Song lyrics generation

David Lekve
Bachelor Informatics / 2. year
davidlek@stud.ntnu.no

Abstract

This paper is for a project in the subject TDT4310-Intelligent Text Analytics and Language Understanding. This rapport will examine concepts of song lyrics generation using a combination of rule-based algorithms and a pre-trained language model. The method uses a given song as a starting point together with an analysis of all collected lyrics from an artist. The generated lyrics should resemble something that the chosen artist could have written. This paper presents the challenges with lyrics generation a starting point for the task.

1 Introduction

Song lyrics generation is distinct from general text generation. Song lyrics includes structure such as, rhyme, syllables and rhythm, rhyme schemes and segmentation into verse, bridge and chorus. Lyrics also includes a mood, topic and story line. Another key difference is that song lyrics has a more creative aspect to it, which can make the reader ignore irrational or not logical statements or even interpret them as creative.

Challenges with generating lyrics include that lyrics is supposed to be sung out loud and not read. Some artists are known for changing the pronunciation of words and even making up new words. Although it would be appropriate to use audio to analyse lyrics, this paper targets lyrics as purely text.

The goal of this project is to generate lyrics that resembles something a given artist could have written. To generate lyrics like a given artist is an arduous task. This is something that has been done with earlier studies, but rather with a verse phrase, than an entire song. The motivation for this work is to explore NLP and text generation in a creative manner.

To complete the task of creating a song that resembles an artist, well known Natural Language Processing techniques have been used analyse and categorize words and phrases. The program then replaces words in the song with chosen words from the artist based on frequency, syllable count or recommendations from a language model.

To substitute words in a lyric appropriately some preliminary aspects were recognized. The following have been regarded for this project:

- Syllable count is important
- The words should be of the same type (noun, verb, single, plural)
- The words should have been used in a similar manner
- The new phrase must have correct syntax

2 Background

This program has a simple structure. It can be summarized in two steps: first use rules to change the noun and adjectives and then use machine learning to change the verbs. The ambition is that the change of nouns and adjectives will make the text seem similar to the chosen and artist and that using machine learning to change the verbs, will make the sentence more logical.

2.1 Text analysis

The first step in the program is to collect all the named entities from both the song and the artist. Named entities are real world objects that are given names, this includes people, organizations, places and more. For this task, spaCy is used, which is a library with focus on language processing and information extraction. An example of spaCy's named entity recognition and its labeling is in Figure 1.

Figure 1: Highlighting of named entities found by spaCy. Visualization by spaCy Vizualizers. (From spacy.io)

The spaCy library is also used for attaining noun chunks. Noun chunks are phrases that have a noun as their base. A simpler definition is that is a chunk of words that consists of the noun and all the words describing the noun. Examples of noun chunks are "the big and juicy burger", "a red ball" and "these long avenues". To further categorize these noun chunks, stop words and part-of-speech tags have been used."Stopwords are words that are manually excluded from a text model, often because they occur very frequently in all documents in a corpus." (Bengfort et al., 2018) This includes words like "a", "the" and "you". And part of speech tags can be defined as "the classes assigned to parsed text that indicate how tokens are functioning in the context of a sentence." (Bengfort et al., 2018). Tokens are words or punctuation. Examples of parts-of-speech tags are nouns, verbs, adjectives, and adverbs. The library spaCy also provides a way for attaining stopwords and for labeling tokens (and entire texts) with POS-tags.

Throughout all of these changes, syllable count has a central role. To get the syllable count of a word or phrase, I have used a rule-based algorithm. It finds the amount of syllables based on a long list of rules written as if-statements.

2.2 Deep learning

This project uses a Masked Language Model (MLM) to predict words. A MLM predicts a masked word in a sentence. An example could be "Paris is the <mask> of France", where the model should correctly predict "capital". This project uses APIs provided by Huggingface Transformers to attain pre-trained models. The benefit of using pre-trained models are that they are trained on huge amounts of data and give state-of-the art results (huggingface.co). The Huggingface Transformers also provides APIs for fine tuning these models. Fine-tuning is a process of using deep learning frameworks to further train the model on a dataset specific to the task, which makes the predictions more suitable to the task. To do this the data has to be tokenized using the language models tokenizer and grouped into blocks of a specified size. On the more negative side, the model still makes biased predictions even after fine-tuning (huggingface.co). The model used in this project is the DistilRoBERTa base model. This is a distilled version of the RoBERTa-base model, which makes the model faster (huggingface.co). The model is intended to be fine tuned and is based on a lot of unfiltered data from the internet (huggingface.co).

