Evaluating Lyric Generation Models for Target Artists

Tristan Hsieh
tch992
tristan.hsieh@gmail.com

Sooyong Lee
sl46538
sooyonglee1@gmail.com

Abstract

Text generation, or the task of generating text with the goal of appearing indistinguishable to human-read text, is a task that has been widely explored in natural language processing. In this paper, we apply this notion of text generation to song lyric generation in order to create lyrics that appear indistinguishable to those written by the original artists. We utilize different lyric generation models and determine their effectiveness through quantitative and qualitative measures.

1 Introduction

Text generation is a popular and critical task in natural language processing, and its applications involve machine translation, text summarization, and chat bot systems. In this paper, we focus on lyric generation for specific artists. Song lyrics often exhibit a different grammatical semantic when compared to a typical text corpus, and are typically more representative of colloquial speech. Additionally, song lyrics differ from a typical text corpus because of their brevity, which results in shorter and more repetitive sentences. However, because different artists employ different vocabularies, we implement lyric generation models that factor in this difference in order to generate lyrics that are uniquely associated with an artist.

After implementing a base model for lyric generation, we evaluate the performance of our neural network model and n-gram model. We quantitatively evaluate the performance of these models through perplexity while also considering the aforementioned qualitative characteristics of song lyrics. We examine the capabilities and limitations of our models, and determine the more appropriate model for our task through quantitative and qualitative metrics.

2 Related Work

The idea of lyric generation has been attempted before, in the context of producing lyrics of a specific genre of language. In a recent work, (Potash et al., 2015), rap music lyrics were generated using a type of recurrent neural network a Long Short-Term Memory language model (LSTM). This study evaluated its model with an algorithm that determined how similar a set of lyrics were to another set of lyrics. Another study (Wu et al., 2018) generated lyrics in the style of a specific artist, working with the mandarin language. This study also used an LSTM neural network which was evaluated using perplexity.

3 Lyric Generation Models

In contrast to the aforementioned works on lyric generation, we build models that generate lyrics in the style of numerous artists. Our models should consider the target artist’s unique vocabulary and speech style in order to generate lyrics that closely resemble authentic lyrics.

3.1 Base Model

For our base model, we implement a model which generates sentences of varying lengths between 8 and 14 words with the knowledge that sentences in song lyrics are typically short in length. The model first processes the corpus representing the target artist and updates the word frequencies in a dictionary. Then, the model generates sentences by selecting the most frequently used words to populate each sentence.

3.2 N-gram Model

The N-gram model is a bigram model which also generates sentences of varying lengths. Unlike the base model, the bigram model considers word context, which results in sentence generation that is not of a fixed length. Instead, the model generates sentences that either terminate when the model determines that the sentence should terminate based on the current word's context, or when the sentences have reached the sentence length limit of 14 similar to the base model. The N-gram model generates sentences by considering the previous word and randomly selecting out of the 5 most likely words to follow given a word context. Through this methodology, the N-gram Model generates sentences more dynamically when compared to the base unigram frequency model.

3.3 Neural Network LSTM Model

The neural network model works by splitting the lyrics of the training set into little snippets of a predefined length. The model then stores what the next character after each of these snippets are. Using this information, the model takes a snippet at random and generates what the most likely next letter would be. Once the model has printed out four hundred characters, the model updates its weights using an RMSProp optimizer algorithm, and the entire process is repeated with a different randomly generated snippet. This algorithm also used a variable known as temperature to determine its results. The lower temperatures generated more conservative lyrics, where the model was more likely to print coherent words but also more likely to have a lot of repetition. The higher temperatures caused the model to make more risky choices, meaning the lyrics would often contain nonsense words. This algorithm was tested over 60 epochs per artist, and four different temperatures (0.2, 0.5, 1.0, 1.2) per epoch.

4 Experimental Design and Results

4.1 Dataset and Data Collection

The data for four different artists (Drake, Queen, Iron Maiden, and Eminem) was collected using both Spotfy's API and a song lyric API from orion.apiseeds.com. Spotify's API was used to return track titles when given an artist's name, and these track titles were fed into the song lyric API from orion.apiseeds.com which returned lyrics when given an artist and track title. The lyrics acquired from this API is considered to be the representative text corpus for each artist, and these lyrics were formatted into a JSON format for easy processing for our lyric generation models.

The training data was in the form of a dictionary. Each key was an artist name, and each value was a list containing songs. The songs were stored as strings with the lyrics inside. The hyperparameters such as the number of epochs and temperature were tuned using the development set.

Line 1: who have with this for who one want take her
Line 2: wan been oh if need ? yeah but man love
Line 3: go 're 'll know your when that need down i
Line 4: how right right you shit have get at it 's
Line 5: still if you how : been tell at now ca

4.2 Results

We utilized our models to generate lyrics which emulated Drake, Queen, Iron Maiden, and Eminem. Below are some lines taken from lyrics generated by the base model for Drake:

In comparison, here are some lines generated by the N-gram model for Drake (explicit content):

Qualitative analysis of the lyrics generated by the base model and N-gram model show that the N-gram model produces lyrics which are more coherent and intelligible. This is because the N-gram model considers word context and chooses words that are likely to follow a given word. Because of this characteristic, when compared to the random selection of words in the base model, the N-gram model generates new lyrics which are more understandable.

