# Music Recommendation through LLM Song Summary

Noah Tekle[1,*,†], Alline Ayala[2,†], Jonathan Haile[1,†], Abdulla Alshabanah[1,†], Corey Baker[1] and Murali Annavaram[1]

[1]University of Southern California (USC), 3740 McClintock Ave., Los Angeles, CA 90089
[2]Texas A&M University, 51015 W 34th St, College Station, TX 77840

**Abstract**

Recommendation systems play a key role in many aspects of life, from housing recommendations to music suggestions. As a result, recommendation systems have become an increasingly significant part of a company's digital business plan. Given the economic impact of music recommendations, research has suggested that using LLM-generated song summaries can result in better recommendation quality as opposed to using other textual features when designing music recommendation systems. This paper seeks to expand on this idea by examining what sort of textual features are helpful for music recommendations. In particular, we study the impact of 3 types of textual features derived from a song. The first option is to use public information about the song, such as the song name, to generate an input feature. In the second and third options, we prompt a LLM with the artist and song names, and with the song lyrics, respectively, to generate a song summary, which is then used as an input feature for the recommendation model. The fourth option we explore is to use part of the song lyrics as an input feature. The third and fourth options require parsing the lyrics of a song, which may be copyrighted. Our analysis suggests that while the context of the song, such as the song name, already provides improved recommendation performance, a more effective input feature would be to directly use the truncated song lyrics or at least use a summary of the song generated from a LLM as an input feature in cases of copyright barriers.
The Code can be found here: https://github.com/ntekle99/recsetters and a short preview of the paper can be found here: https://www.youtube.com/watch?v=DaW5po3AIh0

**Keywords**

Recommendation Systems, Large Language Model, Music Recommendation

## 1. Introduction

Music recommendations serve as the crux of audio streaming services, which is pivotal to extend the time a user engages on a platform [1]. As recommendation quality is key, many approaches have been attempted, to find methodologies to improve the recommendation quality to be as best and cost-efficient as possible. Methods such as combining collaborative and content filtering to develop a hybrid model [2], utilizing negative edges to predict when an interaction isn't happening [3], compressing hyper dimensions for space [4] or simply using CNN on embedding layers to store residuals to save future optimization [5], display the continuous attempts to improve recommendation systems everywhere. Many state of the art recommendation systems attempt to ameliorate recommendation quality by improving the quality of features used. Relevant works in this space includes analysing user skips to determine accurate user opinion on each individual song [6] or adding extremely personal identifiable information (PII) such as gender or age to the model [7]. More recently, features are being generated using large language models (LLM) to generate summaries songs [8]. This paper seeks to provide insights into which type of textual features is most effective for high-quality music recommendations by analyzing the impact of four approaches to incorporating textual features into a deep neural network (DNN) based recommendation model. The first approach utilizes the song name as a textual feature. This method is straightforward, avoids copyright issues, and relies on widely available metadata, offering a practical way to capture basic song attributes. The second approach enhances this by prompting a LLM to generate a song summary. This allows the model to inject additional context into the recommendation model, leveraging the LLM's pre-trained knowledge to enrich input features without requiring access to copyrighted data. The third approach builds on the second by providing more information to the LLM through prompting it with the lyrics rather than just the song and artist names. The fourth approach uses partial song lyrics a textual feature, which offer deeper insight into the song's theme. However, this approach must account for potential copyright restrictions. By exploring these approaches, we aim to identify the most effective textual features for improving recommendation quality while balancing practical and legal constraints.

## 2. Model Architecture and Variants

We use the popular Two Tower Neural Network (TTNN) as the base recommendation system on which we run our experiments and is proven to be very effective at large-scale recommendation [9, 10]. TTNNs are composed of two DNNs: the user tower and the item tower. The user tower transforms dense as well as sparse user features to produce a dense user representation $\mathbf{x_u}$ while the item tower does the same for items and produces a dense item representation $\mathbf{y_i}$. The final relevance score of item $i$ to user $u$ is computed with a scoring function $\mathbb{S} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, where $d$ is the dimensionality of $\mathbf{x_u}$ and $\mathbf{y_i}$. $\mathbb{S}$ can either be a simple inner product like a dot product or something more complex and learnable like another DNN. In practice, a variety of recommendation systems compute relevance between users and items as the inner product between user and item representations. We apply a pre-trained sentence transformer [11] to obtain a dense representations of the textual features before passing them to the item tower. Figure 1 shows an overview of the model architecture.

