Genre-Based Lyrics Generator using a RNN with Word Embedding

Neython Lec Streitz and Jordan Jung

I. Introduction

Semantic relationships between words is a common attribute in all languages. For English, many of these semantic relationships construct the definition and meaning of each word. Given that these relationships are applicable to almost every word in an extensive language, Natural Language Processing (NLP) tracks a unique area of focus on the principles of these semantic relationships. Word embedding is one particular subset of NLP that introduces semantics into its processes. This form of language processing assigns vectors to every word based on surrounding contents. For every vector, the numerical values assigned to each expression can be used as inputs in a language model. Such a language model can produce new content based on the qualities of the training set.

The premise of this project was to generate lyrics based on specific genres of music. Specifically, this generation should include some form of word embedding in order to derive semantic information from the lyrics and into the generation. The purpose of this work is to learn more about word embedding as a technique and get a view into the difficulties of understanding semantic information in NLP.

II. Related Works

There were two main contributing projects that assisted with generating the lyrics. There are two main projects to mention that are similar to our project. The first is work done by Sameerchand Pudaruth et al., that focuses on a semi-automatic lyrics generator powered by context-free grammar techniques. While they did not use the same word embeddings techniques that we used, the goal of the project is the same. Additionally, their method of analyzing the results was similar to ours, in that they also surveyed people about the lyrics they generated. Secondly, is work by Yihao Chen and Alexander Lerch that focuses on generating lyrics using Sequence Generative Adversarial networks (SeqGAN). They focus on preserving melodic information which was outside the scope of our own project.

III. Methodology

To meet the goal of this project, we built a Keras Sequential recurrent neural network that took short, tokenized sequences of lyrics as input and predicted the next word in the sequence as output.

Before feeding lyrics to our model, we pre-processes them by removing punctuation and converting all words to lower case, padding the end of sentences with "\n" as a standalone word, generating a corpus, a vocabulary, word indices, and 5-gram sequences of words corresponding with a next word. Further experimentation on different n-grams or combinations of multiple n-grams is required. The sequences and next words are then split into training and testing portions before being fitted by our model.

Specifically, our model employed an embedding layer, a bidirectional long short-term memory layer, a dropout layer, and a dense layer. Our embedding layer acts as the first layer and takes as input an indexed vocabulary and outputs word embeddings to feed into the LSTM layer of our model. Secondly, we chose a bidirectional LSTM as our RNN model because it performs better than unidirectional in advanced tasks like ours, being able to learn words based on the words that come before and after them. Our dropout layer is included to prevent overfitting of the data that can happen easily with powerful neural networks. Lastly, we have a dense layer to convert our output to probabilities and we chose 'softmax' as our activation function, as that was what was used in tutorials on text generation. Each genre model is fitted for 20 epochs and hovers around 96 percent accuracy by the 20th epoch.

The generation of lyrics happens by giving our model a seed sequence of words, chosen randomly from the total list of sequences in the genre. We've chosen to use a temperature based sampling function to generate the next word to print. Simply opting for the highest probability word might cause us to endlessly loop between certain words, or just print out exact lyrics. Instead our model samples with a temperature of 0.75, which was chosen by running tests on different temperatures and observing what temperature gives the best balance of realistic sentences with some diversity of lyrics. Continuing on, our model builds and prints each next word in the sentence. Because we preserved the newlines from the original lyrics, any word paired with a newline in the original lyrics does the same when chosen by our generator. We chose this method of creating lines since we were not able to implement the same with start and stop tags. After four newlines have appeared in the generation, we end the lyric generation.

In terms of the methods used for analyzing the performance of our data, we opted for human judgement. We took 4 generated lyrics and 2 real lyrics per each genre, randomized them and then had other people guess the genre of the lyric and guess whether the lyric was real or fake. Ideally, our model would create lyrics that would be indistinguishable from real lyrics as well as easily be of a certain genre.

IV. Results and Analysis

Based on the results of our human judgment analysis, our model generated a lyric that reflected its genre 71.875 percent of the time. Rock had the highest percentage correct with 81.25 percent correct, followed by country and pop tied at 75 percent, and ending with metal at 56.25 percent correct.

In terms of the comparison between real lyrics, our model performed worse with our generated lyrics being chosen as real 51 percent of the time. In particular, country was chosen 68 percent of the time, metal 56 percent, pop 43 percent, and rock 37.5 percent.

