Song Lyrics Generation

Nouran Ihab Sabry

March 14, 2024

1 Introduction

In the rapidly developing field of artificial intelligence and natural language processing (NLP), recent years have witnessed remarkable progress in text generation. Researchers and developers have utilized machine learning algorithms to create systems that can automatically produce coherent texts from various sources such as articles and books.

One emerging application of this technology is the generation of song lyrics. Multiple attempts have been made to transfer text generation methods to assist songwriters with their craft. However, researchers were met with various challenges when attempting to generate lyrics, as they vary depending on the genre of music and include special constraints such as word repetition, structural patterns, and line breaks.

2 Motivation

Even for the most talented songwriters, creating memorable lyrics has never been easy. For this reason, researchers have developed various applications of AI to the generation of song lyrics that could prove to be helpful to songwriters. Not only is an in-depth understanding of songwriting techniques and natural language processing (NLP) crucial, but the structural complexity of lyrics also increases the difficulty of this task. This paper discusses the recent approaches made in the field of these generative models for the generation of song lyrics.

The structure of the paper is as follows: Section III introduces algorithms and technqiues used in the field of song lyrics generation like Recurrent Neural Networks (RNNs), Long Short-Term Memory Models (LSTMs), etc. Section IV then discusses the Analysis of the project dataset and it's limitations.

3 Background and Related Work

3.1 Technical Background

3.1.1 Markov Chains

Markov chains or Markov processes are stochastic models describing a sequence of possible events in which the probability of each event depends only on the state which was reached in the previous one. [2] They are widely used as statistical models of real-world processes in fields such as animal population dynamics, airport lineups or lines, cruise control systems in cars, and currency exchange rates. [3]

3.1.2 Recurrent Neural Networks

A recurrent neural network (RNN) is a bi-directional neural network which allows the output from some nodes to affect subsequent input to the same nodes. Their ability to use internal state (memory) to process arbitrary sequences of inputs makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.[5]

3.1.3 Gated Recurrent Units

Gated recurrent units (GRUs) are an extension to RNNs, introduced in 2014 by Cho et al.[6]. GRUs are similar to long short-term memory (LSTM) models, as they contain a gating mechanism to input or forget certain features.

3.1.4 Long Short-Term Memory Models

One flaw of RNNs is that they have difficulty retaining context as the input sequence increase in length. This causes vanishing gradients, known as the Vanishing Gradient Problem, where the gradient shrinks during back-propagation to almost zero [9]. This causes the layers in the neural network to lose the ability to learn due to the very tiny changes to their weights cause by very small gradients.

This issue resulted in the development of Long short-term memory (LSTM) networks [10]. A unit of an LSTM consists of three gates: an input gate, an output gate, and a forget gate. The three gates control the information flow into and out of the cell. Forget gates use a value between 0 and 1 to indicate which information from a prior state should be discarded in relation to the present input. The values 0 and 1 are assigned to determine whether the information is retained or not. Input gates use the same mechanism to determine which new pieces of information will be stored.

3.1.5 Transformers

A transformer is a deep learning architecture developed by Google and based on the mechanism proposed by Vaswani et al.[16]. It requires less training time than RNNs and LSTM Moddels as it contains no recurrent units. Transformers have been used for training Large Language Models (LLMs).

Text is converted to tokens, which are numerical representations. Each token is converted into a vector using a word embedding table. A concurrent multi-head attention method is used at each layer to contextualize each token within the context window with other tokens. This amplifies the signal for key tokens and diminishes it for less important tokens.

3.1.6 Autoencoders

An autoencoder is a type of neural network used to reduce the dimensionality of data by learning two functions: An encoding function that transforms the input data, and a decoding function that recreates the input data from the encoded representation.

3.2 Related Work

3.2.1 Using Context-Free Grammars

Pudaruth et al.[1] proposed a semi-automatic lyrics generator which was implemented using Java and MySQL. Prior to implementation, statistical and grammatical analyses of three song themes (love, pain and cause) were carried out, with each theme containing 100 English songs. In order to generate lyrics, the user is required to input some information including: one of the three chosen themes, details such as the song title, author and potential singers, a verse which will be repeated several times, optional seed words in order to give more focus to the song, and the structure of the song with respect to chorus and verse. An example of a structure is verse-chorus-verse. The sentences in the lyrics were created using the n-sentence grammar, where a random combination of word bi-grams and tri-grams were used to form a verse of specific length. The verse length was randomly generated using information gathered from the grammatical analysis, which contained the frequency of each word type classified using it's PoS tag. For evaluation, 50 people were asked to fill out a form where they had to guess if lyrics were generated or existing from a list of 10 English lyrics. Two thirds of respondents were able to pick out written songs, and more than half (52%) of the respondents evaluated one of the generated lyrics as an existing song, which suggests that some of the generated lyrics were of human-written quality. However, the main limitation was that the generated lyrics often lacked semantic meaning although they were grammatically correct.

