

# Taylor's Tunes

## *Mood-Driven Music Recommendations with Large Language Models*

Group 4

Duong Le: duong.h.le@aalto.fi  
Hieu Pham: hieu.pham@aalto.fi  
Tommaso Canova: tommaso.canova@studenti.unitn.it

February 2024

## Introduction

### Context

In the contemporary digital era, the significance of Language Learning Models (LLMs) has become increasingly paramount. These models serve as powerful tools in understanding and interpreting human language, thereby facilitating effective communication and interaction in various domains. Despite their potential, LLMs are often underutilized, with their full capabilities yet to be harnessed. LLMs hold immense potential in areas such as emotion detection and mental health support, where they can be tailored to identify subtle emotional cues in text and provide empathetic responses. This underlines the need for further exploration and development of LLMs, to unlock their full potential and contribute to societal well-being. Indeed, it is noteworthy to mention that individuals often seek solace in music rather than mere advice. According to [1], music serves as a powerful medium for emotional expression and regulation, self-awareness, and fostering a sense of social relatedness.

### Project statement

For this reason, our project aims to create an end-to-end product that is capable of recommending songs based on the mood of its users. We mixed open source Large Language Models such as *Mixtral-8x7b*, *Llama3-70b-8192*, *Llama3-8b-8192*, *Gemma-7b-it* with *Retrieval Augmented Generation (RAG)* techniques using lyrics from the artist Taylor Swift's songs to be able to understand human emotions and recommend pieces of music accordingly.

We propose a web application called "*Taylor's Tune*" which incorporate a pipeline composed of a Vector database containing the lyrics and the main

features of the song using *Qdrant* and *Langchain* as orchestration tool to allow the RAG mechanism to work with the previously cited LLMs.

The application enables the user to articulate their emotions in a "chat-like environment" and to receive the top  $k$  (where  $k$  is a parameter) songs based on a similarity measure that takes into account the detected emotions and the semantic proximity between the main keywords and the lyrics of the song. In our planned use case, the user starts a conversation with Taylor's Tunes and in each message describes how its feeling are and Taylor's Tunes would capture these emotions based on the context or explicitly from the user's text. At the end of the conversation, Taylor's Tunes outputs  $k$  Taylor Swift songs that best match with the emotion captured during the session. The whole project is open source and available at this *git hub repository* [2].

## Methods

In this section, the systematic approach chosen to achieve the project's objective is explained by describing the primary tools utilized that have subsequently enabled the actual implementation of the project. Initially, an Explorative Data Analysis (Section *Explorative Data Analysis (EDA)*) was conducted on the data pertaining to the available songs. Following this, these data were stored in a Vector Database (Section *Vector Database Usage and Representation*). Subsequently, a pipeline was prepared to allow the users to receive song recommendations based on their prompt (Section *System architecture*).

### Explorative Data Analysis (EDA)

In this subsection, the dataset used in the project is described, along with the primary insights obtained through Explorative Data Analysis (EDA) on the lyrics and features of each song.

#### Taylor Swift dataset

A tailored dataset on Taylor Swift songs has been proposed in *Mansfield* and *Seligman* work [3]. The authors of the paper put a specific focus on highlighting the level of happiness/optimism and strength commitment to a relationship based on the lyrics and chordal tones of each song (180 in total). Analysing the songs a custom marker indicated as *MIQ* ("Male in Question") has been proposed to address the song toward Taylor Swift speaking directly, or in an indirect way by expressing her feelings. The happiness score is composed of four criteria, most of which can reach a score between -3 and +3. The first criterion encapsulates Taylor Swift self feelings in the song, the second is referred to as "*Glass Half Full*" and measures the song author's outlook on life. The third one is split into two halves: the negative one refers to negative emotions such as depression, anger, and denial while the positive half embodies positive emotions. The fourth one measures the tempo and musical feel of the song

based on its key and BPM, while a final  $\pm 3$  score was reserved for songs with either positive or negative "hyperbolic line" that lead to a sudden decrease or increase of emotional intensity.

With a similar structure, the relationship Criteria is proposed. "*Seriousness of Topics Discussed*" takes into consideration how committed the author and the MIQ were to each other based on what topics they discussed together. The second criterion takes into consideration the future prospects in the relationship. the third criterion quantifies how the MIQ feels about Taylor Swift, while the final one is how much time Taylor and the MIQ seem to spend together throughout the course of the song.

A scoring example for one of the happiness criteria is given in Table 1

Table 1: Glass Half Full: one of four happiness criteria

Score	Description	Example
-3	All imagery is depressing	"And then the cold came, the dark days when fear crept into my mind" ( <i>Back To December</i> )
-2	Nearly all depressing imagery	"How you laugh when you lie / You said the gun was mine / Isn't cool, no I don't like you / But I got smarter, I got harder" ( <i>Look What You Made Me Do</i> )
-1	Majority depressing imagery	"Stealing hearts and running off and never saying sorry / But if I'm a thief then / He can join the heist" ( <i>...Ready For It?</i> )
0	Equal amounts of happy and sad imagery	"Rain came pouring down when I was drowning / That's when I could finally breathe" ( <i>Clean</i> )
1	Majority positive imagery	"We're happy, free, confused, and lonely in the best way / It's miserable and magical" (22)
2	Nearly all positive imagery	"This love is difficult, but it's real / Don't be afraid, we'll make it out of this mess / It's a love story, baby just say yes" ( <i>Love Story</i> )
3	All imagery is positive	"And all I feel in my stomach is butterflies / The beautiful kind, making up for lost time" ( <i>Everything Has Changed</i> )

## Lyrical analysis

To analyze the singer's texts (lyrics), it was necessary to remove the *stopwords* from the texts. However, the list of 179 stopwords provided by the *NLTK package* proved to be insufficient for this task, as abbreviations are often presented in the song lyrics. Therefore, a custom extended list with more than 1300 stopwords was designed.

Firstly, the various emotions of the songs were analyzed using two methods. The first method is based on a distilled version of *RoBERTa* [4], [5], which is capable of classifying a text according to the following emotions: *anger*, *disgust*, *fear*, *joy*, *neutral*, *sadness*, *surprise*. The second method is based on the text classification using *NRCLex* [6], which, given a text, allows assigning a score to the following categories: *fear*, *anger*, *anticipation*, *trust*, *surprise*, *positive*, *negative*, *sadness*, *disgust*, *joy*.

Contrary to initial expectations, NRCLex proved to be much more useful than RoBERTa. By assigning a score to each emotion, it was possible to extrapolate the top 3 emotions for each album. On the other hand, as RoBERTa is a classifier, it provided fewer insights. Figures 1 and 2 present a comparison

of the two results obtained on the ‘1989’ album. As we can see the results are slightly different, on one hand we can see that according to NRCLex the top 1 emotion is almost equally divided among a positive and a negative feeling, while in the second and third top emotions cluster we can see how these feelings are unpacked obtaining more insights. On the other hand, a totally different result is obtained using RoBERTa, according to which most of the songs in that album are more inclined to happiness.