2.3 Further definitions

Later in this paper I will introduce some alternative methods that need some background. I have therefore included some useful definitions in Table 1.

Casual language model	A language model that predicts the next token in a sequence of tokens.
Reccurent Neural Network (RNN)	A model with architecture that allows it to maintain long-term dependencies (Bengfort et al., 2018, p. 280)
Long Short Term Memory model (LSTM)	A variation of a RNN that allows for functions like "memory" and "forgetting" (Bengfort et al., 2018, p. 280). Very popular use is in for natural language generation tasks (Bengfort et al., 2018, p. 280)
Vectorization	The process of transforming non-numeric data (e.g., text, images,etc.) into vector representations on which machine learning methods can be applied. (Bengfort et al., 2018, p. 301)

Table 1: Definitions

3 Related Work

The motivating elements for lyrics generation is that it has both academic and industrial contributions Watanabe and Goto (2020). For example it could be used as a tool for creators of music Watanabe and Goto (2020). Although the are different approaches to the lyrics generation, none seem to generate a whole new song, but rather a verse or phrase. In my research I found three relevant studies that address central factors for my work.

3.1 Ghostwriter

Ghostwriter is a project that tries to create hip hop lyrics that resembles a given artist. The name, ghostwriter, is commonly used in the jargon of rap and hip hop as a person that writes and sells lyrics to another artist. The goal of writers, Potash et al., was to "produce a system that can take a given artist's lyrics and generate similar yet unique lyrics". The system uses an n-gram model and a LSTM trained on lyrics from the artist to create entire verses. The results were evaluated based on similarity and rhyme density, which is the number of rhymed syllables divided by total number of syllables. Similarity was calculated using the cosine distance from other verses from the artist. Their results show that the LSTM-model had better results (Potash et al., 2015). This work shows the strengths of using LSTM-models instead of n-gram. The disadvantage is that it can only be used to create verses, which is not suitable for my task.

3.2 LaserTagger

Malmi et al. (2019) casts text generation tasks as text editing task with their LaserTagger process. The method accomplishes tasks by tagging the text (with for example KEEP, DELETE) and uses these tags to generate a final output text (Malmi et al., 2019). The results show that it outperforms other techniques especially when the training data is limited (Malmi et al., 2019). From this we get that text editing is a valuable way of typical text generation tasks. Disadvantages for this project is that have only proven this for text generation tasks that have overlapping inputs and outputs, which is something we want to avoid.

3.3 The Perfect Rap Lyrics

The Master Thesis of Jensen and Sørhaug have demonstrated a way to both analyze rap lyrics and generate it. To analyze and rate rap lyrics they created a framework designed to detect ten distinct rhyme metrics (Jensen and Sørhaug, 2021, p. 23). Lyrics was rated on how often the different rhyme metrics were used, and the aggregated score was called rhyme complexity. (Jensen and Sørhaug, 2021, p. 23) The takeaway from this research is that rhyme can be analyzed and used to rate generated lyrics, especially rap.

4 Architecture

To use the program you have to select an artist and a song from the dataset. The program will use the chosen song as a foundation for the song to be created. The program then analyses both the chosen song and artists and uses chosen techniques to replace phrases and words to generate a new song lyric. The architecture of the program is illustrated in Figure **??**. This figure shows that the architecture includes the following steps:

- Retrieve named entities and noun phrases from chosen song lyric
- Retrieve named entities and noun phrases from all of the lyric from the artist
- Categorize the noun chunks that start with stopwords and exchange these with noun chunks from the artist that start with the same stopwords. The noun in the chunk must also have the same POS-tag (NN, NNP or NNS).
- Replace named entities that have the same label, same syllable count by frequency.
- Replace noun chunks that do not start with stop words. The noun chunks must also have the same POS-tag and they are sorted by freqeuncy.
- Fine-tune the destilled RoBERTa model on all of lyrics from the artist
- Mask the first verb in a lyric line from the song that is not a stopword
- Use the fine tuned language model to predict new words for the masked verb.

The exchange of named entities is the least complex procedure. For this process I found that keeping original capitalization was important for achieving the best results. After retrieving the entities that the spaCy pipeline recognized, I sorted them by frequency of appearance. I looped through all the entities found in the song, and searched for a possible match with the entities found within all of the chosen artists lyric. To be a match, the entities had to have the same label (for example "PERSON", "GPE" or "FCE"). They also had to have the same amount of syllables and the entity from the artist could not have been used before. If a match was found, all occurrences of the entity in the song lyric were exchanged. Because of the sorting by frequency, the most frequent entities in the song got exchanged with the most frequent entities used by the artist.