*The 1st Workshop on Risks, Opportunities, and Evaluation of Generative Models in Recommender Systems (ROEGEN@RecSys 2024), October 2024, Bari, Italy.*
*Corresponding author.
†These authors contributed equally to the research.
ntekle@usc.edu (N. Tekle); allineayala@tamu.edu (A. Ayala); hailej@usc.edu (J. Haile); aalshaba@usc.edu (A. Alshabanah); c.baker@usc.edu (C. Baker); annavara@usc.edu (M. Annavaram)

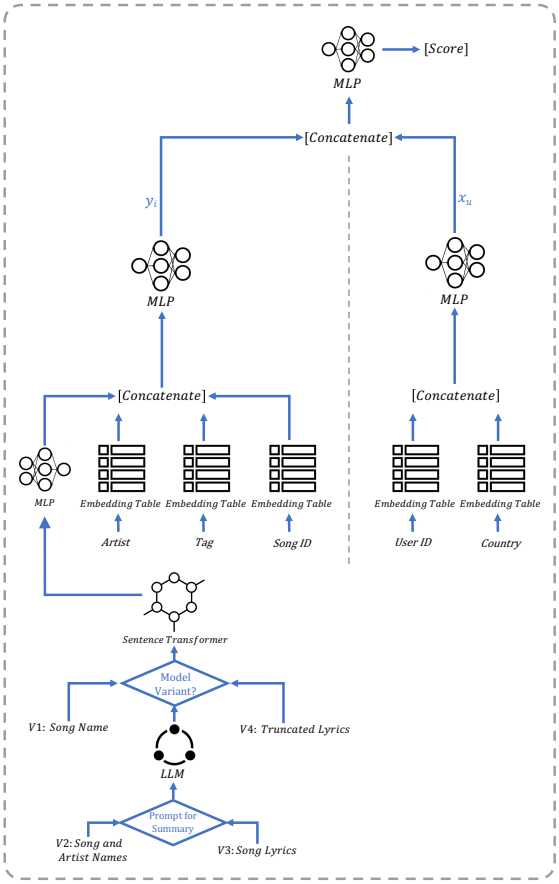


**Figure 1:** Overview of the model architecture.

### 2.1. Model Variants

We included four model variants in our study that differ only by the item textual feature passed to the sentence transformer. First, the 'Song Name' variant (V1), where we simply pass the song name to the sentence transformer. Second, the 'LLM Generated Song Summary (Song and Artist Names)' variant (V2) replaces the song name with an LLM generated song summary by prompting the LLM with the song and artist names. Third, the 'LLM Generated Song Summary (Song Lyrics)' variant (V3), where we prompt the LLM with the song lyrics and ask for the song summary. Fourth, the 'Truncated Lyrics' variant (V4), where we use a truncated version of the lyrics instead of the LLM generated summary. To ensure a fair comparison, remain cost-efficient, and isolate the cause of improvements, the truncated lyrics and the LLM generated song summaries are limited to twenty-five words, ensuring that any observed improvement is due to the feature itself rather than the word count. Additionally, we include results for a collaborative filtering model (CF) that uses only historical interaction data, excluding all user and item features.

## 3. Experiments

### 3.1. Datasets

For this study, we created the user data using the Last.fm API [12], consisting of user names, countries [13], and their liked tracks. Each liked track includes the date/time it was liked and the artist. The items data has two versions: one containing the 10,000 most liked Apple Music songs [14], and the other containing over 100,000 random songs hosted on Spotify [15], allowing us to test our model on datasets with different distributions. Through the Last.fm API, we incorporated tags into our items dataset to enhance our recommendation model, as tags have consistently improved recommendation quality [16]. To create the summaries, we used the Groq API [17] to run the llama3-8b-8192 LLM on their hardware infrastructure for faster compute times [18]. Concerning generating the lyrics, the Deezer API was used to leverage their private dataset of songs[19]. Our interaction data, which is processed using implicit feedback setting, was created by mapping each user to their liked tracks. To ensure a more meaningful collaborative signal in the dataset, we filtered out 1) any song that had fewer than five user and 2) any user who interacted with fewer than five songs. Although other features such as user age or gender could likely increase the model's accuracy, we decided against including them to follow best privacy practices [20]. The statistics of the datasets used in our experiments are reported in Table 1

| | Apple Dataset | Spotify Dataset |
|---|---|---|
| #Users | 8,000 | 8,000 |
| #Items | 6,800 | 3,630 |
| #Feedback | 114,000 | 71,690 |
| Sparsity | 99.753% | 99.790% |