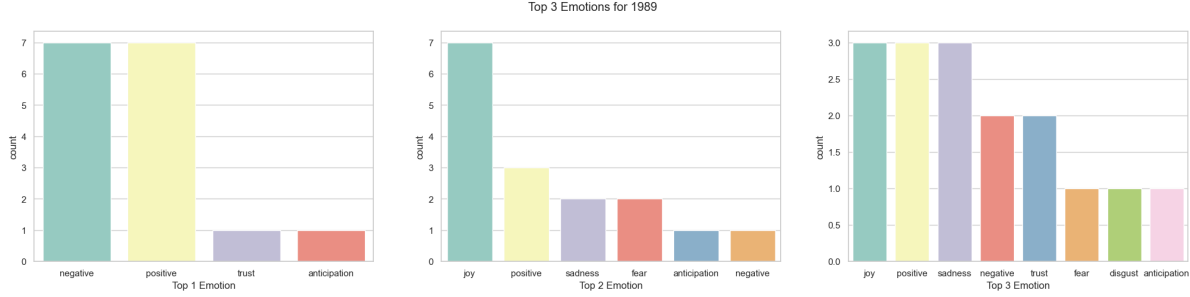


Figure 1: Top 3 emotions for album ‘1989’ obtained using NRCLex

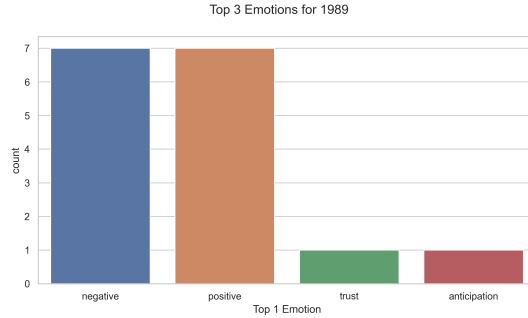


Figure 2: Top emotion obtained for album ‘1989’ using RoBERTa

Utilizing NRCLex as the primary tool, a comparative analysis was conducted on all songs in Taylor Swift’s discography. It was observed that a majority of the songs exude positivity. This could be attributed to the substantial number of songs written with a positive undertone. However, a non-trivial portion of them also conveyed negative emotions. Upon further examination of the secondary and tertiary dominant emotions, it was discerned that joy was the most prevalent emotion, followed by sadness and trust. This pattern could be indicative of the recurring themes in the songs, which often revolve around intense romantic relationships. Such themes could potentially foster a positive perspective toward the future and the individual who is the subject of the author’s affection. Figure 3 illustrates the aforementioned observations. While Figure 4 depicts the

most common words considering all albums, as we can see most of the words are related to love, the physical aspect of the *MIQ*, and a self perceiving of emotions. More visualizations can be consulted in the file `songs-EDA.ipynb` on the repository.

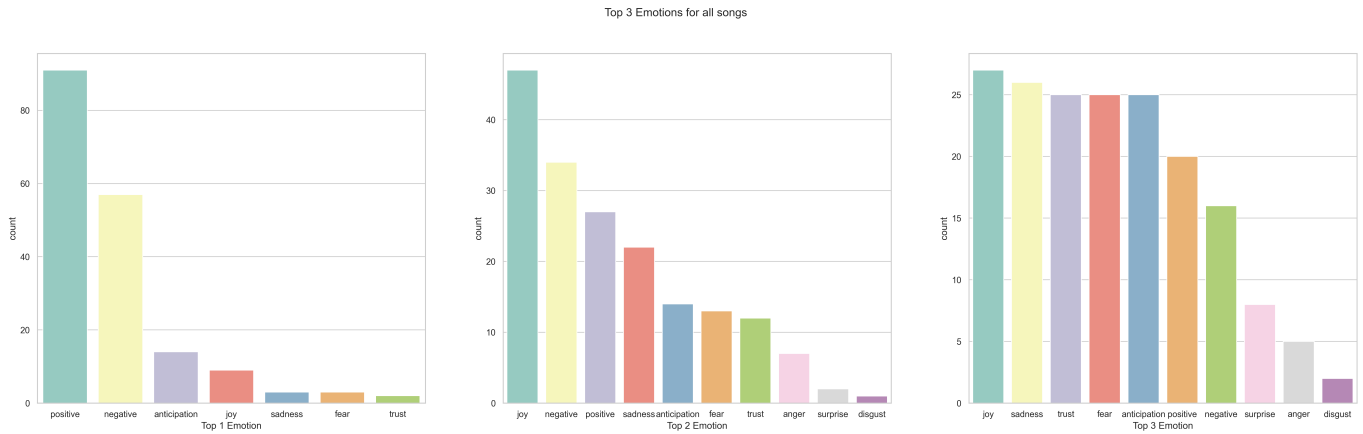


Figure 3: Top 3 emotions obtained using NRCLEX considering all albums

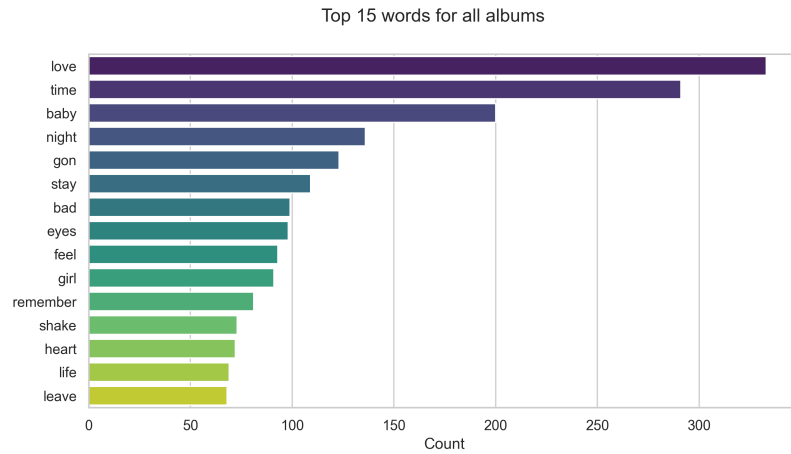


Figure 4: Top 15 words for all albums

## Song features analysis

As elucidated in the dedicated section (*Taylor Swift dataset*), each song is characterized by a set of features with a corresponding score. This allows the algo-

rithm to identify the most suitable song based on the perceived sentiment and the state of the relationship. In fact, in the work of Mansfield and Seligman [3], a statistical model is proposed. This model, based on the responses to certain questions posed to the user, attempts to determine a score to identify the most relevant entries in the dataset.

Upon analyzing all the albums, '*Lover*' emerged as the most "romantic" album, where the perception of the relationship is overall positive, and it is also the happiest album. On the other hand, '*Fearless*' is identified as the saddest album, while '*evermore*' is the album where the relationship is viewed from a purely negative perspective. These results can be observed in Figure 5a and 5b respectively.

