Sicilians Yo remember back on the side of Babylon trying to front like hes down with That can only get down with	When she said park my car down the backstreet Speedin’ Maybe ride on this empty road Why I feel this way I don ’ t play with me I know she was hungry does she want any food Vision pretty baby as I ride on this empty road Maybe ride on this empty road Please don ’ t wan na leave me Maybe ride on this empty road You always knew the way to wow me Linger for far too long Speedin’ It’s It’s ‘Cause Vision pretty baby as I ride on this empty road Maybe ride on this empty road You always knew the way to wow me Linger for far too long Speedin’ Funny you come back around I think of her so much it drives me crazy I didn ’ t wan na leave me I know I ’ ll show you right You think I ’ m

She stole his dog and dyed it key lime green Betty one time I was young All along there was some Youll be flushed when you return Chase two girls lose the one Every time you call when youre back at school Its obvious that wanting me dead Under someones bed With you I fall down down Im dead to you Spinning in my town	your body natural Got me hoping youll page me right now To thinking youre irreplaceable And like a Green Beret Who will buy it for themselves and get more money later	Toss me high only puff lye with my crew Damn another dead pigeon Killing me softly with his song Said youd be there for me Let it sit inside your head like a drag You let go Violence aint necessary unless you provoke me Bust rap toons on flat spoons take no shorts like poon poons	always gon na be yeah I think of her so much it drives me crazy It's Why I feel this way I don ' t replace it

Table 1. Sample lyrics of songs generated from the program for each of the artists chosen to be represented.

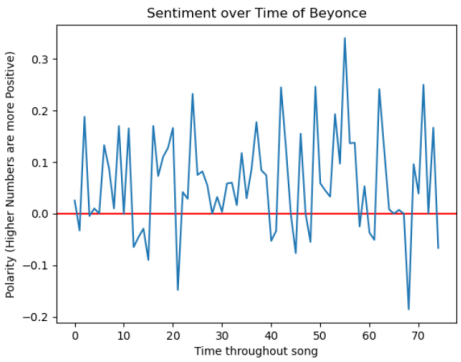


Figure 6a. (Top left) Average sentiment over time of Beyoncé's songs in the database. Note that the sentiment changes more frequently.

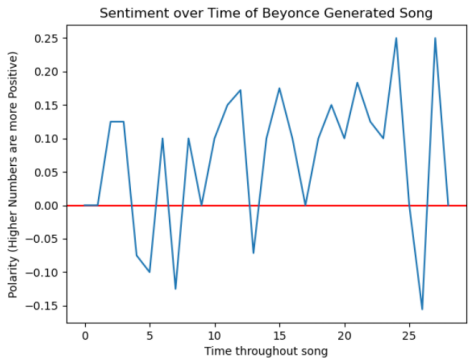


Figure 6b. (Top right) Sentiment over time of the program generated Beyoncé song.

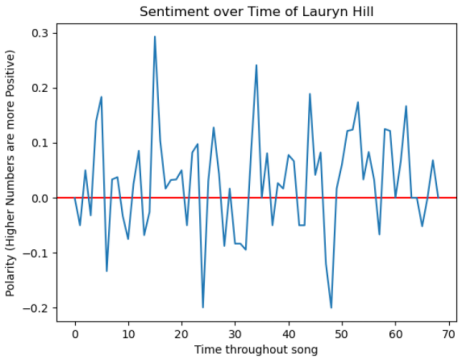


Figure 7a. (Bottom left) Average sentiment over time of Lauryn Hill's songs in the database. Note that the sentiment changes more frequently.

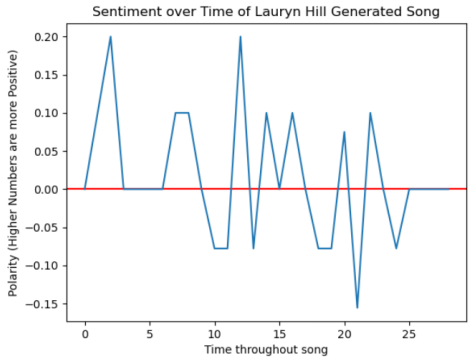
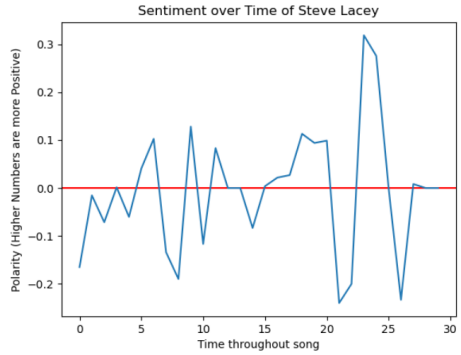


Figure 7b. (Bottom right) Sentiment over time of the program generated Lauryn Hill song.