**Table 1**
Summary of datasets and content information for Apple and Spotify.

### 3.2. Evaluation Criteria

In our exploratory study, we use two metrics: Hit Rate at k (HR@k) and Normalized Discounted Cumulative Gain at k (NDCG@k). HR@k evaluates whether a candidate item is included within the top k recommendations. NDCG@k, on the other hand, assesses the position of the recommended item within the ranked list, giving higher scores to items appearing closer to the top. By combining these metrics, we can comprehensively measure both the accuracy and relevance of our recommendation system.

### 3.3. Results and Discussion

In this section, we present the results of our exploratory study in Table 2 and Table 3 and report our findings. We also include samples of two songs and their corresponding textual features in Figure 2.

#### 3.3.1. Main results

We note the following from the main results reported in Table 2:

- While analyzing the HR for the variants included Table 2, it becomes apparent that the best-performing variant is the truncated lyrics model (V4) for both datasets, as it results in the highest ranking performance. The second-best is the LLM summary generated from the song and artist name (V2), highlighting the value of using a song summary, even if it was solely generated based on the LLM's knowledge.

| | Apple dataset | | Spotify dataset | |
|---|---|---|---|---|
| | **HR @10** | **NDCG @10** | **HR @10** | **NDCG @10** |
| CF | 0.030333 | 0.014603 | 0.075819 | 0.029261 |
| V1: Song Name | 0.031706 | 0.014655 | 0.088115 | 0.035020 |
| V2: LLM Generated Song Summary (Song and Artist Names) | 0.062215 | 0.029037 | 0.121396 | 0.060251 |
| V4: Truncated Lyrics | 0.176677 | 0.085487 | 0.156297 | 0.069138 |

**Table 2**
Performance comparison of different recommendation methods across Apple and Spotify datasets.

These results are consistent with those observed for NDCG as well.

- One potential reason that the truncated lyrics (V4) performed best is that they might have provided the context the model needs from the beginning section of the lyrics. Even if the rest of the song offered more context, the initial portion may have been sufficient for the model to make effective predictions.
- We also note that the song name variant (V1) performs slightly better than collaborative filtering, suggesting the need for incorporating textual features.
- In the absence of original lyrics, such as due to copyright barriers, LLM generated song summary based on song and artist names (V2) remain a viable option, as they improve recommendation performance compared to a collaborative filtering model or using just the song name with other features.

Due to constraints in computing resources, we were unable to obtain the results for the LLM generated song summary based on the song lyrics (V3) for the Spotify dataset. Therefore, it is excluded from the main results; however, it is included in the sensitivity analysis in Section 3.3.2 for the Apple dataset.

#### 3.3.2. Song summary sensitivity analysis

In this section, we investigate how sensitive the recommendation model is to the song summary by examining the results of the LLM generated song summary variants, V2 and V3. We conduct four experiments: 1) 'V2.1: LLM Generated Song Summary (Song and Artist Names, 25 Words),' where we prompt the LLM using the artist name and song name and limit the response to 25 words; 2) 'V2.2: LLM Generated Song Summary (Song and Artist Names, No Word Limit),' where we prompt the LLM using the artist name and song name without any word limit ; 3) 'V3.1: LLM Generated Song Summary (Song Lyrics, 25 Words),' where we prompt the LLM using the lyrics and request a 25-word summary; and 4) 'V3.2: LLM Generated Song Summary (Song Lyrics, No Word Limit),' where we prompt the LLM using the lyrics without any word limit. We report the results in Table 3 and note the following:

- The LLM summary generated using song lyrics, as opposed to the song name, unequivocally performs better. The LLM generated summary (V2.1) has a HR of 0.0622, while the LLM summary generated using song lyrics (V3.1) has a HR of 0.136, making a strong case for the superiority of forming LLM summaries through lyrics. This finding could also pave the way for new research into assessing LLM knowledge to determine if there is any bias or repeating pattern in how summaries are generated.
- The word count limit affects the quality of the LLM summary and, consequently, the recommendation performance, as the performance of the 'no word limit' experiments (V2.2 and V3.2) is better than the '25 words' ones (V2.1 and V3.1).

