Lyrics Generator

A text mining study using an LSTM and Markov Models

Amsterdam University College
Davide Convertino
[davide.convertino@outlook.com]
Claudio Creis Caveiro
[cloclicreis@gmail.com]

ABSTRACT

The goal of our research is to test if an LSTM model and a Markov model can produce lyrics similar to human-created ones. Our research is based on a large dataset of over 170000 bars and a diverse pool of artists. Following a dataset analysis with a word cluster and bar chart, we built the baseline model. We developed a simple LSTM model into a three-layered LSTM with 8 nodes in each layer. Subsequently, we developed the Markov model to output the most probable words next to each other. Furthermore, with the LSTM we filtered the Markov model-generated lines to make them take meaning. Moreover, we performed an ablation study by changing the depth and the batch size and epochs of the LSTM model. Lastly, we picked the 3 lyrics with the highest max_score per line and we made a survey by comparing them to human-created lyrics from artists not included in the dataset. The results of the survey showed that a percentage close to 60% guessed which lyrics were the ones generated by our models. Since the max_score drops significantly with larger datasets, we encourage the introduction of a theme to have better coherence throughout the whole text.

INTRODUCTION

As time passes by, language models have grown exponentially and made progress in poetry and lyrics generation. We have decided to test the limit of these models by making them generate rap lyrics. We combined an LSTM model and a Markov model through a filtering process in which generated lyrics are fitted into the next line if it is the most likely to occur (while being different from the original lyrics), adding coherence to the lyrics. To format the lyrics we count syllables and make rhymes e. Lastly, after conducting an ablation study we ran a Turing test through a survey by creating a Google Form and had participants choose what was the human-created one and the one generated from our model.

RELATED WORK

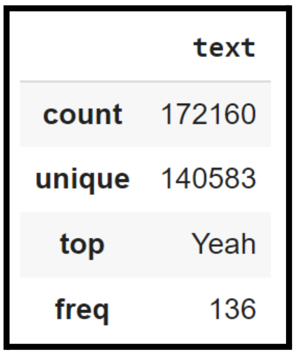
Early implementation of Recurrent Neural Networks (RNN) into lyric generation has come a long way. In the past 12 years, there have been numerous developments ranging from the creation of structured/rule-based generation to the application of statistical probabilities. An example of this is the algorithm of rhyme detection developed by Hirjee and Brown that analysed the sonoric pattern of words which we will be implementing during our work.

Equally, the introduction of Machine Learning models (ML) has opened many doors for the generation of poetry and lyrics. Papers such as Greene et al. used an n-gram language model complemented by a rhythmic model and Oliveira introduced a poetry generator which uses semantics and grammar templates in Portuguese. This has opened the field of lyric generation proceeding with works from Malmi et al. where their algorithm constructs a new poem with existing lyrics of rap songs done with neural embeddings of the lines created through character levels.

Additionally, the most similar works are the ones done by Peter Potash in which his project uses Long Short Term Memory (LSTM) and compares it to an n-gram model in lyrics generation accuracy. While we do not follow this comparison we do try to aim for high-quality output that is similar to actual lyrics and we test this on young adults.

DATASET DESCRIPTION

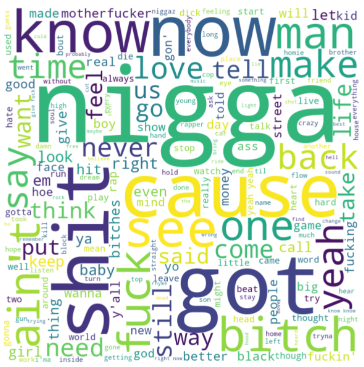
For our project, we used one dataset called Bars. The dataset was taken from Kaggle. However, we had to modify it from a CSV file to a txt file, since the input file in our model has to be a txt file with lyrics. We got rid of the artists' names and the empty lines in the dataset and converted it into a txt file. The final dataset contains over 170 000 lines of text of which 140 000 are unique (Figure 1), with each line being from a bar of a lyric from the most famous rappers, namely Eminem, Tupac, Notorious Big, Nas, Jay-z, and Kendrick Lamar.