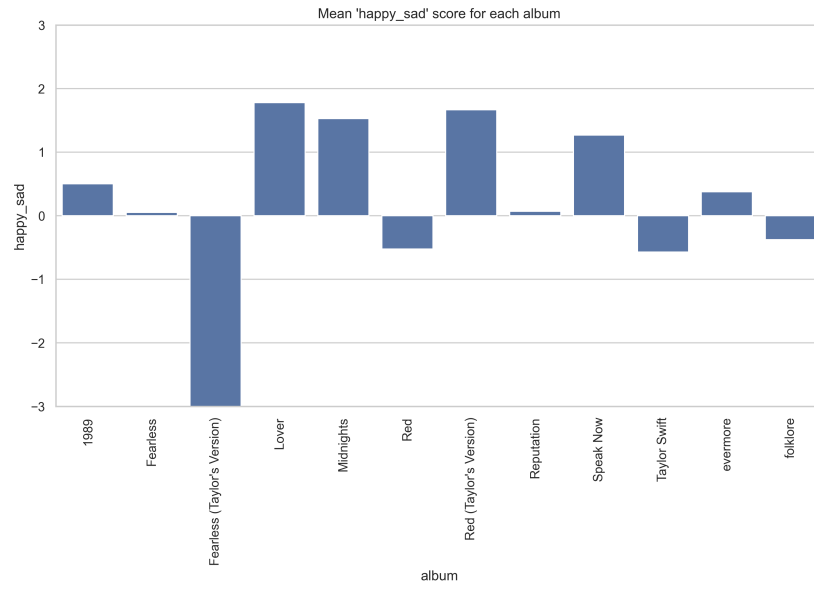
Furthermore, the dataset revealed that the happiest songs are: "*I Think He Knows*", "*Paper Rings*", "*I'm Only Me When I'm With You*", and "*Shake It Off*". Conversely, the saddest songs are: "*I Knew You Were Trouble*", "*Cold As You*", "*White Horse*", and "*All Too Well*". The songs that are most neutral from the perspective of happiness/sadness are: "*You Belong to Me*", "*Clean*", and "*Cornelia Street*", while those neutral from the relationship perspective are: "*Jump Then Fall*", "*Wonderland*", "*...Ready For It?*", and "*Change*".

Finally, for each criterion, two graphs were presented. The first graph is a pie chart which displays the distribution of various scores for that criterion among the songs. The second is a histogram to visually associate each score with the number of songs that achieve it, simultaneously associating the numerical value with a textual label as indicated in Mansfield's work [3]. For simplicity, Figure 6a, which pertains to the "*feeling\_of\_self*", is reproduced here. It clearly illustrates whether the artist primarily feels self-deprecating or secure and trusting of life circumstances.

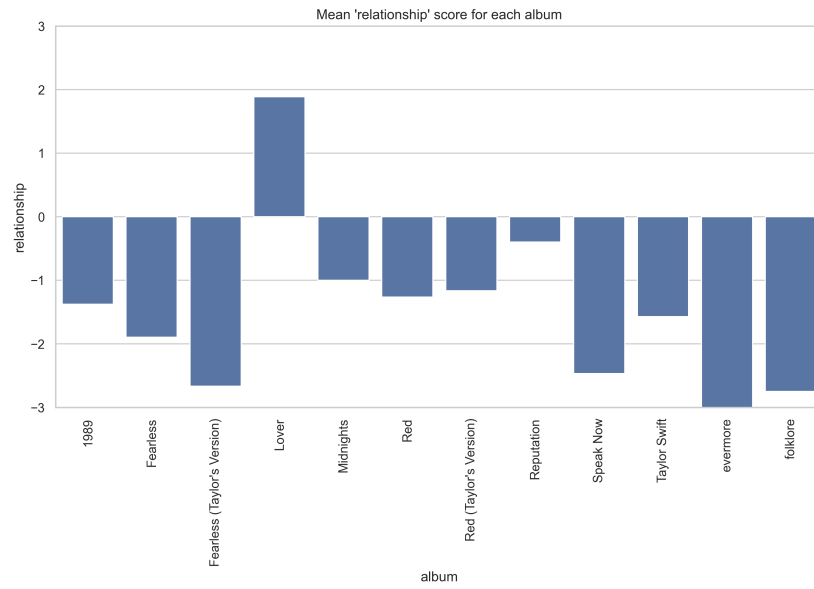
## Vector Database Usage and Representation

To process the lyrics data, we needed to find a way to represent said data. There are many ways to achieve this such as One-hot Encoding, Bag-of-Words, TF-IDF, etc... [7], but the end results would always have been numerical vectors. Thus, it makes sense that we utilized a database system that can effectively store and retrieve such vectors. That is where Vector Database Management System (**VDBMS** or just Vector Database for short) [8] comes in. Essentially, VDBMS is a functional software that focuses on effectively managing high-dimensional vectors. VDBMS is optimized to search for similar vectors to the queried vectors, rather than to look for the perfect match. This can be done effectively through indexing the data in the database. Current challenges with VDBMS include speed-accuracy trade-off, growing dimensionality and sparsity, and general maturity of the system. In our project, VDBMS is used to store the lyrics and metadata of the songs to use in Retrieval Augmented Generation (**RAG**).

RAG is a method applied to text generative models to provide said model a non-parametric memory that the model can retrieve data from [9]. Traditional text generative models are able to gain knowledge from data [10]. However,

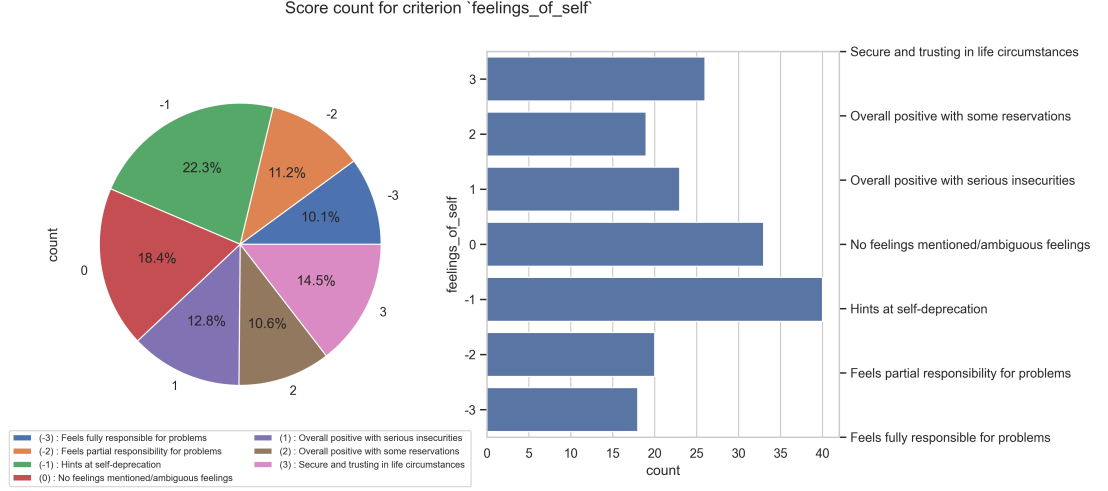


(a) Average value of 'happy\_sad' score for each album



(b) Average value of 'relationship' score for each album

Figure 5: Comparison of 'happy\_sad' and 'relationship' score for each album



(a) Pie chart and countplot used to show distribution of criterion "feeling\_of\_self"

they struggle to provide context for said knowledge and can suffer from hallucinations, where the model presents misinformation as truth [11]. Through the use of RAG, this problem can be mitigated by giving the model a method to revise and retrieve correct information [9]. Compared to other fine-tuning approaches, the former is analogous to giving the model a reference list to help it find the relevant information, while the latter is similar to teaching the model additional knowledge. Both have their own advantages, but for our application, RAG is a better choice. There are three types of RAG: *naive*, *advanced*, and *modular* RAG [12], as illustrated in Figure 8. They are characterized by the use of pre and post-retrieval processes, as well as additional modules in the case of modular RAG. In our project, we used naive RAG due to its simplicity in implementation, but future iterations of the projects are encouraged to try the other types.