### 3.4. Hyper-parameters

We use a batch size of 1024, a learning rate of 0.00002 and an embedding table dimension of 96 for all models. We employ mixed negative sampling, both in batch and out of batch, and we sample 20 negative edges for each positive edge during training. All MLPs use the GeLU activation function [21] and layernorms between layers. MLP dimensions can be found in our code, which we are making public and can be accessed using this link: https://github.com/ntekle99/recsetters

## 4. Conclusion and Future Directions

Overall, we presented an exploratory study to investigate the importance of using song context in a music recommendation model. While solely utilizing lyrics provides the best recommendation quality, our research indicates that LLM summaries are a viable option when copyright restrictions prevent the use of full lyrics. Combining lyrics with other methods, such as analyzing skipping behavior [6] or demographic profiling [7], could lead to a model with superior ranking performance, advancing the development of more effective music recommenders. Future research could examine how different LLM summary lengths impact recommendation quality, identifying the optimal word count, as adding more words may reduce benefits or even cause hallucinations. Additionally, exploring various LLM prompts, such as related songs, could help identify alternatives to lyrics. The use of multi-modal LLMs, different model architectures, and varying LLM sizes might further improve the hit rate for lyrics based LLM summaries. Finally, further investigation is needed into potential biases in LLM generated summaries, particularly whether they perform worse for non-English languages.

## ACKNOWLEDGEMENTS

We sincerely thank all the reviewers for their time and constructive comments. This material is based upon work supported by Defense Advanced Research Projects Agency (DARPA) under Contract Nos. HR001120C0088, NSF award number 2224319, REAL@USC-Meta center, and VMware gift. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

| | Apple dataset | |
|---|---|---|
| | **HR @10** | **NDCG @10** |
| V2.1: LLM Generated Song Summary (Song and Artist Names, 25 Words) | 0.062215 | 0.029037 |
| V2.2: LLM Generated Song Summary (Song and Artist Names, No Word Limit) | 0.064616 | 0.031359 |
| V3.1: LLM Generated Song Summary (Song Lyrics, 25 Words) | 0.13615 | 0.062605 |
| V3.2: LLM Generated Song Summary (Song Lyrics, No Word Limit) | 0.137559 | 0.063157 |

**Table 3**
Performance comparison of Summaries for the Apple dataset.

**Song name:** Fire

**LLM summary:** "Fire" is an upbeat pop-rock anthem that speaks of a desire to break free from the constraints of a relationship that's no longer meeting one's needs. The lyrics convey a sense of longing and frustration, with the narrator yearning for something more passionate and fulfilling. The song features DeGraw's soulful vocals and a catchy, energetic melody.

**Lyrical summary:** A passionate and introspective ballad about the fiery intensity of a love that ignites the soul, consuming everything in its path, leaving scars and memories.

**Lyrics:** Oh if there's one thing to be taught
It's dreams are made to be caught
And friends can never be bought ...
Oh we on fire
We on fire
Dit dit heart and soul
Hey and nothing's going to be the same
Hey the life that you made will not be today.

**Song name:** Why Go

**LLM summary:** "Why Go" by Pearl Jam is a song about the struggles of a toxic relationship. The lyrics describe the emotional pain and turmoil caused by a partner's manipulation and control. The song's narrator is trying to break free from the relationship but feels trapped and unsure of how to escape. The song features Eddie Vedder's powerful vocals and a soaring guitar riff, creating a sense of urgency and desperation.

**Lyrical summary:** A nostalgic reflection on the passing of time, longing for a love that has slipped away, and the bittersweet memory of what could have been.

**Lyrics:** She scratches a letter
Into a wall
Made of stone
Maybe someday another child
Won't feel as alone as she does
She's been diagnosed …
(Why go home?)
(Why go home?)