Applying a quantitative analysis of the

Line 1: are friends if they say you 're a grammy speech because we gon '
Line 2: i get it 's a good girl and i got ta let it 's
Line 3: you got the same as you do it up on a nigga i got
Line 4: oh my face , you 're the way i get to get to get
Line 5: oh michael jordan with my niggas keep reaching and they do it 's a

generated lyrics by evaluating the perplexities confirms the prior qualitative analysis and finding that the N-gram model performed better than the base model. The perplexity of the lyrics generated by Drake's base model was 1004.078 (1109.84 on average), whereas the perplexity of the lyrics generated by Drake's N-gram model was 161.749 (152.11 on average.)

For the Neural Network, the perplexity tended to be lowest only after 4 epochs, on the two lowest temperature settings. Here were the drake lyrics with the lowest perplexity:

```
…en i done
that i don't be care think i get me the
shit think i wanna got the shit shit
they gotta girl i be time the think i
get me done the bust i know when i
get the don't wanna that i get the
simes to be think i get the don't gotta
danna get the shit that i done
think i get it think all the shit think
all the buck on the shit shit shit i
know when i got the buck the light
and i ded to be to get…
```

When analyzing the data qualitatively, it seemed that the lowest perplexity lyrics resembled the original artist the most. The higher perplexities generated lyrics that contained a lot of nonsense words. However, there were still a few problems with the lowest perplexity lyrics. While the generated lyrics made sense on a word to word scale, in the context of the entire sentence they were still nonsensical. The model would also struggle with repetition. For example, several Iron Maiden songs contain the lyrics "Oh, Oh, Oh" and so the model would sometimes generate a never ending list of "oh"s to imitate this, while still scoring a low perplexity. Another potential issue is the model's tendency to generate lyrics that too closely resemble a specific song. For example, the Queen model would often generate the sentence "I'm the invisible man," which is directly taken from an existing Queen song.

5 Conclusion

While the neural network model scored a lower perplexity when compared to the N-gram model and base model, the neural network model still failed to generate lyrics that made sense in the context of an entire sentence. While the N-gram model also did not generate sentences that were entirely intelligible, the lyrics generated by the N-gram model appeared qualitatively better. It suffered less from the sentence context problem pertaining to the neural network model because it at least considered word to word contexts, and the N-gram model did not generate any nonsensical new words. This could be seen as a potential disadvantage to the N-gram model as well, because artists are bound to release new lyrics with new words, and so the lyric generation model should be able to create new words. However, the neural network model failed to create legitimate new words that made contextual sense; even though the neural network model generated lyrics with a "character-to-character" methodology, this did not seem to provide any advantage in our lyric generation task because the model never constructed any relevant words.

Based off these observations, we decided that our N-gram model was more suitable for lyric generation despite its quantitatively worse performance. As mentioned when analyzing the lyrics produced by the neural network model, the neural network model's generated lyrics were often too redundant; while songs are typically more redundant than casual speech, the redundancy that song lyrics employ are usually line-to-line redundancy, rather than words occurring consecutively redundantly. However, there are still improvements we could have considered with our N-gram model; one way of improving the generated lyrics would have been to consider the grammatical structure of the sentences by involving POS-tagging. Although songs are typically looser in terms of grammatical structure, considering grammatical structure in the generated lyrics may lead to sentences which make more sense.

A way of improving the neural network model would be to introduce a corpus which could be compared to when generating words, thus resulting in the generation of legitimate English words. Additionally, the corpus could also incorporate words that are strongly associated with the lyrics that already exist for the artist in order to introduce new vocabulary that an artist would likely use. The neural network model could also be improved by continuously evaluating the generated lyrics as a whole. This would help with repetition at the expensive of computational performance. For example, an Iron Maiden song would likely not have eighty "oh"s in a row, but evaluating every single "oh" every time another "oh" is generated would be computationally expensive. Finally, we could change our neural network model to generate lyrics on a word to word basis in stead of a character to character basis; this would remove many made-up words and could help with grammatical and semantic issues.

6 Who did what?

Sooyong contributed to most of the writeup, the baseline model, and the n-gram model. Tristan helped with the neural networks, creating the dataset using APIs, and finding related work.

References

Jeff Heaten. 2019. Text Generation with Keras and TensorFlow (10.3). Youtube. https://www.youtube.com/watch?v=6ORnRAz3gnA

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2015. 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. Association for Computational Linguistics

Xing Wu, Zhikang Du, Yike Guo & Hamido Fujita. 2018. Hierarchical attention based long short-term memory for Chinese lyric generation. In *Applied Intelligence volume 49*, pages 44–52. Springer