In the context of this project, RAG is used as a knowledge database so that the model can compare and retrieve the songs that are best in line with the emotions of the users. In our project, we used Qdrant vector database. It is simple to set up. By using the free plan of **Qdrant**, we can host the database on their cloud server. We used two separate databases. The **first database** stored the embedded lyrics of the song. The **second database** stored the emotional criteria described in the Dataset section, which we will refer to as the **lyrics database** and **criteria database**, respectively. The reason for using two databases is due to the retrieval method we used. To retrieve a vector from the database, we calculate the cosine similarity between the query vector and the stored vector and select the top  $k$  matches (In the base case,  $k = 3$ ). Thus, if we only use the lyrics database, then essentially the model only retrieves the song that has the most similar lyrics semantic-wise with the user's input. On the other hand, there are no models specifically to predict the emotional criteria



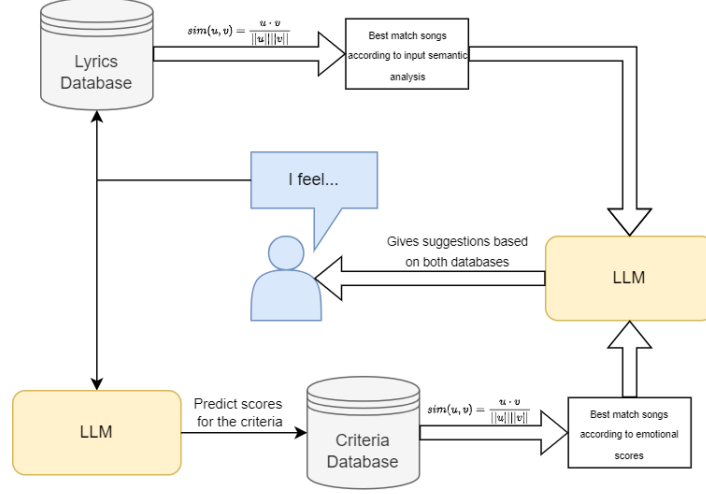


Figure 7: An illustration of how we generate song suggestions using RAG. We used two separate database so that the model can provide better suggestions to the user

from the user’s input, and training a model for this specific purpose was time-consuming and possibly out of the project’s scope. This makes using only the second database is not a good idea. Instead, our approach for this is that we first ask the LLM model to predict what are the criteria based on the input prompt. The predicted criteria is not shown to the user. We then retrieve  $k$  songs from the criteria database and  $k$  songs from the lyrics database using the predicted criteria scores and the input prompt, respectively. We also made it clear to the user which song comes from which database so that the users can make better judgments. An illustration of how RAG is used in our project can be seen in Figure 7.

In addition to the vector itself, we can also store additional metadata to use as context for the vector. When presenting the song to the user, we want the LLM to give a short summary of the song so that the user can understand why that song was chosen. In our first attempt, we stored the full lyrics and asked the model to give a summary based on those lyrics. However, we encountered a problem in which the model will sometimes print out the full lyrics, which is unintended. We theorized that the cause might be because the number of tokens is too much. Instead, we used another LLM to give the summary with at most 150 characters of each song and stored them in the metadata of the vector. That way, the LLM can just use the pre-made summary instead of predicting it itself when retrieving a song.

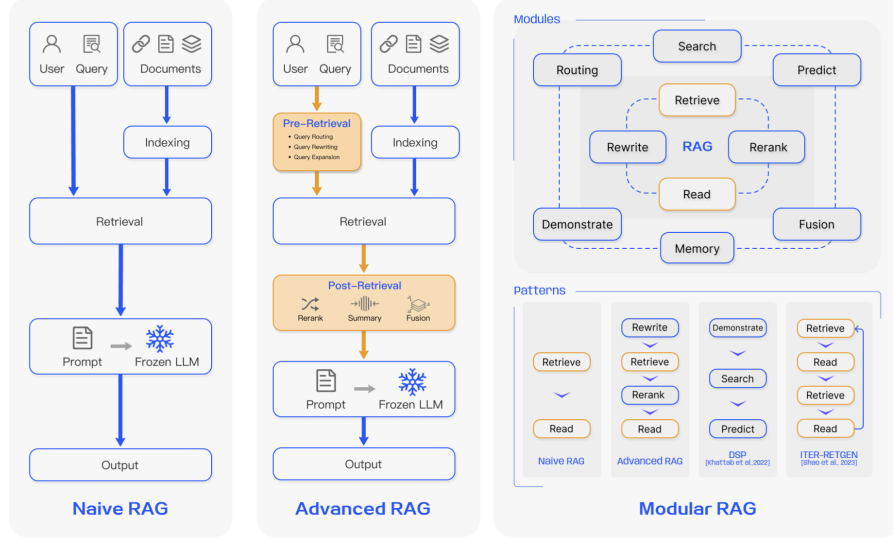


Figure 8: Comparison between the three paradigms of RAG. Figure taken from [12]

## System architecture

The entire system is composed of a web application developed using Streamlit, which encapsulates the system depicted in Figure 7. Specifically, in the web application, interactions with the chosen vector database and Language Model (LLM) are conducted via APIs. As mentioned in the previous section, the vector database is hosted on a free Google cluster provided by Qdrant. To interface with the LLM, the free plan from Groq is utilized, which allows for obtaining prompt responses with minimal inference time by leveraging dedicated hardware called the *Language Processing Unit™ (LPU)*. However, since we had to “specialize” the chosen LLM to perform specific tasks, we employed techniques of *Prompt Engineering*, drawing inspiration from the work of Bsharat et al. [13], which are explained in the subsequent section.

## Prompt Engineering

To ensure that the Language Model (LLM) we interact with performs only the task of song recommendations, it was necessary to break the LLM’s chain of thought using two auxiliary prompts, which we will define as “*pre-prompts*”, into which we will then insert the user’s actual prompt.

The chain of thoughts is thus divided into two parts that we will define as “*Evaluation*” and “*Recommendation*”, as depicted in Figure 9.

In the **evaluation** part, instructions regarding its task are provided to the LLM, dividing them into two tasks. In the **first task**, the LLM is expected

to give a score from -3 to +3 for each criterion (song metadata) present in the dataset, thus forming a vector with the metadata scores of length 8. For example, in this phase given the user input, a score will be associated with “*Feeling of self*”, “*Glass half full*”, and so on.

The second task, on the other hand, consists of trying to answer 6 questions regarding the person’s feelings and their own relationship, inferring this information from the given prompt, in turn constructing a vector of scores, this time of length 6. Both tasks rely on a lookup table provided in Mansfield paper [3], where a textual example is clearly associated with the relative score, as previously shown in Table 1.