The next procedure is to attain and sort all the noun phrases. This procedure is divided in two parts: one for noun phrases that begin with stop words, and one for those that do not. This segregation is used to achieve more suiting results. Because of this split, there is a difference between how 'such a yellow car' (where 'such' and 'a' are stop words) and 'yellow car' will be handled. This is done to make sure that the noun phrases have been used in a more similar manner. Another requirement is that the noun in the phrase must have the same POS-tag. SpaCy provides four different english POS-tags for nouns: NN - noun, singular or mass, NNP - noun, proper singular, NNPS - noun, proper plural, NNS - noun, plural. These are retrieved by using the attribute "tag_" and not "pos_" in spaCy. The effect is that a plural noun does not get replaced by a singular noun and so forth. In addition noun chunks can not be same as an named entity. The words are swapped in the following order: noun chunks that start with stop words, entities and then the remaining noun chunks.

By now, most of the adjectives and nouns have been exchanged by a hopefully more suitable phrase from the chosen artist. The last task is to substitute the verbs. For this task I used a fine-tuned MLM. The MLM is fine-tuned on all the lyrics collected from the chosen artist from the dataset. Afterwards the text is tokenized with the models tokenizer and then grouped in to blocks of the same size. Afterwards, the program loops through line after line in the song and masks the first verb that is not a stopword. Because of this, the program only changes maximum one verb per line. The MLM then predicts a new word. The program looks at the top five recommendations and chooses the one that has the same syllable count as the original verb.

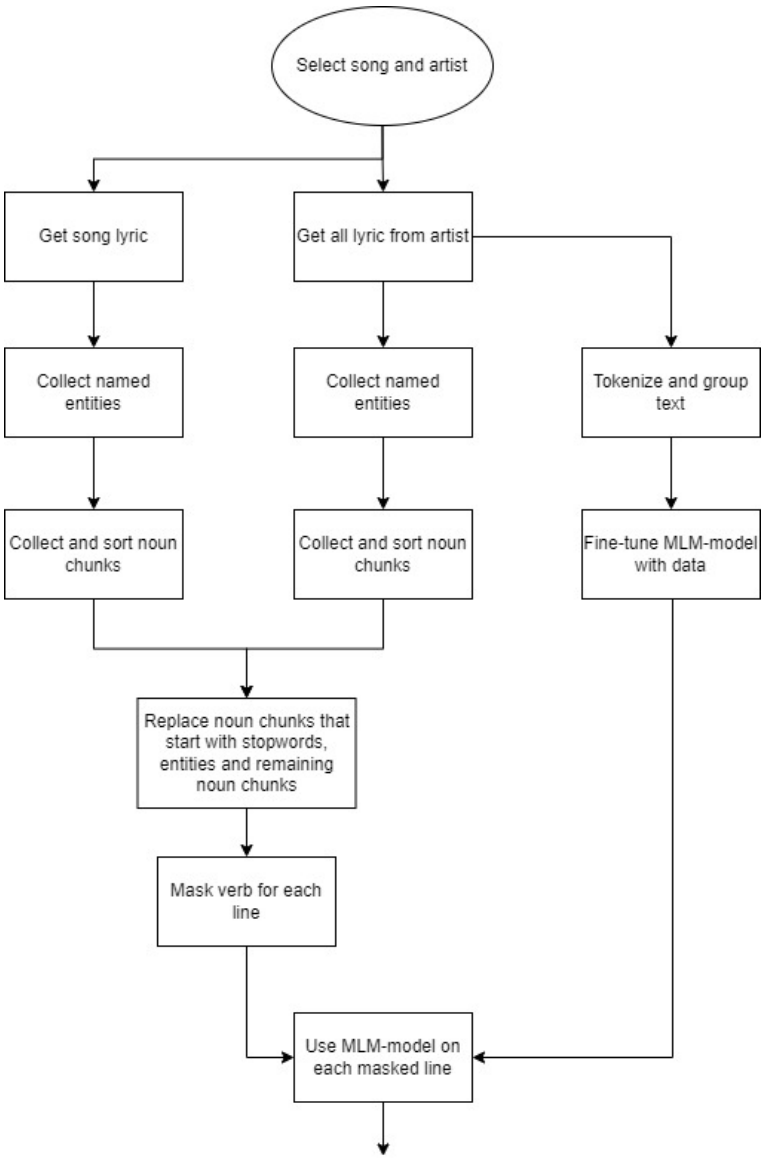

Figure 2: Architecture of this program. Drawn with draw.io

5 Experiments and Results

To highlight the strength and weaknesses of this program, I have chosen to present examples consisting of a few lines from the generated lyrics. The artists and songs have been chosen from different genres and cultures to get some drastic changes in the lyric.