	text
count	172160
unique	140583
top	Yeah
freq	136

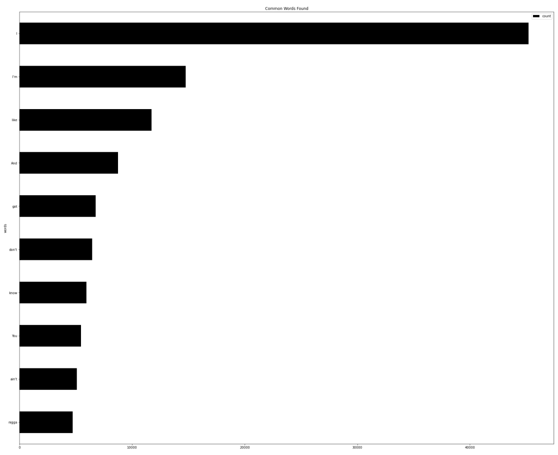
[Figure 1, dataset overview]

During our word analysis in the first part of our project, we created a word cluster by typecasting each value in the dataset to a string and then tokenizing the words. The most common words, as seen in Figure 2, highlight how most of the rappers in the US come from the same background.



[Figure 2, word cluster]

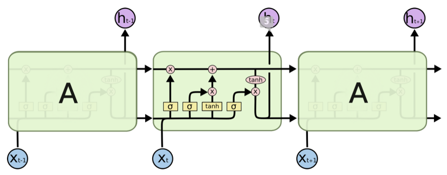
After that, we ran a bar chart with the 50 most common words in the dataset. Our finding was that most of the words have a frequency lower than 1000 with the exception of the top 4 words, which range between 1000 and 200 with the exception of the most used word being I which reaches almost 5000 and is 3 times more frequent than the second one (Figure 3).



[Figure 3, most frequent words bar chart]

LSTM MODEL

We started with the baseline model (simple LSTM), the dataset wasn't pre-processed and the output made no sense. We then tried to make a larger model using the sequential function adding neural networks such as the LSTM, with four nodes a 2 by 2 tensor, four layers with 8 nodes and a final layer with two nodes. We added the rmsprop optimizer for the gradient descent. We use the LSTM model to predict the properties of the next line of the poem with the number of syllables and the rhyme index as well as their similarity to the original lyrics so that only appropriate lines get selected. Accordingly, we decided to create a rhyme index where the model builds a list of rhymes by converting the most common rhyme ending into a float. The LSTM learns by creating cells with layers which extract or discard information creating long-term dependencies between words to give each sentence a logical meaning which will later filter the generated lyrics by the Markov model.



[figure 4, LSTM model structure]

MARKOV MODEL

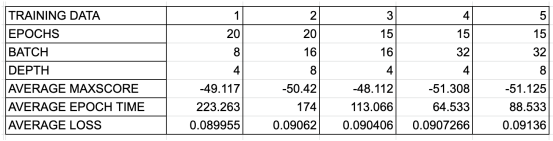
Then we added the Markov function which builds its own model of the text and is able to generate “random” lines of lyrics by calculating the probability of the next word through markovify. We then proceeded to implement a syllable counter that runs until the number of syllables is less than the max syllables which we set at 16 because the average rap line has 16 syllables in it. Subsequently, we create a generate lyrics function with the Markov chain function while specifying that the sentence created cannot overlap too much with the lyrics inputted (https://github.com/jsvine/markovify).

MODELLING AND RHYMES

We then proceeded to create our dataset for our 2 by 2 LSTM model with 2 arrays of data. After our model is trained we compose rap by creating a list of rap vectors and appending the syllables and the rhyme index into a list. Afterwards, the last words between the original and the generated lyrics, granting a penalty if they are the same then we generate a score by subtracting the sum of differences between the predicted and generated syllables as well as the difference between the predicted and generated rhymes. We verified that the syllables would not be over the maximum syllables.

ABLATION STUDY

To test the different outcomes of our LSTM model we decided to make an ablation study. We, therefore, modified three parameters: the depth, the batch size and the epochs trained. We made two tests with 20 epochs. We can compare them and see that with a lower batch size and a lower depth the average loss is the lowest out of all the trained models as well as the second-lowest average max-score. As the model training did not heavily influence the loss function after the 15th epoch we chose to lower it to the aforementioned amount. Afterwards, we tried modifying only one variable per test showing that the average loss is lowest with a lower depth. Therefore, shows a relationship between lowering the depth and better model learning outcomes. Additionally, the model trained the fastest with batch 32 and depth 4 but gave lower than average results. Additionally, we are able to conclude that the model itself does not fluctuate too much showing that our model is quite consistent. However, the lower the depth and batch size leads to better outcomes for our model.



