Song Recommendation Systems: A Qualitative Study of Different NLP Approaches based on Lyrics Semantic Similarity

Alessandro Resta
alessandro.resta@studenti.unipd.it

Nicla Faccioli
nicla.faccioli@studenti.unipd.it

Lorenzo Perinello
lorenzo.perinello@studenti.unipd.it

Silvia Giro
silvia.giro@studenti.unipd.it

Ennio Italiano
ennio.italiano@studenti.unipd.it

***Abstract*— With the rapid growth of the internet, many songs and music are readily available for users on various platforms. The number, however, gets so huge that the user might get overwhelmed when it comes to selecting a follow-up song. A recommender system is handy in such situations, where users can choose a recommended piece based on their likes and dislikes. There can be various metrics in developing a song recommender system, lyrics being one of them. In this paper, we present three recommender systems based on TF-IDF (plus sentiment scores and topic modelling), fastText and Universal Sentence Encoder, to create embeddings out of the song lyrics, identify semantic similarity between them, and give the most similar songs to the user. The dataset was taken from Kaggle and contained 5 000 English songs after sampling. A qualitative study was then performed to evaluate the recommendations quality of the three systems on a set of 5 test lyrics, by asking volunteers to score the similarity perception of the recommended lyrics to the respective query lyrics. Results show that the best method was found to be the one based on TF-IDF, where embedding similarities best conform with human perception of similarity. Code and data are available at this link.**

I. Introduction

In today's digital age, music is more accessible than ever, thanks to the internet and rapid advancements in technology. However, the vast amount of available music can overwhelm users, making it challenging to decide what to play next. Recommender systems help by presenting a curated selection of songs from a vast collection, enhancing user satisfaction. This paper explores three song recommendation systems that aim at identifying semantic similarities between song lyrics, and suggest some recommendations given a starting lyrics (query). Various similarity measures facilitate the automatic evaluation of sentence similarity by focusing on the number and the size of the commonly occurring words. This approach is used in Speech Recognition, Machine Translation, Text Summarization, Paraphrase Identification, and Question-Answering. However, these measures often diverge from human perception, emphasizing "equality" over "resemblance." As a result, sentences with similar meanings but different wording can receive low similarity scores. Additionally, these measures often overlook word-relatedness, such as homophones or synonyms in lyrics, leading to misclassification if word equality is the only criterion. The recommendation systems introduced in this work leverage semantic similarity between song lyrics exploiting TF-IDF (plus sentiment scores and topic), fastText [1], and the Universal Sentence Encoder (USE) [2]. The dataset, sourced from Kaggle [3], originally comprises 25,742 songs with lyrics (subsequently sampled to 5,000). After each lyric is preprocessed, its embedding is created using one of the three methods above. We then calculate cosine similarities between these embedding vectors to establish a ranking and determine which lyrics are the most similar by their numerical cosine scores. To evaluate effectiveness, we conducted a qualitative study assessing human perception of the semantic similarity between query lyrics and recommendations.

The paper is organized as follows:

1. Section II provides background on similar song retrieval systems and underlying technologies.
2. Section III covers the extraction of lyrics from the dataset and their preprocessing.
3. Section IV describes the three embedding methods: TF-IDF, fastText, and USE.
4. Section V outlines the experimental setup and qualitative evaluation study.
5. Section VI concludes the paper and discusses potential future work.

II. Related Works

In recent years, both recommender systems (RC) and Natural Language Processing (NLP) techniques have gained popularity in the academic scenario and in industries, and they are applied in a vast range of application fields [4].

Recommender systems are one of the core features of audio streaming and media service providers; some of them, such as Spotify, Apple Music, and Pandora, rely heavily on different RCs [5].

Academic community has deeply investigated RCs, NLP, and their intersection: songs lyrics have been deeply analyzed in academic NLP-related scenarios, Mahedero et al. [6] explored the usage of language identification, thematic categorization, structure extraction, and similarity searches, demonstrating the usefulness of NLP techniques and their potential for improvement. Kento Watanabe et al. [7] focus on leveraging NLP techniques not only for analyzing but also for generating song lyrics. Their study explores various methods for understanding lyrical content, stylistic features, and emotional tone. Additionally, it delves into applications such as automated lyrics generation, music recommendation systems, and tools for aiding lyricists. Similarity is an aspect investigated by many. Chandra et al. [8] compute semantic similarity using different NLP techniques to categorize lyrics based on their semantic similarities, potentially offering new insights into genres and aiding in the development of more sophisticated music recommendation algorithms. Patra et al. [9] developed a method to enhance music recommendations by analyzing and comparing the semantic content of song lyrics using NLP techniques. This approach aims to suggest songs with similar lyrical themes and sentiments, improving personalization and user satisfaction in music recommendation systems. Kim et al. [10] investigate human perceptions of lyric similarity using six methods, including semantic, phonetic, and stylistic analyses, compared to human judgments. They find that models based on BERT embeddings [11], audio features, and phonetic components correlate well with human perceptions. Revathy et al. [12] utilize BERT to analyze and predict emotions and sentiments in lyrics, aiming to enhance music recommendation systems by understanding the emotional context of songs.