5.1 Experimental Setup

The database that is used is the 'Song lyrics from 79 musical genres' found on Kaggle. It uses data scraped from the website vagalume.com.br and concists of around 191 thousand songs.

For the fine-tuning of an MLM, the DestilRoBERTa base model is used. The song lyrics is split in a train, test ratio with 10% as testing data. The model is fine tuned in three epochs.

The program lets the user specify which labels of named entities to exchange. For the presented results the following labels have been used: PERSON, FAC, ORG, GPE, LOC.

The stopwords used are the default spaCy stopwords with the following additional words: "ya","ai" and "™".

5.2 Experimental Results

5.2.1 Example 1

The first example is where the program uses the pop song "Empire State of Mind part II" and the renowned hip hop artist Kendrick Lamar. The song is a praising song about New York, the growth place of the artist, Alicia Keys. Kendrick Lamar is from Compton, California and is known for his storytelling, imagery, wordplay, passion and varied song structures.

Original:
Hail a gypsy cab, takes me down from Harlem to the Brooklyn Bridge

Kendrick Lamar version:
God a swimming pool
take me down from Nile to the rough Compton

Here we can notice a lot of strengths and weaknesses from the program. Firstly we notice the strange and syntactically wrong first line "God a swimming pool". What has happened here is that "a gypsy cab" has been exchanged with the most frequent noun phrase from Kendrick Lamars music that begins with "a" and where the phrases root word is in singular form (NNP). The most frequently used is "a swimming pool". Here the algorithm has inserted a new line after swimming pool, which has occurred because the spaCy library has assumed the new line as a part of the noun phrase. The word "Hail" has also been assumed as a noun phrase in singular form by the spaCy library and the program has then swapped it for a common singular noun phrase from Kendrick Lamar, which is God. Therefore this line is gibberish. Using God is common for Kendrick Lamar and it makes sense that he would write new lyrics including "God", but "a swimming pool" is a common noun phrase because of his song "Swimming Pools (Drank)". I think that using the noun phrase "a swimming pool" therefore is a cliche for Kendrick Lamar and is not something he would typically use in a new song. If the world "Hail" would get rightly assumed as a verb, the result would be "in a swimming pool", which makes more sense. Without a fine-tuned model, but using the the plain destilRoBERTa base model, the lyric line would have become "Inside a swimming pool".

The next line contains two entities "Harlem" and "the Brooklyn Bridge". SpaCy has correctly labeleb "Harlem" as "LOC" - for location."the Brooklyn Bridge", should be labeled as "FAC" which are buildings, airports, highways, bridges, etc, but the registered entity is "the Brooklyn Bridge(newline)Someone". Therefore the program also registered "the Brooklyn Bridge" as a noun phrase, and it was exchanged for "the rough Compton", which is something that fits very well in a Kendrick Lamar song, but it does not fit together with Nile in that context.

To summarize the above example, the quick verdict is that it is nonsense. One can recognize that the nouns used are typical for Kendrick Lamar, but some have been used in a cliche manner. By this I mean that he would probably not make new lyrics where both "swimming pool" and "Nile" is used. The two lines do not have anything in common (other than being relevant to Kendrick Lamar), which is to be expected with a program that examines words and lines separately without context.

If look at the whole new created song in section 8.1, there are definitely major changes that have been made. The song now contains a lot of profanity and the song is no longer about New York, but rather Compton (or a place called "fuckin"). It is still possible to get the feel that the song is based on "Empire State of Mind part II", and it does not have the typical structure of a Kendrick Lamar song, which often contain long lines of clever rhymes.

5.2.2 Example 2

The next example that I have decided to give attention, is when the program tries to change Drake's pop and RB song " Hotline Bling" to make it look like country artist, Billy Ray Cyrus wrote it.

