Proposal for Automatic Lyric Generation Model

Submission by Second Degree Burn

Abstract

This document outlines a plan to create a model that can generate song lyrics representing various genres and artists. Sequence-to-sequence modelling will be used, with training data from thousands of songs. Outputs will be evaluated based on grammar, rhyming, song structure and genre classification. The goal is to achieve the baselines in at least two of these categories before the milestone. Much of the work will be done together as a group, but each team member will also have individual responsibilities.

1 Introduction

Songwriting, like other art forms, is seen as something very human. When paired with the right music, lyrics have the ability to either fill people with inspiration or bring them to tears. For this reason, the idea of artificially created lyrics is very intriguing. Can something without the capacity to feel write a song that could make an audience cry? Can an A.I. pen a verse that causes a listener to stop and think "wow, that was a good line!"? Could a tool be developed to offer suggestions to songwriters who find themselves stuck in the writing process? Questions like these are at the heart of why this project idea is interesting. Perhaps, however, the most important question to ask first is a simpler one: can an A.I. model create a set of lyrics that appear like they were written by a human?

Our group aims to explore this problem of artificial lyric generation. We will create a model that is trained on many sets of lyrics across many artists and multiple genres. The model will also compose lyrics, either from scratch or with other parts of a song as input. The goal is to create lyrics that rhyme, follow some sort of structure, and make grammatical and semantic sense. We chose this idea because all of our group members enjoy music, and Sean is actually a songwriter himself.

2 Approach

Our group plans to create a sequence-to-sequence model, trained with lyrics from hundreds of songs. Under the *seq2seq* method, the program will be able to create a line or verse, and then use it as input to generate the following line or verse. This will be especially helpful in terms of rhyme schemes, since the model will be able to detect where the rhymes are, so it knows what kind of metre to replicate. To detect rhyming words, the *Pronouncing* Python library will be used.

Several different models can be created to represent various genres and artists. This way, the program will be able to create rock lyrics that look different from its rap lyrics, which also differ from country lyrics, and so on. This also means the program can try its hand at mimicking specific artists.

3 Data

This model's dataset will consist of thousands of song lyrics, split by genre and artist. This data can be gathered using the *Genius* API and *Beautiful Soup*. By combining these tools, lyrics can be mass-imported from *Genius* to text files. To start, we will import the entire discography of 150 artists, split evenly across rock, country and rap. Artists will be chosen based on popularity and representation of eras and styles within genres.

4 Evaluation

For something as subjective as song lyrics, evaluation can be tricky, since there is no "right" answer. Despite this, there are several metrics that can be used to evaluate the output:

4.1 Grammar

The generated lyrics should make grammatical sense. Parts of speech, tenses and verb conjugations should all be used correctly. This can be tested using the *GrammarBot* API. Using this tool, a grammar score can be assigned. As a baseline, at least 50% of generated lines should have no grammatical errors, and at least 75% should have no more than one error. Ideally, semantics and word senses will also be used correctly. This is more difficult to measure automatically, however, and is somewhat subjective. For example, the line "the tree barks at the cat" might be using the wrong word sense of "barks," but it also might be making a clever pun or speaking in metaphor.

4.2 Rhyming

The output words should rhyme in the right places. This can be tested by feeding the seq2seq model a verse with a clear rhyme scheme, and evaluating the output verse to see if the rhyme scheme matches. It is worth noting that some rhymes are more important to a song's metre than others. Therefore, the baseline will be to correctly place rhymes at the end of lines. Ideally though, internal rhymes should also be considered. The baseline will only account for true rhymes, while future optimizations may include near rhymes.

4.3 Song Structure

Songs generally have several distinct sections – verses, a chorus, and perhaps a bridge and/or pre-chorus. Generally, the chorus appears several times, while each verse – though lyrically different – has similar structure. Structure is a fairly nebulous concept here; it may mean that verses share common rhyme schemes, or that they use the same number of lines. There are countless examples of songs that do not follow these rules, however, particularly in rap. Sometimes the similarity between verses is purely musical. Because of this, the baseline for this part of the evaluation will be that the generated lyrics contain some section that appears multiple times (the chorus) and at least two other sections (the verses). While it is true that not all songs follow this pattern either, it is common enough that we are comfortable placing this restriction.

4.4 Distinction Between Genres/Artists

There will be different models to represent different genres and artists, so these models should produce different results. There are various tools on the internet that attempt to classify lyrics based on genre. Using one of these tools, we can measure our model's success in this area. As a baseline, the "correct" classification in each set of outputs should achieve a plurality. That is to say, for a set of 20 generated rock songs, rock should be the most frequently predicted genre. Once this baseline has been reached, we can focus on differences between outputs for different artists. This will have to rely more on the eye (ear?) test, but we may be able to manually spot some patterns. Perhaps, for example, the word "love" will appear more in generated Beatles lyrics than it does for other artists.

5 Timeline and Work Breakdown

Our goal is to reach the baseline in at least two of our four listed categories before the milestone, and reach the other baselines and beyond by the due date. Data creation will be complete by the end of October, with the initial model built by November 13th.

We plan to hold weekly meetings, during which time we will assess our progress and code as together as a team with pair (triple?) programming. We also will assign individual tasks for members to complete before the next meeting. We plan to create the initial seq2seq model together as a group, since this task is very central to the project, and difficult to split up. Despite our focus on group work, however, we will each have individual responsibilities. Below is a rough outline of some of our task allocation, which is subject to change:

- **Sean –** Data creation, rhyme recognition.
- **Amanda –** Genre/artist differences, semantics.
- **Nora –** Grammar, song structure.

As a final task, we plan to write music to match one set of generated lyrics. We then will record the new song as a group.