**Natural Language Processing 6120: Lyrics Generator**

Krishna Venkatesh, hosahudyavenkatesh.s@northeastern.edu

Laasya Anantha Prasad, ananthaprasad.l@northeastern.edu

Shikha Tiwari,

**Abstract**

Lyrics generation necessitates that the model comprehends the context, meaning, and structure of the language. In this study, we compare various NLP models for generating song lyrics: Language N-Gram model, Long Short-Term Memory (LSTM), and Transformer.

The N-Gram Generator is a probabilistic algorithm that generates word sequences based on their co-occurrence frequencies in the training corpus. The LSTM is a deep learning RNN model well-suited for sequence generation tasks, while Transformers use an encoder-decoder structure with two feedforward networks to generate text.

To assess the models' performance, we train them on Ed Sheeran's song dataset and use metrics such as perplexity and BLEU score to measure their ability to predict the next word in the sequence. Additionally, we conduct a qualitative analysis of the generated lyrics to evaluate their creativity and coherence, focusing on rhyming schemes like "aabb" and "abab."

Overall, this study illustrates the potential of NLP techniques in lyric generation and emphasizes the superiority of deep learning models like RNN LSTM and Transformers for this task. The findings could be beneficial for developing lyric generation systems in the music industry, aiding songwriters and artists in creating lyrics more efficiently and effectively.

1.1 **Introduction**

Lyrics generation is a significant task in natural language processing (NLP) and has attracted considerable attention due to its potential applications in the music industry. The ability to generate creative and meaningful lyrics can be a valuable tool for songwriters and artists seeking to create new music content. However, generating lyrics is a complex task that requires understanding the language's context, meaning, and structure.

Creating song lyrics using a machine learning model has numerous potential applications in the music industry, from aiding songwriters in generating new ideas to providing inspiration for musicians experiencing writer's block. However, producing lyrics that not only sound good but also adhere to standard structure and rhyme patterns is a challenging task for AI models.

To address this challenge, our project aims to develop a text generation model capable of producing song lyrics that follow the essential components of a song, including an introduction, verse, pre-chorus, chorus, bridge, and outro. We plan to achieve this by employing a combination of NLP techniques, such as N-Gram models and RNN LSTMs, to capture the statistical patterns of language and generate sequences of words that are both grammatically correct and semantically coherent.

Additionally, we aim to incorporate the constraint of rhyming patterns into our model to generate musically appealing lyrics. Rhyming is crucial in songwriting as it gives a song its unique character and helps establish a sense of rhythm and flow. We plan to implement a rhyming algorithm to ensure that the end of each sentence or verse in the generated lyrics rhymes with the next verse, following "aabb" or "abab" rhyme sequences.

Overall, our project aims to provide songwriters and musicians with a tool that can generate high-quality and creative lyrics while adhering to the standard structure and rhyme patterns of a song. We believe this project can contribute to the fields of natural language processing and music composition, leading to exciting new possibilities for AI-generated music.

1.2 **Related Work and Background**

In our research on lyrics generation, we refer to the work by Mateusz Modrzejewski, Jakub Szachewicz, and Przemysław Rokita, titled "Lyrics Generation Using LSTM and RNN." This study explores the use of Long Short-Term Memory (LSTM) networks and traditional Recurrent Neural Networks (RNNs) for the task of automatic lyrics generation. LSTM networks, a specialized type of RNN, are designed to remember long-term dependencies and avoid issues such as the vanishing gradient problem. They are particularly effective for sequence generation tasks like lyrics generation, as they can maintain context over longer sequences. Traditional RNNs, although simpler, tend to struggle with maintaining context over long sequences due to the vanishing gradient problem.

The dataset used in their study typically includes a large corpus of song lyrics, which are preprocessed to remove noise, tokenize words, and structure the data into sequences suitable for model training. The models are trained on this preprocessed dataset using techniques such as backpropagation through time (BPTT). Evaluation metrics include BLEU (Bilingual Evaluation Understudy), ROUGE (Recall-Oriented Understudy for Gisting Evaluation), and human judgment to assess the coherence, creativity, and stylistic accuracy of the generated lyrics. The study addresses several challenges, including maintaining coherence, ensuring stylistic consistency, and preventing overfitting through regularization techniques like dropout and early stopping.