Original from Drake:
Call me on my cell phone
Late night when you need my love
I know when that hotline bling

Billy Ray Cyrus version:
Call me on my mullet
Late night when you broke my heart
I know when that hotline bling

In this example, the first phrase is grammatically correct, but it makes no sense. A reason behind this is that "call" is listed as a stop word, and therefore it does not qualify as a verb the program should change. If it would have been changed, the new word would be "Put", which hardly makes the phrase more logical. The problem is that there is no logical word to exchange "Call" with. This exemplifies that one can not completely rely on changing noun phrases without looking at the rest of the sentence. The phrase "me on my" is unique, in that there are few noun phrases that make sense regardless of what word is used before the phrase.

The next line is grammatically correct and it makes sense. "my love" has been exchanged with "my heart", and the verb "need" has been changed to "broke". The exchange of verbs demonstrates the strength of using a neural network model, where "broke" is a great fit before "my heart". "Late Night" has been categorized as an entity with the label TIME and is therefore not replaced.

The last line is exactly the same. This is because hotline bling has not been registered as neither a noun chunk or entity. The cause of this is the use of "bling". If the phrase is changed to "I know when that hotline ring", the program registers "when that hotline ring" as a noun chunk. This demonstrates that the program has a weakness for words that are used in a new, abstract ore creative way, which can be expected. The word "know" has remained the same, as the model found it the most suitable.

If look at the entire song created in section 8.2, the results are very similar to the original song. The cause of this is that the song is pritty simple, and often repeats the phrase "hotline bling", which has not been exchanged.

6 Evaluation and Discussion

In general I find the results are appropriate in some places, but there is also a lot of nonsense. In most cases it is easy to recognize the song that the lyric is based on. As a positive, I think that it is possible to recognize at least the genre that the artist is known for. Further, it is safe to say that it is improbable that a human wrote the lyrics. A big impact on the result is how correctly spaCy categorizes the words. Bad results are bound to happen in phrases where there is little flexibility for word changes. An earlier

mentioned example of this is the phrase "Call me on my cellphone". Other examples of little flexibility, is when there are contrast or similarities. This is exemplified in Example 1, where the Nile and the rough Compton should have some relation to each other, but they do not. To summarize, the results are as expected for a program with this simplicity and size of rules.

The definition of good lyric is subjective. It would be desirable to involve a group of people to rank the results on factors like: poetic, written by human/computer, good/bad, creative, but this has not been possible. It would also be desirable to implement rating based on language factors as the one presented by Jensen and Sørhaug. In this evaluation the results are interpreted subjectively by the author, but I have striven to be as objective as possible. In short, syntactical errors and unreasonable statements have been judged as poor performance.

6.1 Strengths of the program

By using a song as a base, the structure of the generated lyrics resembles a song. Although possible, I think it would be difficult to build lyrics with the structure of a whole song with a casual language model. Especially considering the division of verse, bridge and chorus, as these parts often have very different structure. In comparison, this is easy to achieve by "stealing" the structure of a song and replacing words using rule-based replacements. Furthermore, this rule-based replacement makes it convenient to incorporate a masked language model to predict suiting words for an artist. As an example, if an artist mostly sings about riding horses, the results from a masked language model can be more incited towards a word that fits the artist with a phrase like "<mask> from the barn", rather than "<mask> whatever".

The most essential rule for changing a noun chunk or entity is that it should be of the same type. For entities this means that it must have the same label, for example: "PERSON". If the song is about a place, the generated song will also be about a place and so on. The effect of this(if words are correctly labeled), is that the generated song preserves structure and stays cohesive. Although it might not be the best method, sorting on frequency makes the program take consideration of relevance to the artist.

Another factor, for increasing correct syntax, is the use of stop words. By sorting and swapping the noun chunks by stopwords, the chance of ruining the syntax is suppressed. As an example, lets look at the phrase "I like my own car". In this context, the program will change the noun chunk "my own car" with something from the artist that begins with "my own", as both of these words are stop words. This also improves the chance of the words being used in a similar manner.

6.2 Improvements and future work

6.2.1 The means of replacing words

The program has many obvious improvements. The most significant in my opinion is the exchanging of words based purely on frequency and syllable count. From the song "Empire State of Mind", the results of entity exchanges were acceptable, but would have been much worse if it was a resentful, rather than admiring, song about New York. Then the artist would probably never use the entities that the algorithm recommended. The general solution for this is to use more information so that the context the word is used in, is known.