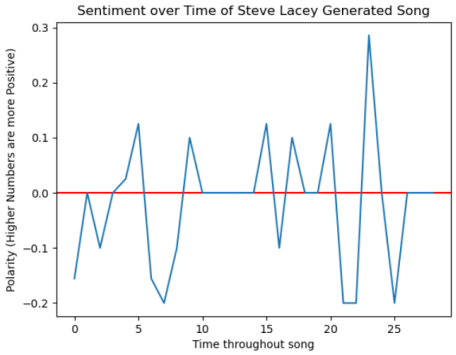
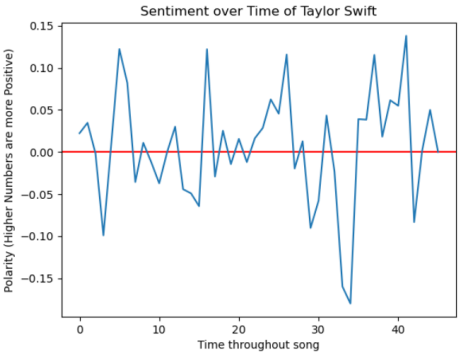


Figure 8a. (Top left) Average sentiment over time of Steve Lacy's songs in the database.

Figure 8b. (Top right) Sentiment over time of the program generated Steve Lacy song.



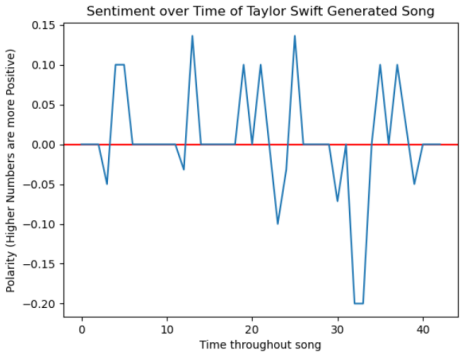


Figure 9a. (Bottom left) Average sentiment over time of Taylor Swift's songs in the database.

Figure 9b. (Bottom right) Sentiment over time of the program generated Taylor Swift song.

Conclusions

The program successfully generated a new song from each artist, using lyrics they used in previous songs (Table 1). A rhyme scheme was implemented as a parameter; this modularity allowed for an AABCBC rhyme scheme for the generated Taylor Swift song, and an ABAB scheme for the other three artists in order to give the songs more rhythm and flow. It can be adjusted to any desired rhyme scheme. The generated songs are considerably similar to the songs of each respective artist (Figure 1, 2, 3, 4). On top of generating songs to match these artists' styles, comparisons were made between their own music, as well as the generated songs.

To analyze Lauryn Hill, a collection of songs from her solo career were compared to songs she created with Fugees. ‘Ex-Factor’ and ‘Doo-Wop’ are songs from *The Miseducation of Lauryn Hill*, whereas ‘Ready or Not,’ ‘Fuu-Gee-La,’ ‘Zealots,’ and ‘Killing Me Softly With His Song’ are songs from *The Score* by Fugees. When comparing the similarity (Figure 1), it’s interesting to see that there are not necessarily many similarities between the songs she made solo versus with the Fugees. Most of her music is generally different, but similarities can be witnessed between “Ready or Not” and “Zealots,” two songs with the Fugees. The generated song is closest to “Killing Me Softly With His Song.”

To create a new song for Steve Lacy, some of his biggest hits, “Bad Habit,” “Dark Red,” “Sunshine,” and “Ryd” were analyzed next to some of his least known songs such as “4Real,” “Infrunami,” and “Cocky Girl.” The resulting song pulled from all of these, but taking a look at the cosine similarity map for this artist (Figure 2), there are interesting comparisons to be made. The generated song is closest to Lacy’s big hit, “Bad Habit,” which reached number one in the charts in 2022. It is the most different from “Cocky Girl,” a relatively unknown song, which is also the most different from “Bad Habit,” “Infrunami,” and “Ryd,” which is an interesting mix of his popular and unpopular music.

To analyze Taylor Swift's lyrics, all the songs from just one album (*Folklore*) were pulled. Keeping in mind the scale of the map, it’s interesting to see how different her songs are from one another despite being from the same “era.” (Figure 3). Her songs “Hoax,” “Illicit Affairs,” and “The Last” were the most different from the rest of the music on this album.

“Betty” has the highest number of similarities to other songs on the album. The generated song is most similar to “Cardigan.”