Our work differs from the ones mentioned above, as we designed three recommendation systems based on the song lyrics embeddings using the little to no explored techniques TF-IDF, fastText [1] and Universal Sentence Encoder [2] for this task. Then, we compared their performance by running a qualitative study on the human perception of semantic similarity between query lyrics and corresponding lyrics recommendations from the three systems. To our knowledge, no such experiment has been performed in the literature up to this date.

III. Dataset

The dataset, sourced from Kaggle [3], contains 25,742 English lyrics from famous singers, along with detailed information about their respective albums. The dataset is structured into three separate CSV files: `album_details.csv`, `lyrics.csv`, and `songs_details.csv`. For the purpose of this project, only the data from `lyrics.csv` were used. Each entry of this file includes, apart from the song's lyrics, the song name and the artist's name.

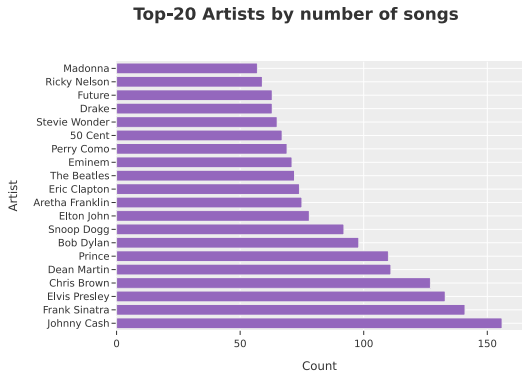

Figure 1: Top-20 artists by number of songs in the dataset.

A. Pre-processing

In order to speed up computation, a sample of the dataset was created by extracting 5,000 songs uniformly at random from the original set of 25,742 songs. This sampling ensures that the analysis remains efficient while still being representative of the overall dataset. After the sampling, several preprocessing steps are applied to ensure that data are clean and suitable for natural language processing techniques. In particular, the following steps are undertaken:

English contractions removal. This step expands contracted forms of English words (e.g., "we're", "y'all", "it's") into their full forms (e.g., "we are", "you all", "it is") in order to standardize the text and improve text processing.

Corpus lowercasing. Corpus lowercasing is a crucial step in text pre-processing where all the characters in the text are converted to lowercase. This step ensures uniformity in the text data by eliminating variations in capitalization.

Punctuation removal. Removing punctuation consists of eliminating all punctuation marks from the text. Punctuation marks include characters such as periods, commas, exclamation points, question marks and many other symbols.

Stop words removal. Frequent words (stop words) such as "and", "the", "in" that do not contribute significantly to the text are removed. This helps in reducing noise.

Tokenization. Tokenization is the process of splitting text into individual units referred to as tokens. In general, these tokens can be words, characters, or subwords. In this project, word-level tokenization is used.

Lemmatization. Lemmatization consists of reducing words to their base or root form, known as a lemma. This

process normalizes different forms of a word to a common base, enhancing the quality and consistency of text analysis.

Porter stemming. Stemming is the process of reducing words to their root form. The Porter Stemming algorithm removes common morphological and inflexional endings from words in English, resulting in the stem of the word.

Non-English song removal: Since the project focuses on English lyrics, any song containing parts in other languages is removed from the dataset to maintain a homogeneous language corpus.

Figure 2: Word cloud of the most frequent words.

B. *Split training and test set*

After the pre-processing steps are applied, the number of remaining songs is 4,741. Finally, the dataset is partitioned into two subsets: a training set and a test set. Specifically, the test set comprises five lyrics, while the remaining lyrics are allocated to the training set (4,736).

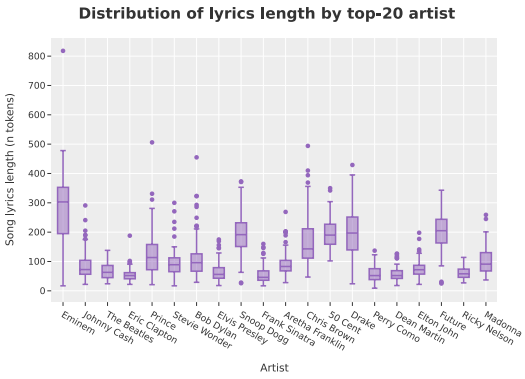

Figure 3: Lyrics length by top-20 artists in our dataset.