3.2.2 Using Markov Chains

Barbieri et al. [4] illustrate that constrained Markov processes can be used to generate texts that imitate a given style while adhering to structural properties. Their evaluation shows that constrained Markov processes generates better texts in terms of syntactic correctness and semantic relatedness than traditional Markov chains.

3.2.3 Using Recurrent Neural Networks

Fernandez et al.[7] continued research into generating song lyrics in the rap genre. They employed three character-level models: a plain recurrent neural network (PRNN), an LSTM, and a Gated Recurrent Unit (GRU) for the task and assessed their performance using three evaluation metrics: Sparse Categorical Cross Entropy, Rhyme Density, and a Turing Test. The models were suffered from poor Rhyme Density Scores, with none of the scores reaching higher than 0.15. For the Turing test, the performance of the models was also compared with the DeepBeat[8] model. DeepBeat's generated rap lyrics attained the highest score, being perceived as human-written by 71% of the total participants not familiar with rap lyrics, followed by the LSTM model, which deceived 67% and 53% of the participants whom are acquainted and unaccustomed to rap lyrics respectively. The RNN and GRU both deceived half of the participants on all tests.

3.2.4 Using Long Short-Term Memory Models

Potash et al.[11] present an LSTM model which successfully generates 'ghostwritten' lyrics in the rap genre that emulate the style of a specific songwriter. The proposed model does not require constraints to generate lyrics, and is demonstrated to produce novel lyrics that also reflect the rhyming style of the target artist. To assess the performance of the model, the results were compared with a basic n-gram model. Their models were evaluated evaluated by correlating the cosine similarity between the existing and generated lyrics using the "Inverse Document Frequency" algorithm, as well as computing the "Rhyme Density" score which was introduced by Hijree et al. [12]. It is defined as the total number of rhymed syllables over the total number of syllables.

Conversely, Gill et al.[13] explore music lyrics generation via LSTMs on multiple genres. In their approach, they focused on only six genres (Rock, Pop, Rap, Metal, Country, and Jazz) to conserve computational resources. They utilized a dictionary of words which were taken from the corpus of songs for a specific genre. The network consisted of three layers: an embedding layer, an LSTM layer with dropout, and lastly, a linear layer to output a vector of the length of the vocabulary with softmax as the chosen activation function. To analyze the resulting lyrics, cosine similarity was utilized to evaluate five chosen metrics in two ways. The first approach was between the metric vectors for original and generated songs for each genre, where the metrics calculated on the generated lyrics were most similar to the original lyrics in the rap genre, closely followed by pop and by rock. The second was between the metric vectors for original and generated songs by metric, which showed cosine similarity above 0.5 for each metric between the generated and original lyrics.

Angle et al.[14] propose a Bi-Directional LSTM Model to generate song choruses that emulates the style of a specific singer. They implement 12 models to emulate 12 different artists. All the models attained scores in a range from 77-88 percent for categorical accuracy. For evaluation, 5 participants were given 12 generated choruses, one from each selected artist, in a random order. No participant was given the same generated choruses as any other participant. The results show that the generated choruses were more than adequate at imitating the style of specific authors. Moreover, they were able to maintain enough coherency to be readable and convey meaning.

Moreover, Tee et al. [15] compared the performance of three methods for lyrics generation: Markov chains, LSTM, and GRU. The models generated lyrics for six music genres: rock, pop, country, hip-hop, electronic dance music (EDM), and rhythm and blues (R&B). Their results demonstrate that LSTM overall scored better in the average readability index, but the GRU model produced the overall highest Rhyme Density score (0.7482 for the pop genre). On the other hand, the Markov chains model scored the lowest due to the random nature of word retrieval from the dictionary.