In comparing Beyoncés most popular songs (those with the most streams), a great deal of very high similarity scores can be witnessed throughout the graph (Figure 4). This is to be expected when comparing her most successful songs, as there is often a theme across those well-liked pieces. While every one of these songs was extremely successful, they all seem to be very different from each other, which might offer insight into how diversity instead of repetition leads to success; there’s something for everyone in Beyonce’s music. “Run the World” is extremely different from all her other hits. “America Has A Problem” and “Formation” are the closest songs to each other, and they are also closest to the generated song.

Across all four artists, generated songs appear to have roughly the highest average cosine similarity value when compared to all other songs. In other words, it is the song most like *all* the other songs. This is to be expected, as the lyrics chosen for this song came from a database of the combination of all the other songs. The value confirms that even after creating some incoherent lines, the cosine similarity function was still able to determine a similarity between the generated song and those in the database.

In Figure 5, one random song was chosen from each artist for comparison. In this cosine similarity map, Lauryn Hill’s music is completely different from all other artists. Steve Lacy and Beyonce are the most similar artists. This comparison might offer insight into how writing styles and lyricism has changed from the 90’s to the 2010’s. But, it is important to note that because this

map only compares one song from each artist, it is subject to change based on exactly what lyrics are being compared. And while all of our analyses only account for lyrics and not also the musicality of the song, chords, rhythms, and singing styles, lyrical comparison is also valuable since it is an important composition of music that is enjoyed by listeners.

Referring to figures 6a-9b, one can deduce that the sentiment over time for each of the program generated songs closely mimics that of the artists' averages for their songs in the database. The goal in adding a changing sentiment over time to the generated song was to be able to create a natural flow of attitude that closely follows the artists' real style. The figures above aid in viewing this similar flux in emotion (polarity) that was artificially generated throughout the songs.

Finally, reflecting on this project, there is much to consider surrounding the ethics of mimicking an artist's style. Using algorithms could be a way for one to "get" new music from deceased musicians. By analyzing their typical lyricisms, rhyme schemes, and melodic patterns, new technology would be able to produce songs "written by" artists who have passed. But how ethical is this; is a legacy best left untouched? The artist would have no way of advocating for themselves or signing off on lyrics being put under their name. And arguably more importantly, the same can be done for living artists and songwriters as well. When their unique writing talents are easily mimicked by such an algorithm, incomes and entire livelihoods are threatened. Or, could this be advantageous to these individuals instead as a way to become inspired during, perhaps, a period of writer's block? To see different ways

for their lyrics and rhyme schemes to be put together? It is more likely that there is more damage done when an artist is stripped of their talents and anyone in the world is able to produce their works. And with the rapid expansion and improvements in technology and artificial intelligence, there might be a need for protective laws in the future.

There have been debates about this in the music industry, specifically music awards. The Grammy's recently created new rules about the use of AI and outlined how the use of it may or may not receive nominations. To summarize their new rules, songs that may be nominated can have and use elements or components of AI, but a human must have meaningfully contributed to the creation of the song[4].

To conclude, what the group is most proud of in creating this project should be highlighted: the complexity of the song generation itself. Implementing a rhyme scheme, weighted n-grams and start words, multi-word 'end-grams,' and a generated sentiment over time that matches that of the artists' averages were all challenges that the group was eager to solve in the creation of their program.

Limitations and Future Work

Many obstacles and limitations were faced throughout the production of this project. The process of making trigrams was made decently easy through the use of the text blob library, but a probability aspect also needed to be added to this in order for there to be a complete randomization based on associated probabilities or n-grams. This implementation went well, but the first obstacle came about as there was a struggle with coherency from line

to line of each generated song because of the randomization of the start of each line. To try to get the lines to make more sense, a rhyme scheme was added. Throughout this process, the program kept either misflagging two words as a rhyme incorrectly, or the other way around. Then, when the end of each line was not coherent, the solution was to add an ending context rather than an end word. This process took a lot of trial and error, but ultimately worked a lot better because if the line's ending word is, for example, "in," this doesn't make sense in a line like "it was in," but it does make sense in something like "when I left is the only time you came in." Giving the program ending context for each line, essentially the last two stop words, helped a lot with this obstacle that was overcome.