C. *Analysis of the dataset*

After completing the preprocessing steps, an analysis of the dataset was conducted, followed by the creation of several graphical representations to visualize the results. In the original dataset, songs from 150 different artists are included. Following the sampling process, we focused on the top 20 artists by the number of songs, as shown in Figure 1. This visualization highlights the distribution of songs among the most common artists in the dataset.

Additionally, an analysis of the length of the lyrics for these top 20 artists was performed. This analysis aimed to provide insights into the verbosity of each artist. The results of this analysis are depicted in Figure 3, offering a comparative view of the average song lengths across these artists.

Finally, part-of-speech (POS) tagging was performed to categorize the words according to their grammatical roles. Specifically, words have been divided into four different categories: nouns, verbs, adjectives and adverbs. A word cloud was also built to visually represent the most frequently occurring words in the dataset, emphasizing the key terms and prevalent themes (Figure 2).

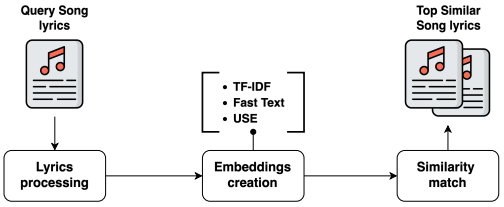

Figure 4: Workflow of the proposed system.

IV. Methods

The project aimed at developing three song recommendation systems based on song lyrics. The main idea is to start from the lyrics of a song, preprocess them according to the previous section, and create an embedding vector to numerically represent them. Figure 4 shows the overall workflow. The embeddings are created using one of the three methods: TF-IDF (short for sentiment, topic modelling and TF-IDF), fastText [1] and USE [2]. The following sections will describe them more in details.

A. *Sentiment Analysis, Topic Modeling and Vectorization (TF-IDF)*

The first and simplest method involved the extraction of sentiment scores, topic and words frequency from the song lyrics. Then, a concatenation of them was performed to build a unified numerical representation (embedding), to be used for similarity computation.

Valence Aware Dictionary and sEntiment Reasoner (VADER) [13] has been used to analyze the sentiment of each song's lyrics. It is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. The polarity scores - positive, negative, neutral and compound - have been calculated for each song; these scores were then stacked together to form the first set of features.

A Latent Dirichlet Allocation (LDA) [14] model has been trained on the processed lyrics to identify five topics. This algorithm operates on the premise that topics can be defined by their word usage, learning both how topics are distributed within a set of lyrics and how words are distributed across each topic. Each song's lyrics were then represented as a distribution over these topics, resulting in a topic vector, leading to the second set of features.

Finally, the lyrics were vectorized using the Term Frequency-Inverse Document Frequency (TF-IDF), converting the text into numerical vectors (third set of features) that represent the importance of each word in the lyrics, weighting down the most frequent and scaling up the rare ones.

Eventually, the three sets of features were concatenated into a single embedding vector for each song, and cosine similarity was used to measure the similarities between these vectors.

By integrating sentiment analysis, topic modeling, and vectorization, this method captures different aspects of the lyrics, enabling a multifaceted comparison between songs.

B. fastText

The second method involved the application of a pre-trained fastText [1] model to obtain a sentence embedding for each song lyrics. Since fastText works at the word-level, in order to build the embedding for the overall lyrics (which can be seen as a sentence) we first computed the word embedding for each word (token), then we averaged these embedding to form a unified vector representation. fastText embeddings are 300-dimensional vectors and were chosen for their ability to capture both the meaning of words and subword information, which is particularly useful for handling out-of-vocabulary words and morphological variations.

Analogously to the previous method, the similarity between two songs is determined by the cosine similarity of their fastText embeddings.

C. Universal Sentence Encoder

The third method consisted in applying a pre-trained version of the more advanced Universal Sentence Encoder (USE) [2] model directly to the lyrics to obtain a vector representation. This model was designed to create 512-dimensional vectors directly from the sentence, differently from fastText. USE provides robust and efficient embeddings that capture semantic nuances by leveraging a transformer-based architecture. Its ability to encode text into fixed-length vectors makes it suitable for handling variable-length lyrics while maintaining semantic consistency.

Again, the similarity between the two songs is determined by the cosine similarity of their USE embeddings.