The first vector, the one with the metadata, is used to perform a search in the vector database to obtain the songs most similar according to the detected criteria, while the second vector is passed to a snippet extracted from the “*taylor-swift*” library (developed from [3] authors), which provides us a baseline for comparison for the retrieved songs (note that these results are not used for the final recommendation). In addition, the second query to the vector database is made for lyrics similarity by embedding the user’s initial input using *all-MiniLM-L6-v2* embedder.

Subsequently, in the **recommendation** phase, the songs are incorporated into the second pre-prompt, accompanied by the following metadata: “*Title*”, “*Spotify Link*”, and “*Song Description*”. The latter is one of the most crucial features in the recommendation phase, as it justifies to the user why a particular song has been suggested and what significance it holds. In this final task, we explicitly ask the LLM to detect 5 keywords in the user’s prompt, identify the most relevant feeling, and after illustrating the top ‘k’ songs according to the emotional analysis and the top ‘k’ songs according to the lyrics matching, we request the LLM to provide a final comment on the user’s prompt (as a supportive friendly advice).

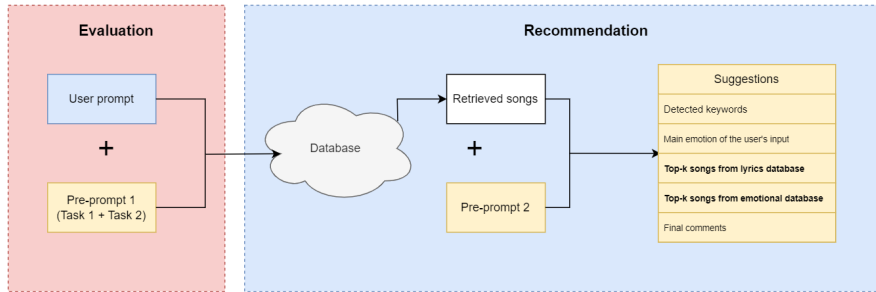


Figure 9: Prompt engineering pipeline

## Web application

A webapp with three main windows has been developed using *Streamlit*.

As can be seen from pictures 13, 14, 15, the first window, called *Chat* is split in two sides. On the left side it is possible to choose the model among the ones available, then select the  $k$  parameter concerning the amount of suggested songs, then the memory length of the agent (as the number of past prompts present in the context) and finally the upper limit of generated tokens from the LLM. Given a prompt, the agent will reply reporting the main keywords of the prompt, subsequently the overall mood and the suggested songs based on emotional and lyrical criteria, and finally a friendly advice to better handle the feelings described. In the second page, illustrated in Figures 16 and 17, the user can interact with the original `taylor swift` library developed in the Mansfield work [3], by answering to some questions. Based on these answers, inference will be performed on the statistical model described in Mansfield paper. The third page (Figures 18 and 19) contains the two databases used in the RAG process. We wanted to show this window to provide transparency to the end user and to allow it to familiarize itself with the dataset in case of curiosity (e.g. checking a word in the lyric of a song). As mentioned before, the project is open source and available at this link [2].

## Results

Taylor’s Tunes is an attempt to capture the emotions and thoughts of its user through the means of communication and use its understanding to interact with the user. Thus, to assess its performance we need to be able to quantify its Emotional Intelligence (EI), which is necessary for effective communication and social interactions [14]. Traditionally, the Theory of Mind (ToM) test has been applied to measure the ability to understand and correctly depict another’s mental state. LLMs before 2022 mostly showed no ability of ToM, being on par with small children’s ability [14]. ToM is a questionnaire ranging from false belief which means people have different understandings that diverge from reality [15], to pragmatic reasoning [16]. Due to ToM’s heterogeneous nature, it has been considered to not meet the reliability and validity standards to act as a psychometric test and has not been included in standardized tests on EI [17]. Wang et al. [14] has proposed a standardized test for Emotional Understanding (EU), which is a core component of EI, to be suitable for both humans and LLMs, termed the Situational Evaluation of Complex Emotional Understanding (SECEU). The test requires participants to evaluate complex emotions in realistic scenarios expressed by sentences. Specifically, the study conducted the test on both human volunteers and LLMs through 40 questions, each has 4 options of complex emotions. Participants were then asked to give the score for each emotion that sums up to 10. LLMs’ answers were compared with humans’ answers to see the similarities. In our study, due to the need to understand emotions through communicating with users, the original data from the SECEU study do not satisfy the purpose which is not a story but a conversation. Thus, we evaluated Taylor’s Tunes using a combination of the SECEU test and single statements such as tweets from Saif.M et al. study [18]

and general statements that resemble day-to-day conversation from Kaggle [19]. Further description of the data will be explained in the next parts.

Then, using the information about its user’s state of mind, Taylor’s Tunes recommends the songs after careful consideration. Thus, we carry out the second phase of evaluation: assessing its retrieval and recommendation ability. In this part, we will consider our implementation of Taylor’s Tunes with or without RAG, against `taylorlib`.

## Emotional Evaluation

### Single Statement Emotion Inference

To evaluate emotions, we let the model assess emotions after receiving 1000 prompts from 2 different datasets: one dataset of Twitter’s tweets, and another dataset of general statements. Both of these datasets include different human-labeled emotions. The Twitter dataset was labeled into 4 emotions: anger, fear, joy, and sadness. Each emotion was given a score of intensity between 0 and 1. The Kaggle data consists of 6 emotions: anger, fear, joy, love, sadness, and surprise. Unlike the Twitter dataset, each statement here has the value 1 for the emotion it conveys and 0 otherwise. Some examples of the two datasets can be seen in Table 2 and 3.

Table 2: Twitter dataset

<b>Tweet</b>	<b>Label</b>
When you lose somebody close to your heart you lose yourself as well	[0.0,0.0,0.0,0.708]
Omg someone asked them to sit and they reluctantly moved with a huff. Biiittccch-hhhhh	[0.771,0.0,0.0,0.0]
Man city’s kit is dreadful!	[0.0,0.0,0.0,0.522]
”Idk why but this Time around its so hard that it hurts, I already miss them all so much silly family friends ”	[0.0,0.604,0.0,0.0]
At the end of the day you gotta be happy for you	[0.0,0.0,0.6,0.0]

Table 3: Kaggle dataset

Prompt	Label
i love this weather i feel rejuvenated and energetic	[0,0,1,0,0,0]
i don t naturally feel like i ve got it going or where i feel vulnerable the lord has already got it covered	[0,1,0,0,0,0]
im feeling so terribly distracted these days	[1,0,0,0,0,0]
i feel lame just writing about life things anyway	[0,0,0,0,1,0]
i convinced tina to ride pharoahs fury which she doesnt like because it makes her face feel funny	[0,0,0,0,0,1]

To get the predictions from the model, we provided it with prompts using the Few-Shot prompting technique in combination with the data. We then asked the model to produce a vector of intensity scores with either 4 or 6 values based on the kind of data. The cosine similarity between the predicted vector and the label was then calculated and plotted into histograms in Figure 10.

