# Deep Learning

## Final Project Report

**Project Title:**

**Song lyrics generation by genre using LSTM Approach**

**Submitters:**

Daniel David Glazman 318172848

Yuval Fisher 313598484

**April 2024**

University of Haifa

## Link to the Drive

## Introduction:

In recent years, deep learning techniques have revolutionized the field of natural language processing (NLP) and creative content generation. One application of these techniques is the generation of lyrics, a task that traditionally relies on human creativity and linguistic expertise. In this project we are implementing a Long Short-Term Memory (LSTM) network, a type of recurrent neural network, to generate lyrics specifically adapted to two distinct music genres: pop and rap.

The generation of lyrics poses several challenges, including maintaining coherence, and capturing the stylistic elements unique to each genre. By leveraging the sequential nature of LSTM networks and training them on large datasets of existing lyrics, we aimed to create a model capable of producing original and genre-appropriate lyrics.
The choice of LSTM networks is motivated due to their ability to capture long-term dependencies in sequential data, making them well-suited for tasks involving text generation.

## Motivation:

The rise of neural network-based language models has demonstrated promising capabilities in generating original long-form text from minimal initial one. Expanding on this base, recent work has used these models to take on the task of generating song lyrics. However, lyric generation presents unique challenges that differ from conventional text. For example, effectively modeling line breaks, capturing stylistic elements such as flow, rhyme, and repetition that are essential to lyrics, and understanding the structural frameworks.

Despite notable progress, existing approaches have yet to fully account for the diverse nature of lyrical content across different music genres. Lyrics, as a category, show significant variations influenced by genre-specific linguistic features. These features encompass diverse aspects such as line length, word repetition patterns, semantic nuances and etc. Our work aims to address this gap by implementing neural networks that generate genre-specific song lyrics while keeping the unique language features of each genre.

## Dataset:

The dataset comes from Kaggle. There are 5 million song lyrics – Link.
It's noted that approximately 33% of the dataset consists of rap songs, while 43% comprises pop songs, with the remaining data being of other genres, which are not relevant for the current analysis. The dataset is initially loaded in chunks to efficiently manage memory usage, with each chunk containing a specified number of records (set to 10,000 in this case).
We filter the dataset to by genre (Pop and Rap) and focus specifically on English songs.

After filtering, we combine the chunks of data into a single Data, which contains the lyrics and corresponding genre tags of English songs by the given genre.

We then further down sample the dataset by randomly sampling a specified number of rap and pop songs (5000 of each genre in this case) to ensure a balanced representation for analysis. This sampling process helps in managing computational resources and ensures that the dataset is appropriately sized for subsequent analysis.

## Preprocessing Approach:

In the beginning of the preprocessing, we cleaned the punctuation and parentheses from the lyrics dataset. That's because the frequency of those characters is higher comparing to other words, and in order to get a better result we had to remove them. In addition, we removed a bunch of words that we considered as "garbage words" (song instructions) that led to worse results, for example: "verse", "intro", "outro" and etc. After cleaning the text, we tokenized the lyrics for the embedding. We created a dictionary of words based on the lyrics sampled from the corpus of songs. The concept behind our neural network is straightforward: given a sequence of k words, the network predicts the next word, newline character, or punctuation in the lyrics. Subsequently, the network takes the preceding k-2 words, appends the newly predicted word, and feeds this concatenated sequence back into the model as input. After experimenting with different configurations, we settled on an input size of 10 words. This decision was the a of balancing computational resources and model complexity. While larger values of k tended to yield better models, they also increased computational demands, necessitating compromises in training dataset size, layer dimensions, or training epochs. At k = 10, our network began capturing many of the structural patterns present in lyrics. Although the results are not always perfect, this inputs size typically prevented repetitive lines and enabled coherent lyrical progression. Smaller values of k were prone to repetition and lacked the ability to maintain thematic consistency.