V. Experiments

A. Experimental Setup

We structured our experiments with the goal of assessing the quality of the provided recommendations of our three systems for five test songs. Since no annotation was provided in the adopted dataset about the ground-truth similarity of any two lyrics, a quantitative evaluation of the methods was not possible, hence we setup a qualitative study.

In particular, we asked 38 volunteers to answer a simple questionnaire containing the lyrics of one of the five test songs, along with the "best" and (one) "bad" recommended lyrics retrieved from the training set by the three systems. We created five variants of the questionnaire, one for each test song, and distributed them randomly to the candidates to account for variation.

The best recommendation is the one achieving the highest cosine similarity score to the test song, while the bad recommendation was chosen to be the first reaching a value 0.4 lower than the best (enough to be considered bad). This approach is justified by the fact that we are interested in understanding whether lyrics embeddings close in the numerical space have their corresponding (original) lyrics perceived as similar by humans. Vice versa, we also want that far-away embeddings (bad recommendations) reflect a negative shift in the similarity perception. We find that this contrastive strategy in the evaluation, ideated by us, can better score our systems, and clearly define what we want to achieve when a given pair of lyrics is provided: high cosine similarity for a high human score and low cosine similarity for the low human score.

B. Evaluation Metrics

In order to run the qualitative study, we provided the candidates with specific guidelines, to be used as a means of comparison during the evaluation, as done in [15]. More in details, the key metrics are: **main theme/context** of the lyrics (e.g., war, deceit, meeting a new person), **message conveyed** (e.g., "Life is very short," "Enjoy your life"), **feelings/emotions** of the subject of the lyrics (e.g. disappointment, sadness, hope), **literal meaning** of the lyrics/vocabulary/setting (e.g. cosmos, sailing, life in the big city), **relationship** between sender and receiver of the lyrics (e.g. man singing to a girl), **language style** (e.g. teenage slang), **sociocultural context** of the song (e.g. author's biography/music genre, age of creation).

The scoring values also follow the work in [15]. They range from 1 to 5 and are to be intended as an overall scoring of a given recommendation based on the key metrics above. Here we report them for reference:

1. The lyrics are entirely dissimilar.
2. There is no semantic similarity, but the lyrics can be considered thematically related.
3. The lyrics resemble each other in the message, feelings of the protagonist/singer, lyrical situation, or literal meaning.
4. The lyrics share the same message and feelings but differ in lyrical situations and or literal meaning.
5. The lyrics share the same message, intentions, and lyrical situation, differing only in lexicon and genre.

C. Results

The results of the study for the three systems are shown in Figure 5, Figure 6 and Figure 7. The x-axis corresponds to the five queries, while the y-axis is the average score that candidates assigned for best (blu) and bad (red) recommendations. For all three systems, a pattern clearly emerges. The best recommendations always received the highest score in human similarity perception, often by a large margin. In addition, bad recommendations are always scored the lowest. This achievement tells us that all the three systems we designed perform as expected, meaning that embedding similarities conform reasonably well with human lyrics' semantic perception.

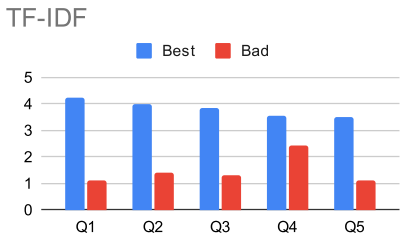

Figure 5: Average scores for TF-IDF.

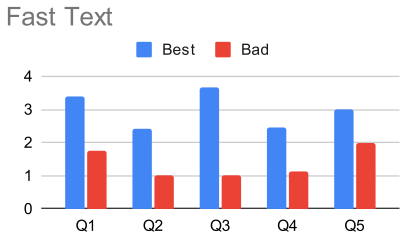

Figure 6: Average scores for fastText.

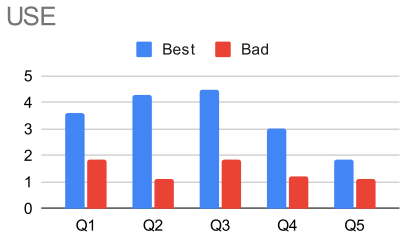

Figure 7: Average scores for USE.

To understand which model reached the peak performance, we can take the ratio between the (average) scores assigned to best and bad recommendations, that is:

- $\text{ratio} = \frac{\text{Avg. best scores}}{\text{Avg. bad scores}}$.

In order to have a value between zero and one, where zero means lowest performance and one best performance, we could apply the min-max normalization to the ratio:

- $\text{ratio}_{\text{norm}} = \frac{\text{ratio} - \text{ratio}_{\text{min}}}{\text{ratio}_{\text{max}} - \text{ratio}_{\text{min}}}$

The normalized ratios for all the test songs (Q1, Q2, Q3, Q4 and Q5) and methods (TF-IDF, fastText and USE), are summarized in Table 1, along with the average over the queries.