For this task, Long Short Term Memory (LSTM) models are excellent, because they adapt to earlier generated words to stay on track with the context. The strengths of LSTM-models is demonstrated with Ghostwriter and The Perfect Rap Lyrics. In these papers only a part of a song is created and it therefore does not prove good results for generating a whole song using an LSTM-model. However, this method would be very useful when their is little flexibility exchanging words, due to strengths in keeping context. A similar approach is the use of n-grams and a hidden Markov model. Although this would make it so that nearby words and therefore context would influence the replacements, LSTM-models seem to be superior, as shown in (Potash et al., 2015).

A completely different approach would be to analyze sentimental value. With this, one could categorize words, phrases and songs to categories as: resentful, neutral or pleasing. For this task, deep learning models could have been used, which have shown state-of-the-art results for sentiment analysis tasks (Zhang et al., 2018). This categorization could be used to limit the chance of the above mentioned

problem. Future work could also includes the implementation of analysis of mood, topic and storytelling. Methods for estimating this information has been developed Watanabe and Goto (2020).

Retrieving factual data on both the chosen artist and the artist that has written the song is another improvement. With the example "Empire State of Mind part II", the program could acknowledge that New York is the birth place of the artist, Alicia Keys. Furthermore, information about the names of family members, known nicknames of the artist and more could be useful in generating more convincing results. Another example is that "hotline bling" could be acknowledged as a song name and be tagged and replaced as such. A useful tool for gathering information like this is Wikidata.

To further improve results, word vectoring could be used. With this, one could accomplish adding and subtracting different genres, or even artists, from a word. For example if the word to be changed is 'lambo' (an often used slang in hip hop for Lamborghini) and you subtract hip-hop from it, maybe it becomes 'car', and then you add country to it, and maybe it becomes 'pickup'.

TF-IDF could also be helpful in understanding how important a word is for an artist. The corpus to measure a song with could either be all English songs or just a genre and even just the artist. This could be used to achieve important phrases from an aritst, but it could also be used to notice cliches. For example Kendrick Lamar sings about Compton in multiple of his songs, which proves that it is an important word for him and a word he will probably use again. The Nile though, is only used in one song. This could be noticed using TD-IDF with the artist as corpus and compare the value with other songs. With this, one could temper unlikely use of phrases and words.

6.2.2 Preprocess

Preproccessing of the data should have been adapted for different tasks. In this program, original capitalization was best suitable for named entity recognition, but this data was also used for every other task. This has led to weaknesses that can be corrected. For example, when stop words are capitalized in the beginning of a line, the program makes a difference between these and lowered stop words when sorting noun chunks. To get the best results, using both capitalized and lowered text should have been used for different tasks.

A common bug for spaCy is that it includes the newline symbol ("\n") inside of entities. This has caused the results to have a different amount of lines than the song it was based on. This should have been handled in the preprocessing phase.

Some tokens should have been removed from the data accordingly with the tasks. For example ad-libs and words that are not directly part of the lyrics should have been removed. This includes words as '[chorus]' or '(First verse Lennon)'. It does not make sense to fine-tune the MLM on these words, when the model is only used to to predict masked verbs.

Stemming and lemmatization are two forms of text normalization. This "ensures that different forms of tokens that embed plurality, case, gender, cardinality, tense, etc., are treated as single vector components, reducing the feature space and making models more performant." Watanabe and Goto (2020). For this program, text normalization would yield more telling results for the frequency of noun chunks. For example "book" and "books" would be treated as the same word.

6.2.3 Structure

Although syllable count was acknowledged as an important factor in the preliminary research, it has been integrated incorrectly. This causes the structure of the generated song to not match the chosen artist, but rather the chosen song, which is especially lacking when the songs are of different genres. Having the same amount of syllables prohibits the program from achieving a structure that resembles the given artist and even genre of the artist. As an example, Kendrick Lamar, would probably use more syllables per line than the other artists. Syllable count would still be useful in achieving an optimal structure, but I think it should be used in a different manner. By counting the average syllables per line, one could use this number to make lyrics seem more like a given artist.

Using rhyming is also be a big advancement for the program. Rhyming is essential in lyrics, especially hip hop. For this task the python framework "pronouncing" could have been used or the framework to analyse rhyme made by Jensen and Sørhaug.

Songs are known for being segmented in to verse, chorus, bridge and sometimes intro and outro. Although machine learning approaches are not available Watanabe and Goto (2020) for this task, it is something that would give beneficial information for lyrics generation.