## Genre Feature:

In order to train a model that that gets as input the genre we had to add another feature for the data so the network will be able to generate a genre-based song. Each song in the dataset is associated with a genre label, either 'rap' or 'pop'. During the preprocessing stage, the genre information is encoded numerically, with 'rap' represented as 0 and 'pop' as 1. This encoding ensures that the model will be able to distinguish between the two genres during training process. These genre embeddings are then concatenated with word embeddings along the feature dimension, enabling the model to capture genre-specific patterns and relationships between words within the context of their associated genre. Subsequently, the concatenated embeddings are passed through an LSTM layer, allowing the model to learn dependencies between words over sequential input data while considering both word meanings and genre context.

## Network Architecture:

The `Word_LSTM` class represents a neural network architecture designed for word-level language modeling using LSTM cells.

1. **Embedding Layer:**
    - **Input:** Word indices representing the input sequence.
    - **Output:** Word embeddings of dimension (256).
    - This layer converts input word indices into dense word embeddings. Each word index is mapped to a high-dimensional vector representation, where similar words have embeddings closer in space.
2. **LSTM Layer:**
    - **Input:** Word embeddings concatenated with genre embeddings, resulting in a dimension of (256 + 128) for each word.
    - **Output:** LSTM hidden states of dimension (256).
    - The LSTM layer processes the word embeddings sequentially to capture temporal dependencies in the input sequence. It consists of two LSTM layers with 256 hidden units each. Dropout with a probability of 0.4 is applied to mitigate overfitting by excluding certain connections during training. It is implemented between the LSTM layers to regularize the network. The dropout technique prevents overreliance on specific patterns in the input data.
3. Fully Connected Layer:
    - **Input:** LSTM hidden state of dimension.
    - **Output:** Scores for predicting the next word in the vocabulary.
    - The fully connected layer takes the last hidden state of the LSTM sequence and maps it to a vector of size of unique words of the lyrics' corpus, where each element corresponds to the likelihood of each word in the vocabulary being the next word in the sequence. No activation function is applied to the output of this layer since it is followed by a softmax layer during training to obtain the probability distribution over the vocabulary.

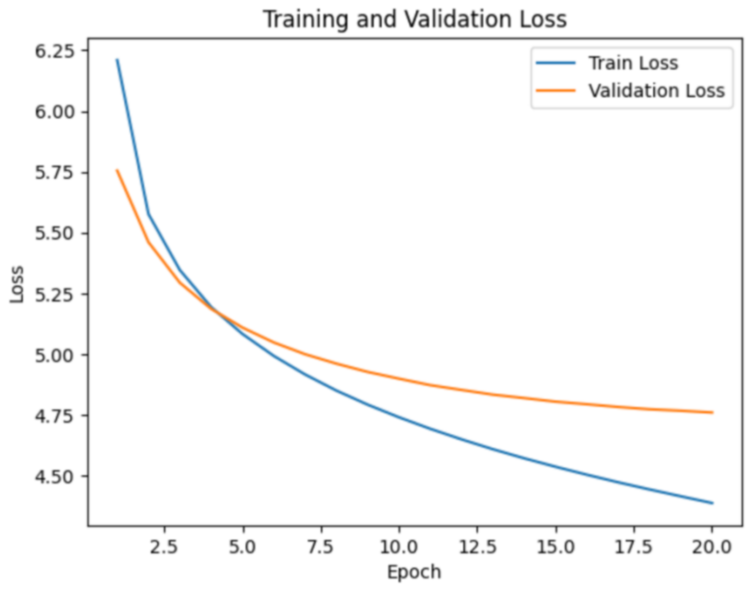
## Model Training:

- **Training Parameters:**
  - **Data Preparation:**
    The input data and corresponding labels are organized into a TensorDataset, which is then split into training and validation sets using a 70-30 split. This dataset is then loaded into a DataLoader for efficient batch processing, where each batch contains 2048 samples.
  - **Model Initialization:**
    Instantiates an instance of the model's class with specific parameters: Length of the vocabulary, dimensionality of the hidden state in the LSTM (set to 256) and the dimensionality of word embeddings (set to 256).
  - **Optimizer Selection:**
    Utilizes the Adam optimizer for training. It is configured with a learning rate of 0.002 to control the step size during optimization.
  - **Loss Function:**
    Utilizes the Cross-Entropy Loss as the loss function. Cross-Entropy Loss is commonly used for multi-class classification tasks, which fits the scenario of predicting the next word in a sequence.
  - **Epochs:**
    The model has been trained for 20 epochs.