The findings from this study indicate that LSTM networks generally outperform traditional RNNs in generating coherent and contextually relevant lyrics. The generated lyrics are evaluated to be closer to human-like writing, with better handling of long-term dependencies. This foundational work by Modrzejewski et al. provides critical insights into the use of LSTM and RNN architectures for lyrics generation, highlighting the strengths of LSTM networks in maintaining lyrical coherence and stylistic consistency. These insights have significantly influenced our approach to developing a text generation model capable of producing high-quality, musically appealing song lyrics.

The paper "Deep Learning in Musical Lyric Generation: An LSTM-Based Approach" by Harrison Gill, Daniel Lee, and Nick Marwell, published in the Journal of Artificial Intelligence Research (JAIR) in 2021, investigates the capability of LSTM networks to generate genre-specific lyrics that maintain stylistic and contextual coherence. The LSTM model is chosen for its ability to handle long-term dependencies and sequence data effectively. The training data consists of lyrics from various genres, which are preprocessed and tokenized. The model is trained to predict the next word in a sequence given the previous words, learning the patterns and structures typical of each genre. The quality of the generated lyrics is evaluated using metrics such as average line length, word variation, and thematic consistency, with human evaluators also assessing the quality, coherence, and creativity of the generated lyrics. The findings demonstrate the effectiveness of LSTM networks in producing genre-specific lyrics that are both coherent and stylistically consistent.

While these studies provide valuable insights, more advanced research is necessary to generate high-quality lyrics that capture the nuances of human creativity. Better architectures and more sophisticated models are required to improve the diversity and contextual relevance of the generated lyrics. Our paper aims to close this gap in knowledge by comparing and contrasting three different architectures: Naïve Bayes Language N-Gram model, Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM), and Transformers. Through this comparison, we seek to identify the strengths and limitations of each approach, contributing to the development of more advanced and effective lyrics generation systems.

2 **Methodology**

**2.1 Dataset Used :**

The dataset used in this project is sourced from Kaggle and contains lyrics from Ed Sheeran's songs, provided in a CSV (Comma-Separated Values) format. This format ensures that the data is easily readable and accessible for analysis. The dataset consists of 179 rows and 3 columns, structured as follows:

1. **Index Column**: The first column represents the index of the entries.
2. **Title**: The second column contains the titles of Ed Sheeran's songs.
3. **Lyrics**: The third column includes the full lyrics of each song.

Each row in the dataset corresponds to a unique song by Ed Sheeran, with its title and lyrics clearly separated into distinct columns. This structured format facilitates efficient preprocessing and analysis, allowing us to train and evaluate various NLP models for lyrics generation. The comprehensive collection of lyrics provides a robust foundation for developing models that can generate new lyrics in Ed Sheeran's distinctive style.

**2.2 Data cleaning :**

For this project, the following text preprocessing techniques have been employed:

**2.2.1 Tokenization**: Split the lyrics into individual words or tokens.

**2.2.2 Lowercasing**: Convert all text to lowercase to ensure uniformity and reduce complexity.

**2.2.3 Punctuation Removal**: Remove punctuation marks to streamline the text.

**2.2.4 Stop Word Removal**: Eliminate common stop words that do not contribute significant meaning.

**2.2.5. Lemmatization**: Reduce words to their base or root forms to improve recognition of different word forms.

**2.2.6. Stemming**: Similar to lemmatization, but more aggressive in reducing words to their stems.

**2.2.7 Noise Removal**: Remove special characters, extra spaces, and other non-essential elements.

**2.2.8 Padding Sequences**: Pad sequences to a uniform length for consistency, particularly for RNNs and LSTMs.

These preprocessing steps will enhance the dataset's quality, making it more suitable for effective model training and generating coherent and meaningful lyrics.