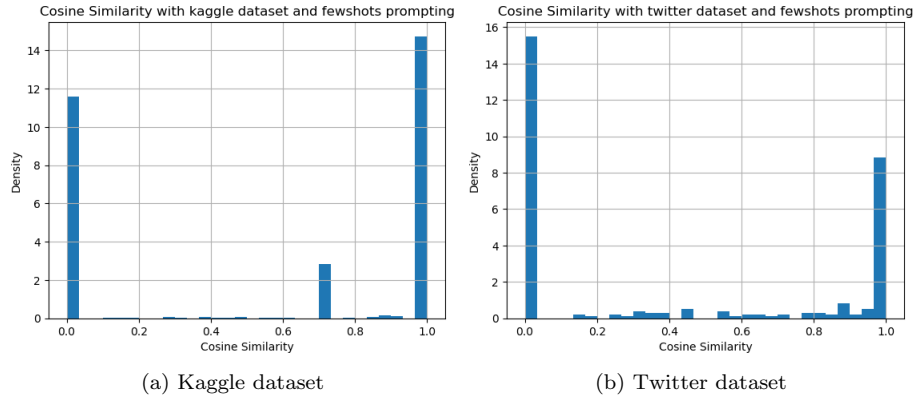


Figure 10: Cosine similarity

It can be seen that the cosine similarities for each dataset were polarized into either close to 1 or close to 0. This is not too surprising because of the nature of the data point: most vector consists of 1 or 2 intensities for emotions, and the rest are 0. However, it is worth noticing that the model evidently performs better given the prompt from the Kaggle dataset (Figure 10a), with higher density towards the 1 similarity. This can stem from the fact that the Kaggle

data more closely resembles actual human conversations, while tweets can have some characteristics that hinder the model’s ability. The characteristics can be the use of internet slang, some hints of sarcasm, trends, and topics that the LLM model fails to learn about, etc. With this knowledge in mind, we further analyzed the model’s capability to detect each emotion using the Kaggle dataset to see which emotion was identified similarly to humans. The result can be seen in Figure 11.

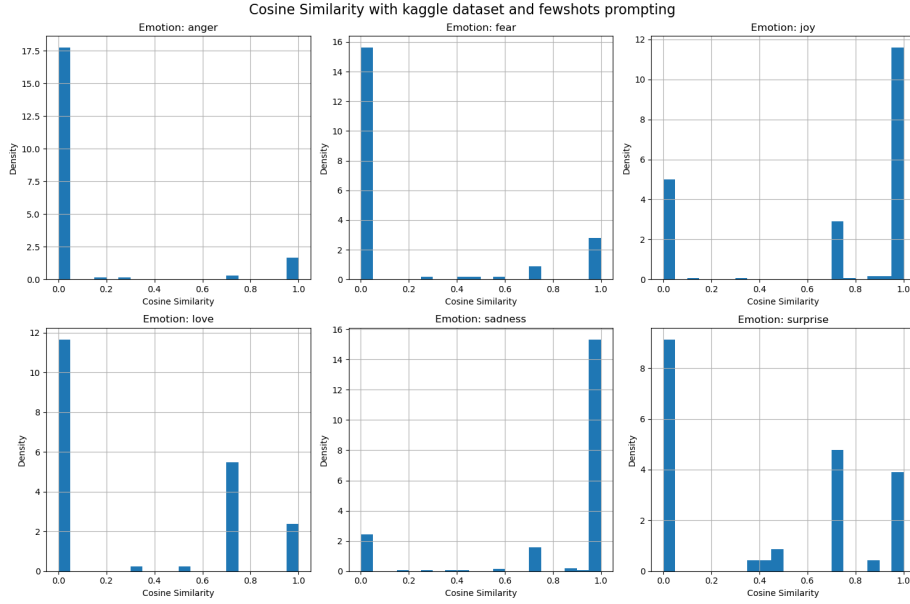


Figure 11: Cosine Similarity of each emotion

It can be seen that the Mixtral model seemed to catch joy and sadness emotions most effectively with cosine similarities of mostly 1, whereas love and surprise detection were a bit worse with a more varied range of cosine similarities. Anger and fear were detected incorrectly most of the time, leading to most similarity measurements being around 0. This series of evaluations shows that our LLM model tends to perform better with conversational statements, in comparison to social media statements (tweets in our case). Moreover, negative emotions on the more extreme spectrum such as anger and fear are challenging to notice, while more basic, to some extent, emotions such as joy and sadness were easier to predict.

### SECEU Test

For the next step, we want to assess the Mixtral model’s ability to understand emotions in the context of a small story by using the data from the SECEU study [14]. The SECEU test suggests using 40 items, each has a short story and





$$\text{Pearson} = \frac{\sum_{i=1}^{40} (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{40} (X_i - \bar{X})^2 \sum_{i=1}^{40} (Y_i - \bar{Y})^2}} \quad (4)$$

The results for our Mixtral model can be seen in Table 4.

SECEU Score	EQ	Pattern Similarity
3.23	108.04	0.21

Table 4: SECEU test results

Compared to the original study [14], our Mixtral model seems to perform better than most other modes, with the SECEU score of 3.23, the EQ of 108.04, and the Pattern Similarity of 0.21, only lower than that of GPT-4 and text-davinci-003 ##, outperforming numerous other LLMs such as LLaMA, Alpaca, previous versions of text-davinci and GPT, etc. This EQ result Mixtral also exceeds around 80% of the human population, showing a lot of potential in emotional intelligence.

## Retrieval Evaluation

After evaluating the emotional intelligence of our LLM model, we want to assess its main function: to recommend Taylor Swift’s songs that match users’ emotions. We compare the base version of Taylor’s Tunes with and without RAG, against the statistical model created by Mansfield and Seligman [3]. To simulate a user’s request, we use the prompt:

**Prompt:** "I feel lonely because my girlfriend and I recently broke up. I still care about her, but she wants nothing to do with me. I feel like I need some space to figure out how I truly feel, even though I want her back."

From the prompt, Taylor’s Tunes managed to extract the general sense of emotion as sadness, confusion, and longing, and generably fit the criteria. However, without RAG, Taylor’s Tunes recommends different songs every iteration with the same prompt, since there is no prior knowledge about the criteria for selecting songs and their respective score. The first iteration returned songs such as: "All Too Well", "Back to December", "We Are Never Ever Getting Back Together", "I Almost Do", "Red", while the second iteration returned: "All Too Well", "Back to December", "Clean", "The 1", "Cornelia Street", and repeating the process a combination of different songs were recommended even for the same prompt. Even though the songs recommended did manage to follow the general mood of the user, we found the lack of concrete measurement, which undoubtedly played a part in the randomness of the song recommendation, to be undesirable. Thus, we want to utilize the set of criteria in the recommendation phase by applying RAG.

By applying RAG, Taylor’s Tunes also considered the lyrics from the vector database that matched the criteria and returned a consistent set of songs. For

the above prompt, the recommended songs were: "How You Get The Girl", "We Are Never Ever Getting Back Together", and "A Place in This World". With this approach, we can simultaneously receive consistent and explainable results, with these songs still meeting the predefined criteria.

Our criteria took inspiration from the ~~study by the~~ Python library `taylorlib` [3] and developed upon it. Thus, we can compare how the original framework functions and see our improvements. Using the sentimental analysis that our model returns, we fed the score of criteria into the model. The framework essentially asks 6 questions about the user's state of mind, then uses a linear combination of those scores to find the song fitted. However, due to the nature of the linear combinations, there was a much smaller representational ability for emotions in songs. This led to some questions having drastically different points given that eventually played no role in the song recommendation. Moreover, the song "Cold As You" is heavily biased by the model for some undiscovered reasons. This song can be recommended multiple times in one iteration, and changing the score for some questions can still not change the recommendation. In short, Taylor's Tunes utilized the criteria system that `taylorlib` proposed, and in turn, built a more powerful and robust recommendation system.