6.2.4 Optimizing the language model

The fine tuning of the model is not optimized. Optimizing the number of epochs and other parameters is conventional, but it has been outside the scope of this project. It would also be interesting to compare results with a model purely trained on the lyric from an artist. Although this might give replacements more suitable for the artist, the strength of a pre-trained model is that it is trained on more unique circumstances. A major drawback from the program is that it can only change one verb per line.

7 Conclusion

This paper presents the complexity of song lyrics generation and a possible fuse of methods to handle lyrics generation. These contributions are not significant in the sophisticated world of text generation, but can function as introductory for newcomers to the field. This work has also achieved its goal to a subjective and deficient degree. The results show that the possibility for collecting and utilizing information from lyrics generate new lyrics.

References

Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda. *Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning*. O'Reilly Media, Inc., 1st edition, 2018. ISBN 9781491963043.

Christian Ziegenhahn Jensen and Espen Sørhaug. The perfect rap lyrics-ai generated rap lyrics that are better than lyrics from existing popular and critically acclaimed rap songs. Master's thesis, NTNU, 2021.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. Encode, tag, realize: High-precision text editing. *arXiv preprint arXiv:1909.01187*, 2019.

Peter Potash, Alexey Romanov, and Anna Rumshisky. Ghostwriter: Using an lstm for automatic rap lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1919–1924, 2015.

Kento Watanabe and Masataka Goto. Lyrics information processing: Analysis, generation, and applications. In *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, pages 6–12, 2020.

Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.

8 Appendix

8.1 Generated Kendrick Lamar version of "Empire State of Mind part II"

money, fuckin (2x)

grew up in a fan,
That is famous as a world of picket signs
Bitch is always loud
There are people all around
And the streets are mean
If I could make it here
I could make it anywhere
That's what they say
get my vibe in hoes

Or my name in Things going down Metro

Even if it ain't all it seems
I'm a traffic jam of bitches
Baby, I'm from

(Compton)
fuckin, everybody where bitches are made of
There's nothing you can't do
Now you're in fuckin
These streets will make you feel Blow Hurt people will inspire you
do it for fuckin, fuckin, fuckin

On the city, there ain't never a nigga
problems, so hard
Such a melting pot on the mirror selling life
Babies pray to God
God a swimming pool
take me down from Harlem to the rough Compton
Someone was tonight with a woman
For more than from an empty fridge

I'm not to make it by any means
I'm a traffic jam of bitches
Baby, I'm from

(Compton)
fuckin, everybody where bitches are made of
There's nothing you can't do
Now you're in fuckin
These streets will make you feel Blow Hurt people will inspire you
do it for fuckin, fuckin, fuckin

(Compton)
One curse in the truth for the illuminati, big bitches, all looking pretty No place in the truth that can compared

Put your feelings in the truth
communication say yeah, yeah yeaah

(Compton)
fuckin, everybody where bitches are made of
There's nothing you can't do
Now you're in fuckin
These streets will make you feel Blow Hurt people will inspire you
take it for fuckin

8.1.1 Changes to the song above

Named entities:
('Grew', 'don'), ('Broadway', 'Metro'), ('the Brooklyn Bridge \n Someone', 'Long Beach Boulevard \n Flagging'), ('Chorus', 'Compton'), ('New York', 'fuckin'), ('Harlem', 'Nile')

Noun chunks without stopwords:
('dreams', 'bitches'), ('concrete jungle', 'everybody'), ('brand', 'Blow'), ('new \nBig lights', 'Hurt people'), ('Oooh oooh', 'money'), ('movie scenes', 'picket signs'), ('Noise', 'Bitch'), ('sirens', 'people'), ('lights', 'hoes'), ('marquees', 'Things'), ('Ladies', 'problems'), ('rock', 'life'), ('Preachers', 'Babies'), ('Hail', 'God'), ('big dreams', 'dollars'), ('Everybody', 'communication'),

Noun chunks with stopwords:
('a pocketful', 'a traffic jam'), ('a town', 'a fan'), ('a place', 'a world'), ('a curfew', 'a nigga'), ('a gypsy cab', 'a swimming pool'), ('a hunger', 'a woman'), ('the air', 'the world'), ('the streets', 'the streets'), ('the avenue', 'the city'), ('the corner', 'the mirror'), ('the Brooklyn Bridge', 'the rough Compton'), ('the big city \n Street lights', 'the illuminati'), ('the world', 'the truth'), ('my face', 'my vibe'), ('These streets', 'These streets'), ('any means', 'any means'), ('One hand', 'One curse'), ('all looking pretty \n No place', 'all looking pretty No place'), ('your lighters', 'your feelings')