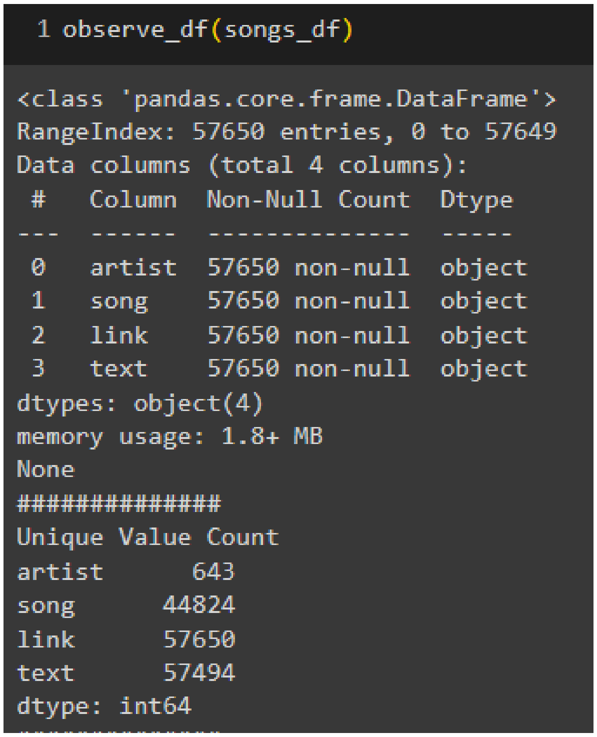
3.2.5 Using Transformers

Continuing with Lyrics generation in the rap genre, Nikolov et al. [17] proposed a transformer-based denoising autoencoder to generate rap verses conditioned on a list of content words. The resulting verses deceived participants 8% of the time with lyrics generated from news articles, and 25$ when lyrics were generated from random sources.

Additionally, Chang et al.[18] proposed a framework to generate the singable lyrics using a pre-trained transformer-2 (GPT-2) based framework for text style transfer. The model generates various lyric styles under specified conditions such as key sentences and genre style. In addition, the structure and rhythm of the lyrics are considered. The results demonstrated that the proposed method outperforms competitive baselines such as LSTMs.

4 Dataset Analysis

4.1 Analysis



```
observe_df(songs_df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 57650 entries, 0 to 57649
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   artist  57650 non-null  object
 1   song    57650 non-null  object
 2   link    57650 non-null  object
 3   text    57650 non-null  object
dtypes: object(4)
memory usage: 1.8+ MB
None
##############
Unique Value Count
artist      643
song      44824
link      57650
text      57494
dtype: int64
```

Figure 1: Overview of Dataset

4.1.1 General Overview

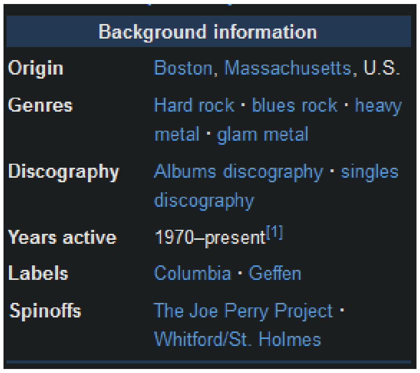
As seen in figure 1, the dataset consists of 57650 rows, with the following columns: artist, song, link, text. Seeing that the ”song” and ”link” columns are irrelevant to the task of generating lyrics, they were dropped. There are only 643 unique artists present in the dataset, and there are no null entries.

4.1.2 Web Scraping for the Genre

Since we require the artist genre which is missing from the dataset, web scraping was the chosen method to get the genre for each artist. The ”BeautifulSoup” Python library was used to scrape the data off of the Wikipedia website.

To simplify the data analysis as well as the generation tasks, only the first genre of the artist was considered, since the majority of artists/bands had multiple genres (Figure 2), which complicated the analysis process.

However, even further simplification was required, as there was a large number of unique genres such as ”Progressive Rock”, ”Europop”,”Alternative”, etc. Therefore, similarly to previous works in the literature, all unique genres were grouped under five main genres:



Background information	
Origin	Boston, Massachusetts, U.S.
Genres	Hard rock · blues rock · heavy metal · glam metal
Discography	Albums discography · singles discography
Years active	1970–present[1]
Labels	Columbia · Geffen
Spinoffs	The Joe Perry Project · Whitford/St. Holmes

Figure 2: Multiple Genres for the band ”Aerosmith”

1. **Pop**

2. **Rock:** further includes all sub-genres such as ”alt” and ”metal”.
3. **Country:** further includes all sub-genres such as ”folk” and ”celtic”
4. **Hip Hop:** further includes ”reggae”
5. **R&B:** further includes ”blues” and ”gospel”

4.1.3 Analysis Before Cleaning

With the genres grouped, the following observations were made: As seen in Table 1, the pop and rock genres make up approximately 80% of the dataset (Figure 3).

left

Genre	Occurences
Rock	23700
Pop	20524
Country	6605
Hip Hop	2428
R&B	1991

Table 1: Occurence count per genre.