TRAINING DATA	1	2	3	4	5
EPOCHS	20	20	15	15	15
BATCH	8	16	16	32	32
DEPTH	4	8	4	4	8
AVERAGE MAXSCORE	-49.117	-50.42	-48.112	-51.308	-51.125
AVERAGE EPOCH TIME	223.263	174	113.066	64.533	88.533
AVERAGE LOSS	0.089955	0.09062	0.090406	0.0907266	0.09136

[figure 5, Ablation study]

SURVEY FINDINGS

Lastly, we decided to pick the three best-performing models after the ablation study to observe which one the participants will have a difficult time distinguishing the lyrics generated from human-created ones. The first question contained the output produced by the model with depth 4, batch size 8 and number of epochs 20. The lyrics resulting from this model were compared to the lyrics of Wu Banga 101 by Ghostface Killah. Out of the 43 people who participated in the survey 56% were able to identify the model-generated lyrics. The second question contained the lyrics generated by the model with depth 8, batch size 16 and the same number of epochs as the previous model. The lyrics resulting from this model were compared to the lyrics of Mamacita by Outkast; 58% of participants guessed the model_generated lyrics. Lastly, the third question contained the lyrics generated by the model with depth 4, batch size 16 and 14 epochs. The lyrics resulting from this model were compared to the lyrics of Snake by Big Thymers; 54% of participants guessed the model-generated lyrics

The survey results are based on a low number of participants, 43, therefore a clear distinction between the three models is hard to justify. However, it appears that the second model was the one, which resembled the most human-generated lyrics. Furthermore, the survey findings highlight how with a low average max_score with a range between -45 to -50, only an average of 56% of the participants were able to identify the model-generated lyrics. For these findings to be accurate we would need a larger sample size, in order to have a scientific correlation between the max_score of the model and the prediction. However, this result points out how even a low max_score can produce lyrics that close to half of the sample size can distinguish.

CONCLUSION

The results indicate that the two models can be combined to generate rap lyrics that an average of 56% of participants can guess correctly, meaning that the LSTM and Markov models can generate lyrics similar to humans-created ones. However, the max_score of these lines changes drastically by a negative 50 points from a small dataset to a larger dataset. To further improve our research, a better connection between the two models needs to be implemented in order to have a higher max_score. Furthermore, the implementation of a theme in the input function could be added to our model in order to have a more realistic generation, since our model does not follow a specific theme. Lastly, we firmly believe that a significant increase in the max_score with a large trained dataset can build model-generated lyrics almost indistinguishable from human-generated ones.

RESOURCES

Cruys, T. V. de. (2020, May 24). *Automatic Poetry Generation from Prosaic Text*. GitHub. https://github.com/timvdc/poetry

Dolphin, R. (2021, March 26). *LSTM Networks | A Detailed Explanation*. Medium. https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9

Hirjee, H., & Brown, D. (2010). Using Automated Rhyme Detection to Characterise Rhyming Style in Rap Music. *Empirical Musicology Review*, *5*(4), 121–145. https://doi.org/10.18061/1811/48548

Markov Models for Text Analysis. (n.d.). Www.stat.purdue.edu. Retrieved June 2, 2022, from https://www.stat.purdue.edu/~mdw/CSOI/MarkovLab.html

OLIVEIRA, H. G., HERVÁS, R., DÍAZ, A., & GERVÁS, P. (2017). Multilingual extension and evaluation of a poetry generator. *Natural Language Engineering*, *23*(6), 929–967. https://doi.org/10.1017/s1351324917000171

Potash, P., Romanov, A., & Rumshisky, A. (2015). *GhostWriter: Using an LSTM for Automatic Rap Lyric Generation* (pp. 17–21). Association for Computational Linguistics. https://www.emnlp2015.org/proceedings/EMNLP/pdf/EMNLP221.pdf

Potash, P., Romanov, A., & Rumshisky, A. (2016). *Evaluating Creative Language Generation: The Case of Rap Lyric Ghostwriting*. https://arxiv.org/pdf/1612.03205.pdf

Ying, K. (2020). *Coding a bot to generate new lyrics using Markov Chains*. www.youtube.com. https://youtu.be/UtAgC6zCwME

Zhang, X., & Lapata, M. (2014). *Chinese Poetry Generation with Recurrent Neural Networks* (pp. 670–680). Association for Computational Linguistics. https://aclanthology.org/D14-10