**Figure 2:** Song Samples.

## References


[1] H. Ko, S. Lee, Y. Park, A. Choi, A survey of recommendation systems: recommendation models, techniques, and application fields, Electronics 11 (2022) 141.

[2] K. Yoshii, M. Goto, K. Komatani, T. Ogata, H. G. Okuno, An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model, IEEE Transactions on Audio, Speech, and Language Processing 16 (2008) 435–447. doi:10.1109/TASL.2007.911503.

[3] P. Seshadri, P. Knees, Leveraging negative signals with self-attention for sequential music recommendation, arXiv preprint arXiv:2309.11623 (2023).

[4] J. Morris, M. Imani, S. Bosch, A. Thomas, H. Shu, T. Rosing, Comphd: Efficient hyperdimensional computing using model compression, in: 2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), IEEE, 2019, pp. 1–6.

[5] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, X. He, A simple convolutional generative network for next item recommendation, in: Proceedings of the twelfth ACM international conference on web search and data mining, 2019, pp. 582–590.

[6] E. Pampalk, T. Pohle, G. Widmer, Dynamic playlist generation based on skipping behavior., in: ISMIR, volume 5, 2005, pp. 634–637.

[7] G. Vigliensoni, I. Fujinaga, Automatic music recommendation systems: Do demographic, profiling, and contextual features improve their performance?., in: ISMIR, 2016, pp. 94–100.

[8] A. M. Taief, Application of LLMs and Embeddings in Music Recommendation Systems, Master's thesis, UiT Norges arktiske universitet, 2024.

[9] X. Yi, J. Yang, L. Hong, D. Z. Cheng, L. Heldt, A. A. Kumthekar, Z. Zhao, L. Wei, E. Chi (Eds.), Sampling-Bias-Corrected Neural Modeling for Large Corpus Item Recommendations, 2019.

[10] M. Naumov, D. Mudigere, H. M. Shi, J. Huang, N. Sundaraman, J. Park, X. Wang, U. Gupta, C. Wu, A. G. Azzolini, D. Dzhulgakov, A. Mallevich, I. Cherniavskii, Y. Lu, R. Krishnamoorthi, A. Yu, V. Kondratenko, S. Pereira, X. Chen, W. Chen, V. Rao, B. Jia, L. Xiong, M. Smelyanskiy, Deep learning recommendation model for personalization and recommendation systems, CoRR abs/1906.00091 (2019). URL: http://arxiv.org/abs/1906.00091. arXiv:1906.00091.

[11] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[12] Last.fm, Last.fm music discovery api, https://www.last.fm/api, 2002. Accessed: 2024-08-04.

[13] S. Setiowati, T. B. Adji, I. Ardiyanto, Context-based awareness in location recommendation system to enhance recommendation quality: A review, in: 2018 International Conference on Information and Communications Technology (ICOIACT), IEEE, 2018, pp.

90–95.
[14] KANCHANA1990, Song dataset: 10,000 apple music tracks, https://www.kaggle.com/datasets/kanchana1990/apple-music-dataset-10000-tracks-uncovered, 2024. Accessed: 2024-08-04.
[15] MAHARSHIPANDYA, Spotify tracks dataset, https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset, 2022. Accessed: 2024-08-04.
[16] K. Bischoff, C. S. Firan, W. Nejdl, R. Paiu, Can all tags be used for search?, in: Proceedings of the 17th ACM conference on Information and knowledge management, 2008, pp. 193–202.
[17] onathan Ross, Groq api, https://console.groq.com/docs/quickstart, 2021. Accessed: 2024-08-04.
[18] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al., The llama 3 herd of models, arXiv preprint arXiv:2407.21783 (2024).
[19] Deezer, Deezer api, 2024. URL: https://developers.deezer.com/, accessed: 2024-08-04.
[20] A. J. Jeckmans, M. Beye, Z. Erkin, P. Hartel, R. L. Lagendijk, Q. Tang, Privacy in recommender systems, Social media retrieval (2013) 263–281.
[21] D. Hendrycks, K. Gimpel, Bridging nonlinearities and stochastic regularizers with gaussian error linear units, CoRR abs/1606.08415 (2016). URL: http://arxiv.org/abs/1606.08415. arXiv:1606.08415.