Likely, the biggest factor in the judgment of our models was the absence of any harmony or rhythm in the lyrics. The lyrics that were guessed as correct typically had some kind of 'accidental' rhyming and harmony to them. Our model performed better in creating a genre-specific lyric likely because it featured words that are very common in the genre. For instance, in 'Rock Generated 2' the lyrics mention 'rock 'n roll'. Metal was certainly the worst category for genre determination, partly due to the ambiguity of the lyrics and some human bias in not being familiar with metal as a genre.

Because our model moves to a new line only when the newline word is chosen, it was easier to determine whether a lyric was real of fake because the line would stop in an awkward place. This speaks again to the lack of musicality in our lyric generator. In 'Pop Generated 4' the fourth line ends with "im the one that you" and even though we told the human judges that they were excerpts of lyrics, the real lyrics did not have the cut-offs. Additionally, there were examples of lines that sounded very musical but were interrupted by a word that did not make sense contextually. For example, 'Pop Generated 2' starts with "hey hey hey I got a condo in" but then prints out "my sweet and sweet" throwing off what was a good starting line. With that said, there were some lyrics that actually sounded very good and if the model was able to continue printing lines until a good stopping point was found, they could easily pass as real lyrics.

V. Threats to Validity

From the perspective of the model itself, there are two main threats to validity of this project. Firstly, because the model is built on already existing songs, the generated lyrics reflect those inputted lyrics. Changing the lyrics, changes the results of the model entirely. Secondly, some of the generated lyrics almost completely match real song lyrics. One of the generated lyrics used in the human judgment analysis was word-for-word a real country song. This was a clear example of the limitations of the model, which is that with perfect accuracy our model would not be creating new songs, but simply correctly printing out the songs we inputted.

Other threats to validity are based on the human judgment analysis. As mentioned before, one of the generated lyrics was word-for-word a real country song. This meant that participants all guessed it as a 'real' lyric because it technically was. In that same vein, some of the generated lyrics included parts of real songs which the participants recognized, skewing their judgment of the category and whether it was real or generated. This points to the largest flaw in the evaluation which is human bias in evaluating lyrics. The genre guessing and real or fake determination is entirely based on what the people judging the lyrics know of the genre and of music.

VI. Conclusion

Ultimately, our model adequately printed out lyrics corresponding to a specific genre using word embedding techniques. Nonetheless, there are big limitations to our model and lyric generator. The biggest such limitation being the fact that a perfectly accurate model would in theory just print out the same lyrics that were inputted into it. In more musical terms, our model lacks any sense of creativity. The generated lyrics have some difference to them, but this is more due to the models lack of 100 percent accuracy and not to any intentional lyric generation feature. In addition, there is no musicality in the lyrics. That is, the lyric lines don't have rhythm to them, rhymes are usually unintentional and sporadic, and the lyrics content is scattered without a clear subject and does not make any sense oftentimes. With those limitations in mind, future work on the project is clear. For one, finding a way to stop the lines when it makes sense would go a long way in making realistic lyrics. While we attempted to add start and end tags to our lyrics, the model would simply catch itself in loops of start and stop tags and we couldn't find a way to bypass this. Additionally, adding part-of-speech tags to the lyrics would allow us to create somewhat grammatically correct lyrics, simply ensuring that the lines we were printing had the features of an actual sentence. However, sometimes music does not follow proper syntax and so some balance would have to be found. Lastly, finding a way to generate more original lyrics, either through more hyper-parameter optimization or other features would give our generator an edge over one that is simply attempting to recreate inputted lyrics. In essence, our dropout layer and use of sampling was not enough. All in all though, we believe that we were successful in our attempt to generate genre-specific lyrics with the use of word embedding. This project goes to show difficulty of song-writing/song-production and of recreating features of natural language.

VII. References

References

[1] S. Pudaruth, S. Amourdon and J. Anseline, "Automated generation of song lyrics using CFGs," 2014 Seventh International Conference on Contemporary Computing (IC3), 2014, pp. 613-616, doi: 10.1109/IC3.2014.6897243.

[2] Y. Chen and A. Lerch, "Melody-Conditioned Lyrics Generation with SeqGANs," 2020 IEEE International Symposium on Multimedia (ISM), 2020, pp. 189-196, doi: 10.1109/ISM.2020.00040.