## Conclusion

In this project, a solution called "Taylor's Tune" based on Large Language Models and Retrieval Augmented Generation has been developed to assist a user in finding the top  $k$  songs by the American artist Taylor Swift, based on the user's prompt. The user, by describing their emotional state and sentimental situation, receives from the agent a series of musical suggestions and friendly advice on how to better handle the situation. The application has been officially deployed online at the following link: <https://taylors-tune.streamlit.app>

We mainly implemented the model based on the LLM Mixtral, which has been shown to possess considerable emotional intelligence. It passed the SECEU test and performed better than many other LLMs, only to be outdone by GPT-4 and text-davinci-003 ~~###~~. We also noticed that Mixtral can label emotions more accurately when given conversational texts in comparison to when dealing with tweets. In both cases, Mixtral seemed to have problems detecting some strong negative emotions such as anger and fear, while joy and sadness were the easiest to recognize.

We also evaluated different implementations of our model, with or without RAG, against the existing recommendation library `taylorlib`. The comparison concludes that `taylorlib` has weaker representational power than our LLM due to its linear nature, and is also less robust to different user's inputs. However, the original paper has proposed a criteria system that prove to be working well with Taylor Swift's discography. By utilizing LLM and RAG, we created a more robust and efficient recommendation system for Taylor Swift songs that still took novelty from the criteria set, effectively improving upon the original library.

## Limitations

From the conducted tests, we observed that, unlike the *Mixtral* and *LLama* models, the *Gemma* model is not always able to understand the context provided by the user prompt, nor format its prediction output correctly. Nevertheless, it was decided to include it to allow users to compare it with the other models. Unfortunately, due to limited hardware capabilities, we are unable to perform inference on the models. Therefore, the project relies on API calls to *Groq*, which, in its free tier, imposes a limit on the number of *tokens per minute* (TPM) based on the chosen model. This limitation prevented us from fully utilizing the preprompting function, as we had to craft it more succinctly, providing a limited number of details to the LLM models. Moreover, the TPM limit, unfortunately, prevents the application from scaling to numerous users.

## Future works

Although the type of application made is totally tailored for Taylor Swift’s songs and is aimed at extrapolating information from the user’s prompt - mainly in terms of emotions and perceptions of their own relationship -, we think that this type of RAG-based architecture can also be used for other artists. In fact, in the case where the dataset is generalized considering only lyrics, summary of the main themes of the song and title, it is easily possible to make the architecture more "generic" allowing it to perform a retrieval based on semantic similarity with the lyrics and the meaning of the song. Therefore, this type of highly specialized architecture can, with some caution, be used with almost every type of artist who has songs with lyrics in their repertoire. For more "instrumental" tracks, one would have to rely on other features, such as those provided by Spotify’s APIs, such as: *key*, *liveness*, *loudness*, *speechiness*, *tempo* and so on.

Additionally, few-shot learning [20,21] could be utilized to improve the quality of the chat model further. This is a machine learning paradigm in which the model learns to accomplish the task using just a limited amount of examples (data). This method is useful in cases where the data is hard to obtain such as drug discovery [22] and most relevant to our cases, NLP [23–25]. To do this, we can design our own prompt and returned text from the chatbot, and due to the low number of examples needed, we can focus more on making the examples with high generalizability for the model to learn from. If succeed, we can avoid the need to specify the format in the pre-prompt and mitigate the problem of running out of tokens since the model can learn the suggestion format based on our examples. There have been previous attempts in utilizing few-shot in creating chatbot [26–28], and by incorporating their methods into our model, Taylor’s Tunes ability to suggest accurate songs can be further improved. Finally, it would be interesting to test how the effect of the '*temperature*' parameter could influence the responses of the LLM (as in this project it is implicitly set to 0)

On evaluation, we lack an evaluation system for the recommendation process to visualize the improvement in the performance of the different implementa-

tions. No prior dataset with pre-labeled scores for Taylor Swift’s songs was created, thus human assessment is required. For this purpose, we can have a dedicated group of audiences to experience Taylor’s Tunes and create a scoring system for each song. This will not only benefit the evaluation process of our project but will also facilitate further studies on the same subject of music recommendation systems.

## Divison of labor

*Division of labor: In a group work, describe who did what.* Duong Le was in charge of preprocessing the dataset, enhancing it with the lyrics summary, loading it into the vector database, and performing some early experiments with RAG. Hieu Pham was responsible for evaluating the architecture and running most of the experiments. Tommaso Canova handled the Explorative Data Analysis, developed custom prompts for the LLM, fine-tuned the RAG process, and built the overall architecture of the application (from chaining the LLM with the vector database up to publishing the frontend of the webapp).