Verbs:
('don', 'grew'), ('Seeing', 'get'), ('found', 'going'), ('got', "'m"), ('feel', 'feel'), ('Hear', 'do'), ('work', ','), ('pray', 'pray'), ('swimming', 'swimming'), ('Takes', 'take'), ('sleeps', 'was'), ('going', 'not'), ('got', "'m"), ('feel', 'feel'), ('Hear', 'do'), ('feel', 'feel'), ('Hear', 'take')

8.2 Generated Billy Ray Cyrus version of "Hotline bling"

You used to call me on my, you used to, you used to
You used to call me on my mullet
Late night when you broke my heart
Call me on my mullet
Late night when you broke my heart
I know when that hotline bling
That can only mean one thing
I know when that hotline bling
That can only mean one thing

Ever since I picked the wrong time, you
Pick a touchy subject for yourself now
Everybody leaves and I feel left out
Girl you let me down, you got me stressed out
And ever since I left the wrong time, you
Started wearing less and goin' out more
Lyrics of baby out on the fastest horse
Cyrus' with some steps I've never seen before

You used to call me on my mullet
Late night when you broke my heart
Call me on my mullet
Late night when you broke my heart
I know when that hotline bling
That can only mean one thing
I know when that hotline bling
That can only mean one thing

Ever since I picked the wrong time, you, you, you
You and me we just don't get along
You make me feel like I did you wrong

the wings where you don't belong
Ever since I picked the wrong time, you
You got exactly what you asked for
Running out of guns in your picture
Cyrusg with some steps I've never seen before

You used to call me on my mullet
Late night when you broke my heart
Call me on my mullet
Late night when you broke my heart
I know when that hotline bling
That can only mean one thing
I know when that hotline bling
That can only mean one thing

These days, all I do is
color if you bendin' over backwards for someone else
color if your sweet kiss' over guitars for someone else
Doing wings I love you
Somebody for someone else
You don't know no one else
You don't need nobody else, no
Why you never alone
Why you always touching love
Used to always, at time, be a thin line
You was in the way
You should just be yourself
Right now, you're someone else

You used to call me on my mullet
Late night when you broke my heart
Call me on my mullet
Late night when you broke my heart
I know when that hotline bling
That can only mean one thing
I know when that hotline bling
Ever since I picked the wrong time

8.2.1 Changes to the song above

Noun chunks without stopwords:
('Everybody', 'Everybody'), ('Glasses', 'Lyrics'), ('champagne', 'baby'), ('Hangin', 'Cyrus'), ('places', 'things'), ('exactly what', 'exactly what'), ('pages', 'guns'), ('Wonder', 'color'), ('backwoods', 'guitars'), ('things', 'wings'), ("Gettin' nasty", 'Somebody'), ('road', 'love'), ('home', 'time')

Noun chunks with stopwords:
('my cell phone', 'my mullet'), ('my love', 'my heart'), ('one thing', 'one thing'), ('the city', 'the wrong time'), ('the dance floor', 'the fastest horse'), ('the zone', 'the way'), ('a reputation', 'a touchy subject'), ('a good girl', 'a thin line'), ('some girls', 'some steps'), ('your passport', 'your picture'), ('your rollin', 'your sweet kiss')

Verbs:
(’need’, ’broke’), (’need’, ’broke’), (’know’, ’know’), (’mean’, ’mean’), (’know’, ’know’), (’mean’, ’mean’), (’left’, ’picked’), (’Got’, ’Pick’), (’knows’, ’leaves’), (’got’, ’let’), (’Cause’, ’And’), (’seen’, ’seen’), (’need’, ’broke’), (’need’, ’broke’), (’know’, ’know’), (’mean’, ’mean’), (’know’, ’know’), (’mean’, ’mean’), (’left’, ’picked’), (’feel’, ’feel’), (’Going’, ’the’), (’left’, ’picked’), (’got’, ’got’), (’Running’, ’Running’), (’seen’, ’seen’), (’need’, ’broke’), (’need’, ’broke’), (’know’, ’know’), (’mean’, ’mean’), (’know’, ’know’), (’mean’, ’mean’), (’taught’, ’love’), (’need’, ’know’), (’need’, ’need’), (’stay’, ’,’), (’need’, ’broke’), (’need’, ’broke’), (’know’, ’know’), (’mean’, ’mean’), (’know’, ’know’), (’left’, ’picked’)