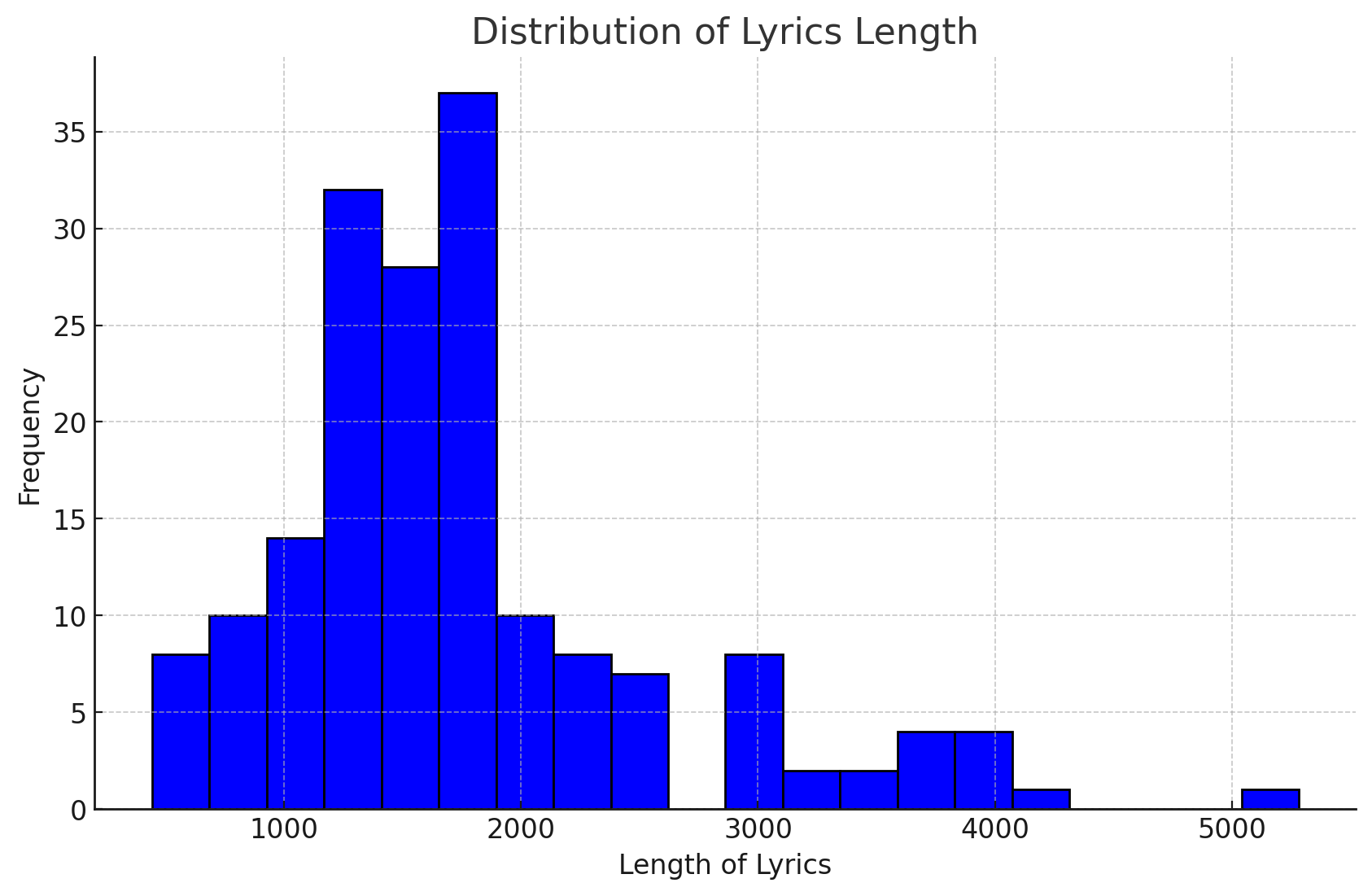


Figure 1 : This graph depicts the Distribution of Lyrics Length

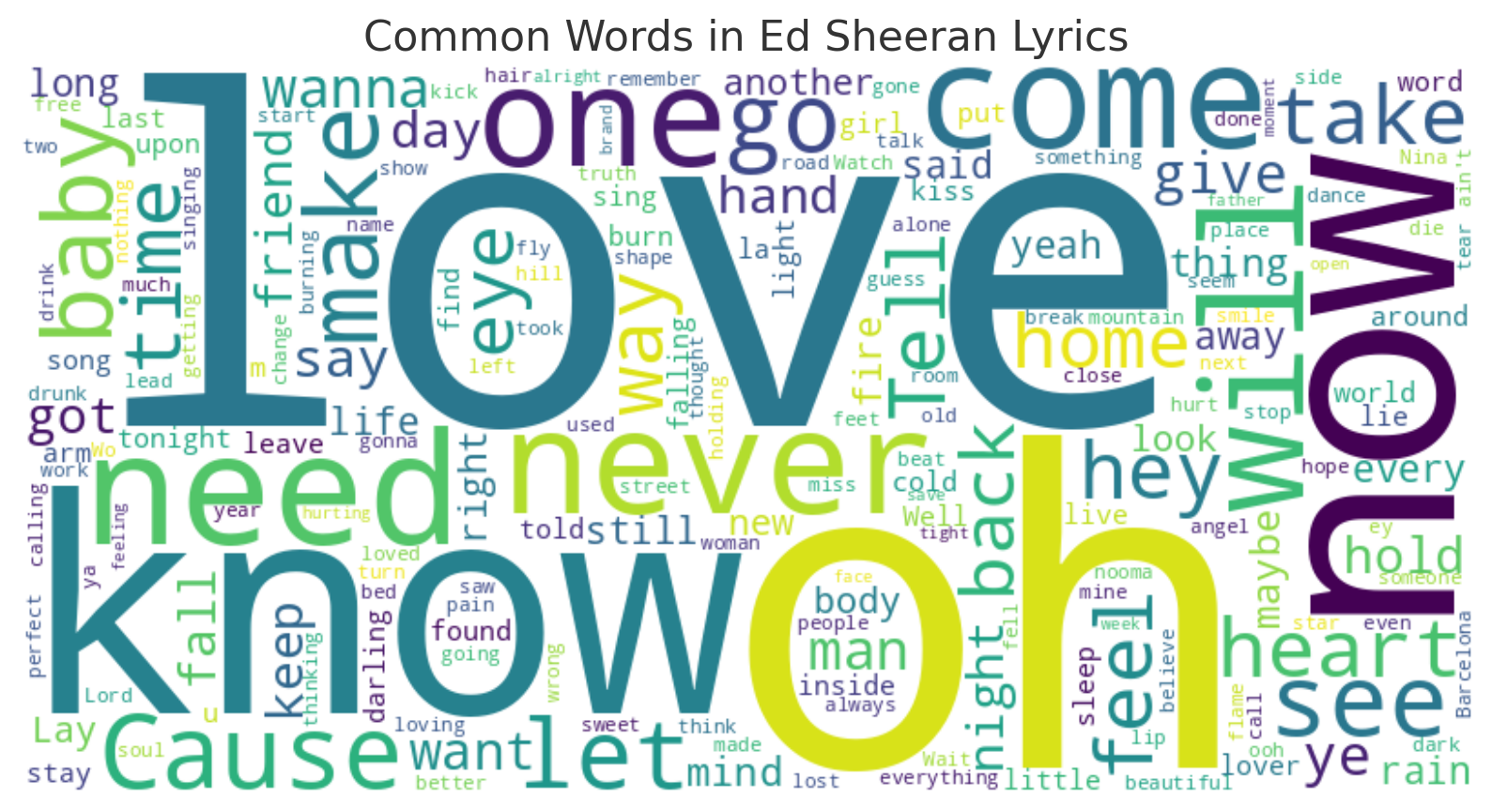


Figure 2 : This is a word cloud of the most common words in Ed Sheeran’s lyrics

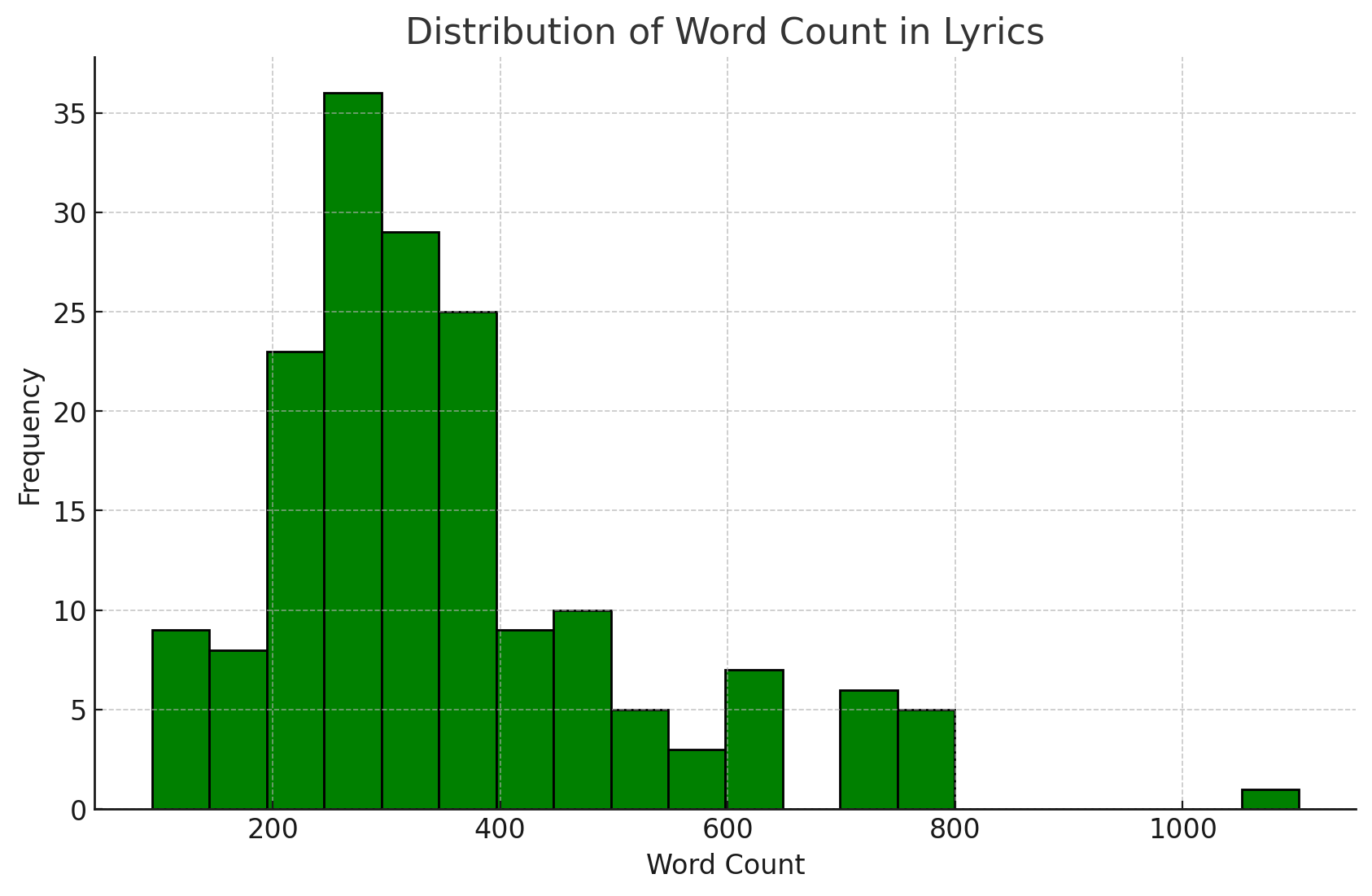


Figure 3 : This graph depicts the distribution of word count in Ed Sheeran's Lyrics

**2.3 Architecture**

2.3.1 **N-Gram Model**

The N-Gram model is one of the simplest and earliest language models used in natural language processing. It is a probabilistic model that predicts the next word in a sequence based on the occurrence of the previous N-1 words. The N in N-Gram represents the number of words considered for making the prediction. For instance, a bigram model (N=2) predicts the next word based on the previous word, while a trigram model (N=3) uses the previous two words.