Additionally, one hope for the outcome of this project was also to analyze the notes and chords of each song, finding patterns for each artist, to be able to produce not only lyrics but also a sort of general melody or musical arrangement for the generated songs. Unfortunately, there were limitations in the ability to do so. The Python library that was attempted to use is called omnizart[11], which allows music transcription by uploading a audio file. The audio file was going to be uploaded using a Python library called youtube_dl[13], which extracted audio from a YouTube video link. This would have required the installation of omnizart in a M1 Mac, which was incompatible and not possible for the group's devices. Attempts were made to transcribe music on a M1 Mac with other Python libraries such as librosa[6], TensorFlow[5], and magenta[5]. However, given the time constraints, it was determined that this approach was inefficient and required lots of time. Eventually, an attempt to install Omnizart on an Intel Mac was made. However, there were

errors that persisted around the installation until other libraries and dependencies are installed as well, such as NumPy, cython[2], homebrew[3], and vamp[12]. This library also requires the user to work with version 3.6 - 3.9 of Python. A lot of time and effort was put into trying to get this installed and working in order to analyze the musicality of our songs. When exploring the omnizart GitHub[8], some people had errors working with the library even within the specified Python versions and that there are multiple unresolved issues, some being open from November 2020.

After figuring out how to install omnizart on an Intel Mac, a function to perform the vocal transcription from each song was attempted and ran, however, after running the code, it took a long time to run and ultimately crashed. The omnizart library was applied first to Lab 4: Hambot, so that it would be easier to perform and try on a smaller scale before using it in the Song Generator. After several attempts and due to limited time, a musical transcription and arrangement could not be performed or generated.

In terms of future work, the next step to this project would definitely be to make it a step further from lyrics and analyze chords as well. Evidently unable to make this work despite a big attempt for this deadline, with more time and resources, examining the chords, chord progressions, and melodies for each song, there is potential to find very interesting patterns. Hopefully, the creators of omnizart are still working, improving, and solving errors of the library so that it could be used in the future. Perhaps adding these into a generated song would be the next step in “completion,” allowing for a tune to be put to the words. After that

would be to analyze the rhythms of each song to see if there are patterns between each artist. Once these are taken, they can be added to the lyrics and chords to complete the generated song and make it playable. It would be very interesting to hear this final product, and how the cosine similarity maps would change when notes and rhythms are added to the song.

Although songs that mimicked the styles of each artist were successfully created, and while each line is coherent on its own, the complete songs do not always make much sense put together. In the future, with an improved expertise in natural language processing, this program could be improved to create entire songs that are coherent and likable.

Author Contributions

The entire group brainstormed goals and plans for the project. Jeremiah contributed the majority of the code, and Andrea reviewed the code and aided in the cosine similarity. Kaydence put in all the efforts in installing and working with omnizart to include notes and chords in the analysis and code in hopes of generating some sort of musical arrangement or melody. Although this was not possible, she spent time researching and trying to learn how to incorporate this. All members collaborated on the abstract and introduction, with Andrea and Samantha writing most of the sourcing/methods, analysis, and conclusion, with help and editing by Kaydence. The poster was made by Andrea, with help by Samantha.

References

1. Jameswenzel. “GitHub - Jameswenzel/Phyme: Python Rhyming Dictionary for Songwriting.” GitHub, https://github.com/jameswenzel/Phyme. Accessed 19 June 2023.
2. “C-Extensions for Python.” *Cython*, cython.org/#documentation. Accessed 22 June 2023.
3. *Homebrew*, brew.sh/. Accessed 22 June 2023.
4. Kim, Juliana. “And the Award Goes to Ai Ft. Humans: The Grammys Outline New Rules for AI Use.” *NPR*, 18 June 2023, www.npr.org/2023/06/18/1183013852/grammys-ai-music-awards.
5. *Magenta*, magenta.tensorflow.org/. Accessed 22 June 2023.
6. McFee, Brian, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. “librosa: Audio and music signal analysis in python.” In Proceedings of the 14th python in science conference, pp. 18-25. 2015.
7. *Musixmatch*. https://www.musixmatch.com/. Accessed 19 June 2023.
8. Music-and-Culture-Technology-Lab. “Music-and-Culture-Technology-Lab/Omnizart: Omniscient Mozart, Being Able to Transcribe Everything in the Music, Including Vocal, Drum, Chord, Beat, Instruments, and More.” *GitHub*, github.com/Music-and-Culture-Technology-Lab/omnizart. Accessed 22 June 2023.
9. NLTK’s list of english stopwords. Gist. Retrieved June 22, 2023, from https://gist.github.com/sebleier/554280
10. “OMNIZART: Music Transcription Made Easy¶.” *OMNIZART: MUSIC TRANSCRIPTION MADE EASY*, music-and-culture-technology-lab.github.io/omnizart-doc/. Accessed 22 June 2023.
11. “Omnizart.” *PyPI*, pypi.org/project/omnizart/. Accessed 22 June 2023.
12. “VAMP.” *PyPI*, pypi.org/project/vamp/. Accessed 22 June 2023.
13. “Youtube_dl.” *PyPI*, pypi.org/project/youtube_dl/. Accessed 22 June 2023.