## Validation Set:

- **Data Preparation:**
  Utilizing the `train_test_split` function from Scikit-learn, we divided the dataset into training and validation sets, allocating 70% for training and 30% for validation. This ensures an adequate sample for model evaluation while preventing overfitting.
- **Validation Process:**
  During training, we regularly checked how well the model performed on the validation set after each epoch. By making predictions on this separate set of data and calculating the average validation loss, we could understand how well the model generalized to new, unseen examples.
- **Result Monitoring:**
  Throughout the training process, both training loss and validation loss were tracked, providing insights into the model's learning dynamics and potential overfitting tendencies.

- **Training Results:**
  Training loss and Validation loss functions:



## Results and Evaluation:

In order to generate the songs, we used the method of seeding. Seeding provides an initial phrase for text generation, guiding the model towards a specific theme or style. It acts as a creative catalyst, blending the model's language generation with our own ideas to produce coherent and contextually relevant lyrics. For that matter, we used a first sentence of length of 10, that because the training was demanded a sequence (window) of length of 10.

Therefore, we loaded the model we trained, and we expect the user to enter the input genre and the seeding sentence, and the output will be the generated song that starts with the input sentence.

**Pop song example:**

i was a little child when she came to me
her eyes in her eyes
then she was the only one she had me

the only one i gave her
every morning in the mirror it's all in the air
in a room its time to go on and on

dear boy dont you know that i'm alone ?

here they go you just keep the feeling
you know that you got it all i need
just like that you're this my way
you're my life i'm a prick

feel my own dream
to see the tears in my eyes
i do a lot of words
see i'm not afraid for you
and if you just love me
i will love you unconditionally

i love the love for you
i love you i love you
i love you i love you
i love you baby

i love you more than i love you
and i do what i do
i'm in love with the heart that i can't hear
it feels like i can't stop what i can
i got my money on the new track
show a kick up and be a fool
that's the way you never feel
and then i just wanna be here with you

my lady might never stop still in the night

i wanna see you say the time
i want you into my life 'cause i hate to be
i said i was counting on my bed like the bar on the side

she asked me when he was
and we got a hand in the crowd
and that's the reason
to the judge
with a new light
tiny widow hallelujah
if i will
i don't care if i could be
but i'm so robin
if i've been waiting for you to see
and i was hanging on the distant beat
i seen my parents so give the cool

**Rap song example:**

i was a little child when she came to me
i said it's like that
you know when i'm gon get that bitch
i'm sick for that nigga yeah i want it to step
and that's right
come on and let me stay through
show me how i get top
check it yo it's a clique i got a suicide
but when my eyes go to sleep then i move a nigga
i wreck the new york nigga nigga i'll get the money
like it's so wrong
so i ain't never seen your baby
i'm always like this cursedass

here's me i'll just take a bitch a little bit

so take a bit of flowers
we can rely the streets bar at the store
so whoever already wreck the nigga who never had to flatten
hear the world that was enraged
but my girl was in the friend's i can't stand
it can see my life i will be a felony
i got this road i had sent to act so
especially i'm all about the shit nigga what the fuck

and i will not wishing what i am
and for what it's going to be so hot they said we got the same right here on the floor

i guess i never heard

if i was the big girl who would be me
who caught me on my side and i was looking
i got my own belief i got a friend for my friends
but i dont want to be
and you can't stand it again

you know what i mean you was real
you can't forget me but
you know why he would be through the shadows
i was a loser but it's been a patient and lyrics is leaded
hey hey ! we are the same boys for all y'all
the hate is a gangsta
that's what you know you got to get your balcony
and hold your hand and put your hands in your ass
put my lighters up the ice and erase my face
i get a hundred and i ain't caught it all too