The N-Gram model works by calculating the conditional probability of a word given the previous N-1 words. This probability is derived from the frequency of word sequences in the training corpus. Mathematically, the probability of a word sequence w1,w2,…,wnw1​,w2​,…,wn​ is approximated as:

P(w1,w2,…,wn)≈∏i=1nP(wi∣wi−(N−1),…,wi−1)P(w1​,w2​,…,wn​)≈∏i=1n​P(wi​∣wi−(N−1)​,…,wi−1​)

For example, in a trigram model, the probability would be:

P(wi∣wi−2,wi−1)=Count(wi−2,wi−1,wi)Count(wi−2,wi−1)P(wi​∣wi−2​,wi−1​)=Count(wi−2​,wi−1​)Count(wi−2​,wi−1​,wi​)​

The N-Gram model captures the local dependencies between words, making it useful for maintaining short-term context and producing sequences that resemble the training data. However, it has limitations in capturing long-term dependencies and syntactic structures due to its fixed window size. Despite these limitations, the N-Gram model can generate coherent phrases and short lines that reflect Ed Sheeran's lyrical style by training on his lyrics corpus.

2.3.1 LSTM

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) designed to address the limitations of traditional RNNs, such as the vanishing gradient problem. Introduced by Hochreiter and Schmidhuber in 1997, LSTMs can maintain long-term dependencies by using a gating mechanism to control the flow of information.

The LSTM architecture consists of a series of gates: the input gate, forget gate, and output gate, which regulate the addition and removal of information to the cell state. This structure allows LSTMs to retain relevant information over long sequences and discard unnecessary information. The key components of an LSTM cell are as follows:

1. **Forget Gate**: Decides what information to discard from the cell state. ft=σ(Wf⋅[ht−1,xt]+bf)ft​=σ(Wf​⋅[ht−1​,xt​]+bf​)
2. **Input Gate**: Decides which new information to add to the cell state. it=σ(Wi⋅[ht−1,xt]+bi)it​=σ(Wi​⋅[ht−1​,xt​]+bi​) C~t=tanh⁡(WC⋅[ht−1,xt]+bC)C~t​=tanh(WC​⋅[ht−1​,xt​]+bC​)
3. **Cell State Update**: Updates the cell state with new information. Ct=ft⋅Ct−1+it⋅C~tCt​=ft​⋅Ct−1​+it​⋅C~t​
4. **Output Gate**: Decides what information to output. ot=σ(Wo⋅[ht−1,xt]+bo)ot​=σ(Wo​⋅[ht−1​,xt​]+bo​)ht=ot⋅tanh⁡(Ct)ht​=ot​⋅tanh(Ct​)

2.3.2Trsansformer

The Transformer model, introduced by Vaswani et al. in the paper "Attention is All You Need" in 2017, revolutionized the field of natural language processing. Unlike traditional Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, which process data sequentially, Transformers leverage an attention mechanism that allows them to process all tokens in a sequence simultaneously. This architecture comprises an encoder-decoder structure, where both the encoder and decoder are built using layers of self-attention and feed-forward neural networks.

Transformers use self-attention to weigh the importance of different words in a sequence, enabling the model to focus on relevant parts of the input when generating each word in the output. This capability allows Transformers to capture long-range dependencies and contextual relationships more effectively than RNNs and LSTMs. The parallel processing capability of Transformers also significantly reduces training time and computational resources compared to sequential models.