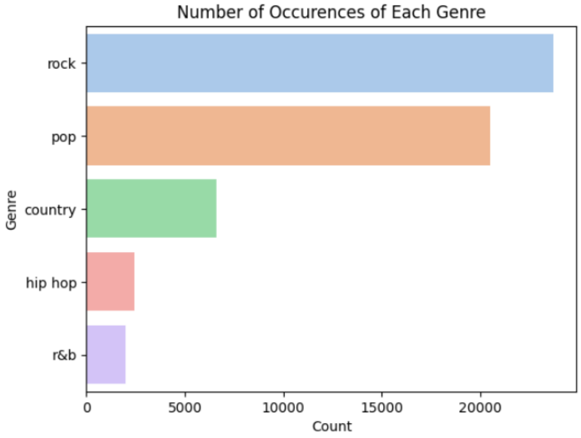
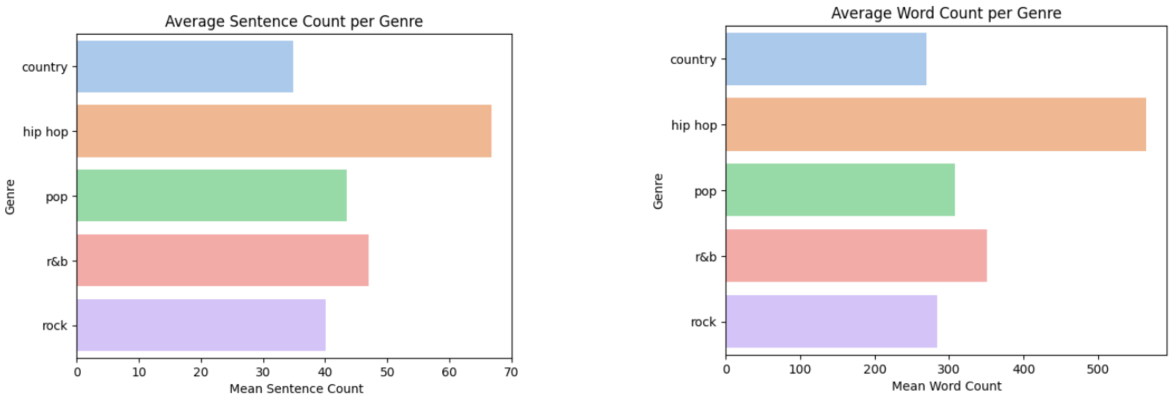


Figure 3: Genre Occurences in Dataset

Moreover, the average sentence count per genre was calculated, followed by the average word count. Both showcase the same trends, with hip hop having the longest sentences and largest word count (67, 564), followed by R&B (47, 351), Pop (43, 308), Rock (40, 284), and finally, Country (35, 269) (Figure 4).



(a) Mean Sentence Count per Genre

(b) Mean Word Count per Genre

Figure 4: Average Sentence Count (Left) and Average Word Count (Right)

4.1.4 Normalization

The dataset was cleaned by converting the lyrics to lowercase, tokenizing, then removing stop words and punctuation. Afterwards, lemmatization was applied onto the text instead of stemming, since stemming might remove parts of the word which are essential for proper context and placement in the sentence position.

4.2 Limitations

One critical limitation of this dataset is the bias towards the pop and rock genres, as seen in the Analysis section. This may cause a decline in model performance in the other less frequent genres.

References

[1] Sameerchand Pudaruth, Sandiana Amourdon, and Joey Anseline. Automated generation of song lyrics using cfgs. In *2014 Seventh International Conference on Contemporary Computing (IC3)*, pages 613–616. IEEE, 2014.

[2] Paul A Gagniuc. *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

[3] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[4] Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. Markov constraints for generating lyrics with style. In *Ecai*, volume 242, pages 115–120, 2012.

[5] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11), 2018.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[7] Aaron Carl T Fernandez, Ken Jon M Tarnate, and Madhavi Devaraj. Deep rapping: character level neural models for automated rap lyrics composition. *International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN*, pages 2278–3075, 2018.

[8]

[9] Sunitha Basodi, Chunyan Ji, Haiping Zhang, and Yi Pan. Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, 3(3):196–207, 2020.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] 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.

[12] Hussein Hirjee and Daniel Brown. Using automated rhyme detection to characterize rhyming style in rap music. 2010.

[13] Harrison Gill, Daniel Lee, and Nick Marwell. Deep learning in musical lyric generation: an lstm-based approach. *The Yale Undergraduate Research Journal*, 1(1):1, 2020.

[14] John Angle and Tyler Wengert. Generating artist styled song lyrics using a bi-directional long short-term memory neural network.

[15] Tze Huat Tee, Belicia Qiao Bei Yeap, Keng Hoon Gan, and Tien Ping Tan. Learning to automatically generating genre-specific song lyrics: A comparative study. In Boris Villazón-Terrazas, Fernando Ortiz-Rodriguez, Sanju Tiwari, Miguel-Angel Sicilia, and David Martín-Moncunill, editors, *Knowledge Graphs and Semantic Web*, pages 62–75, Cham, 2022. Springer International Publishing.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[17] Nikola I. Nikolov, Eric Malmi, Curtis G. Northcutt, and Loreto Parisi. Rapformer: Conditional rap lyrics generation with denoising autoencoders, 2020.

[18] Jia-Wei Chang, Jason C Hung, and Kuan-Cheng Lin. Singability-enhanced lyric generator with music style transfer. *Computer Communications*, 168:33–53, 2021.