	Q1	Q2	Q3	Q4	Q5	Avg.
TF-IDF	**0.74**	0.54	0.56	0.26	**0.61**	**0.54**
fastText	0.36	0.46	**0.72**	0.42	0.27	0.45
USE	0.36	**0.74**	0.47	**0.47**	0.30	0.47

Table 1: Normalized score ratios.

From the table we can clearly see that the method based on sentiment scores, topic modeling (LDA) and TF-IDF (in the table just TF-IDF) has the highest avg. ratio. This means that, on average, it received the highest and lowest human scores for best and bad recommendations, respectively. We can conclude that it is the best method among the three for the task at hand, followed by the USE and fastText models.

This finding is pretty surprising, because it shows that advanced techniques like fastText and USE, failed in this specific task. We must say, though, that we used pre-trained models, and fine-tuning strategies might be the game changer. However, it would require a lot of annotated data that is usually expensive to collect.

VI. Conclusion

In this work we designed three song recommendation systems based on the semantic similarity of pairs of lyrics. The first approach explored the sentiment scores, topic modeling (LDA) and TF-IDF techniques to extract meaningful information from the preprocessed lyrics, and build a fixed-size vector representation. The second and third approaches made use of advanced embedding strategies based on fastText [1] and USE [2] to create such a vector. The similarity between two lyrics was said to be the cosine similarity of their embeddings, allowing us to build an ordered list of recommendations based on this metric for each queried song. After the models's creation, we ran a qualitative study involving 38 participants and asked them to rate, given a comprehensive set of guidelines, the best and (one) bad recommendations retrieved by each of the three systems. The analysis of the results showed an exciting finding, i.e. that the best system is the one denoted TF-IDF, which is also the simplest. This shows how advanced techniques are not always the best option, and that the best model depends on the application at hand (no one-fits-all).

In future works, we plan to reach more people to confirm (or deny) our findings, create a web application that enable users to use our systems more easily and, finally, try to fine-tune the fastText and USE models to seek for performance improvement over the simple strategy.

References

[1] "fastText." [Online]. Available: https://fasttext.cc/

[2] D. Cer *et al.*, "Universal Sentence Encoder." 2018.

[3] "Songs Lyrics - English Lyrics of Famous Singers." [Online]. Available: https://www.kaggle.com/datasets/terminate9298/songs-lyrics/data

[4] Ko *et al.*, "A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields," *Electronics*, vol. 11, no. 1, 2022, doi: 10.3390/electronics11010141.

[5] Baracskay *et al.*, "The diversity of music recommender systems," in *27th International Conference on Intelligent User Interfaces*, 2022, pp. 97–100.

[6] Mahedero *et al.*, "Natural language processing of lyrics," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, in MULTIMEDIA '05. Hilton, Singapore: Association for Computing Machinery, 2005, pp. 475–478. doi: 10.1145/1101149.1101255.

[7] K. Watanabe and M. Goto, "Lyrics Information Processing: Analysis, Generation, and Applications," in *NLP4MUSA*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:227217037

[8] Chandra, J., Santhanam, Akshay, Joseph, and Alwin, "Artificial Intelligence based Semantic Text Similarity for RAP Lyrics," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, 2020, pp. 1–5. doi: 10.1109/ic-ETITE47903.2020.211.

[9] Patra, Braja, Das, Dipankar, Bandyopadhyay, and Sivaji, "Retrieving Similar Lyrics for Music Recommendation System," 2017, p. .

[10] H. Kim and T. Akama, "A Computational Analysis of Lyric Similarity Perception." 2024.

[11] Devlin *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[12] V. R. Revathy, A. S. Pillai, and F. Daneshfar, "LyEmoBERT: Classification of lyrics' emotion and recommendation using a pre-trained model," *Procedia Computer Science*, vol. 218, pp. 1196–1208, 2023, doi: https://doi.org/10.1016/j.procs.2023.01.098.

[13] E. Gilbert and C. Hutto, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," in *Eighth International AAAI Conference on Weblogs and Social Media*, 2014. doi: 10.1609/icwsm.v8i1.14550.

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[15] A. Benito-Santos, A. Ghajari, P. Hernández, V. Fresno, S. Ros, and E. González-Blanco, "LyricSIM: A novel dataset and benchmark for similarity detection in Spanish song lyrics," *Proces. del Leng. Natural*, vol. 71, pp. 149–163, 2023, [Online]. Available: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/6550