In generating Ed Sheeran lyrics, the Transformer model offers several advantages. Its ability to handle long-range dependencies means it can understand and replicate the complex structure and thematic continuity of entire songs. The model's self-attention mechanism ensures that the generated lyrics maintain contextual relevance and coherence, producing lines that flow naturally and reflect the style of Ed Sheeran's music. Furthermore, Transformers can generate creative and diverse lyrics by attending to different parts of the input sequence simultaneously, leading to richer and more nuanced lyrical content. This makes the Transformer model an ideal choice for generating high-quality, stylistically consistent, and contextually rich Ed Sheeran lyrics.

**3. Approach and Implementation**

3.1 N-Gram

From the dataset, we observe that each sentence in the song is approximately 5-8 words long. After every 5-8 words, a new phrase or sentence typically appears in the song. Usually, the last word of each sentence rhymes with the last word of the next sentence. To distinguish these ending words, we append "/n" to the word. For example, consider the following sentences from the dataset: "Had to listen to all this drama/n" and "Me and drama always go on/n."

In these sentences, the words "drama/n" and "drama" will be treated as different words by the tokenizer. This method is used to separately capture ending words, as they tend to rhyme with the last word of the next sentence.

The perplexity has been calculated. Lower the perplexity, better the accuracy of the model. We can see that the perplexity is extremely high, indicating low accuracy. Laplace smoothing has been included for reducing the perplexity, but it remains high even after the smoothening.

We further define the N for the N-Gram model.

3.2 LSTM

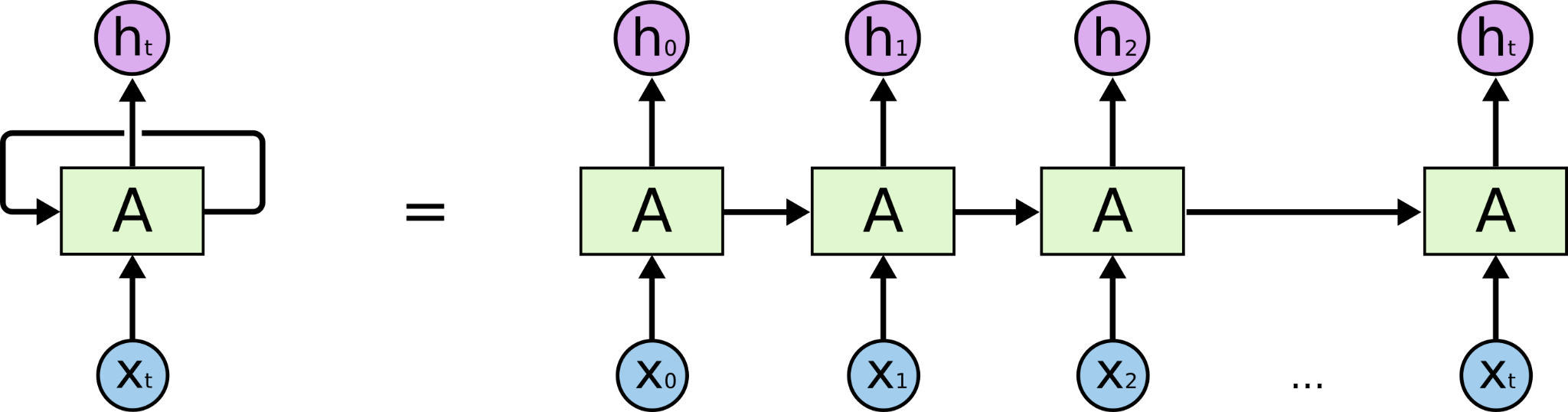


Figure # : LSTM Architecture

3.3 Transformer

**3. Results**

**4. Future Enhancements**

**5. Conclusion**

**References**

**[1]** Mateusz Modrzejewski, Jakub Szachewicz, and Przemysław Rokita, *"Lyrics Generation Using LSTM and RNN."*

**[2]**Harrison Gill, Daniel Lee, and Nick Marwell, *"Deep Learning in Musical Lyric Generation: An LSTM-Based Approach"*

**[3]**