## Appendix

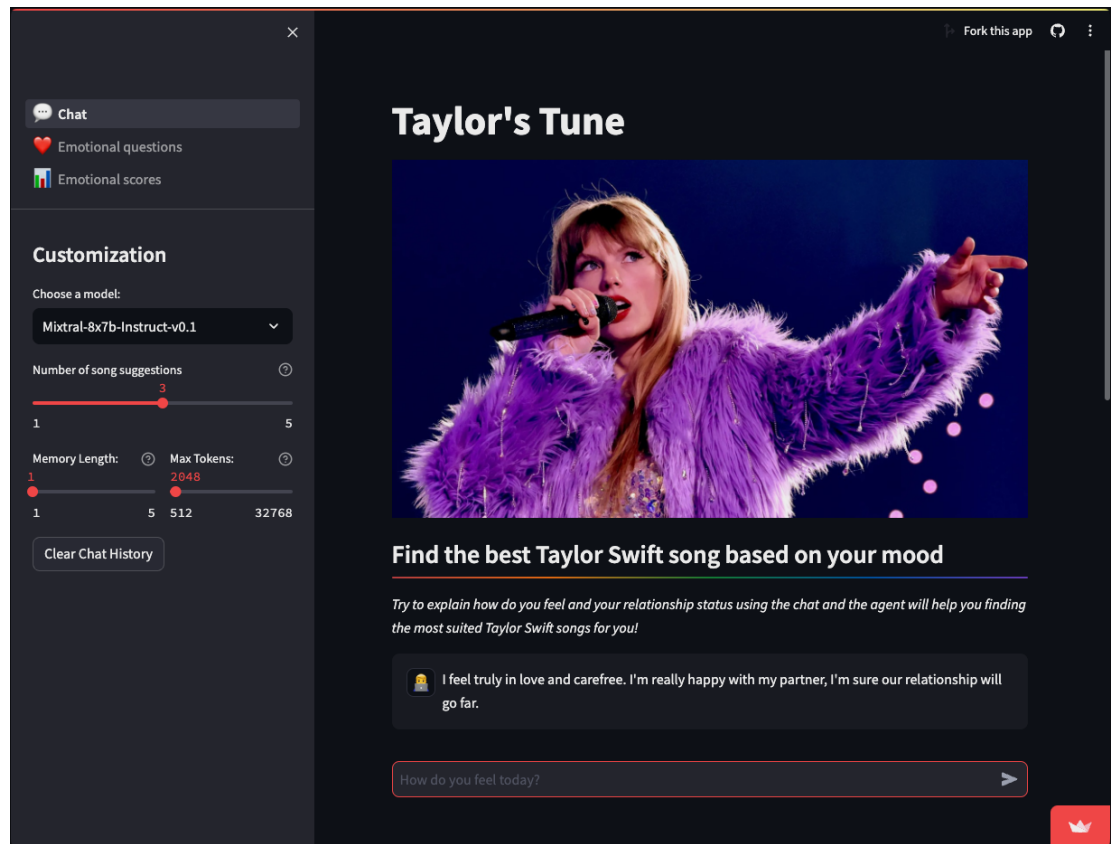


Figure 13: Main window with model parameters and user char

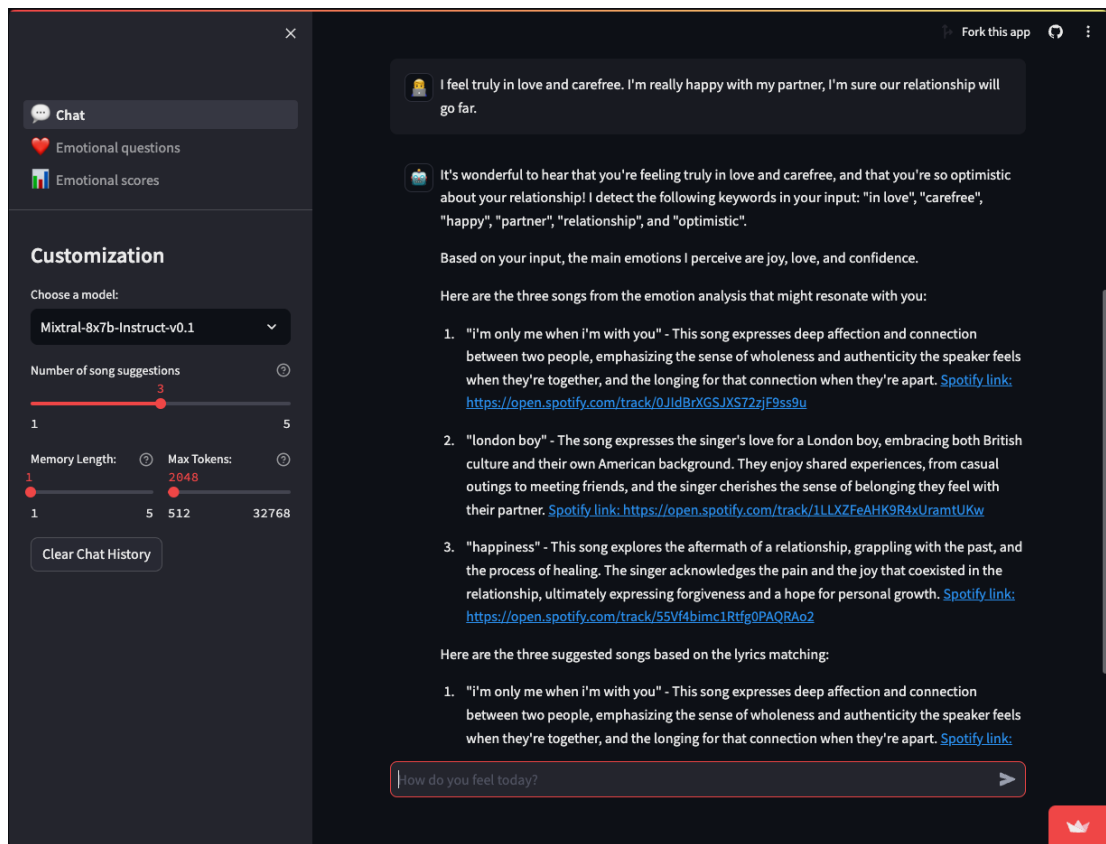


Figure 14: Main chat: agent reply

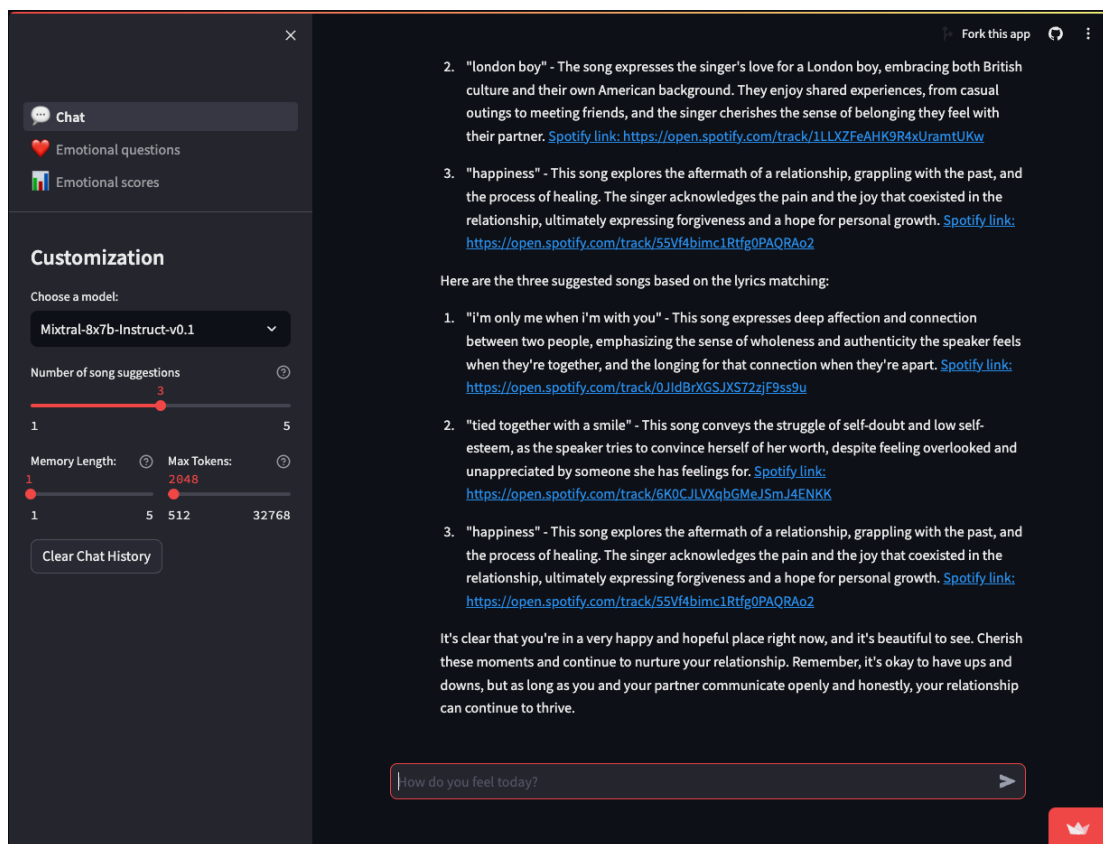


Figure 15: Main chat: agent reply (contd.)

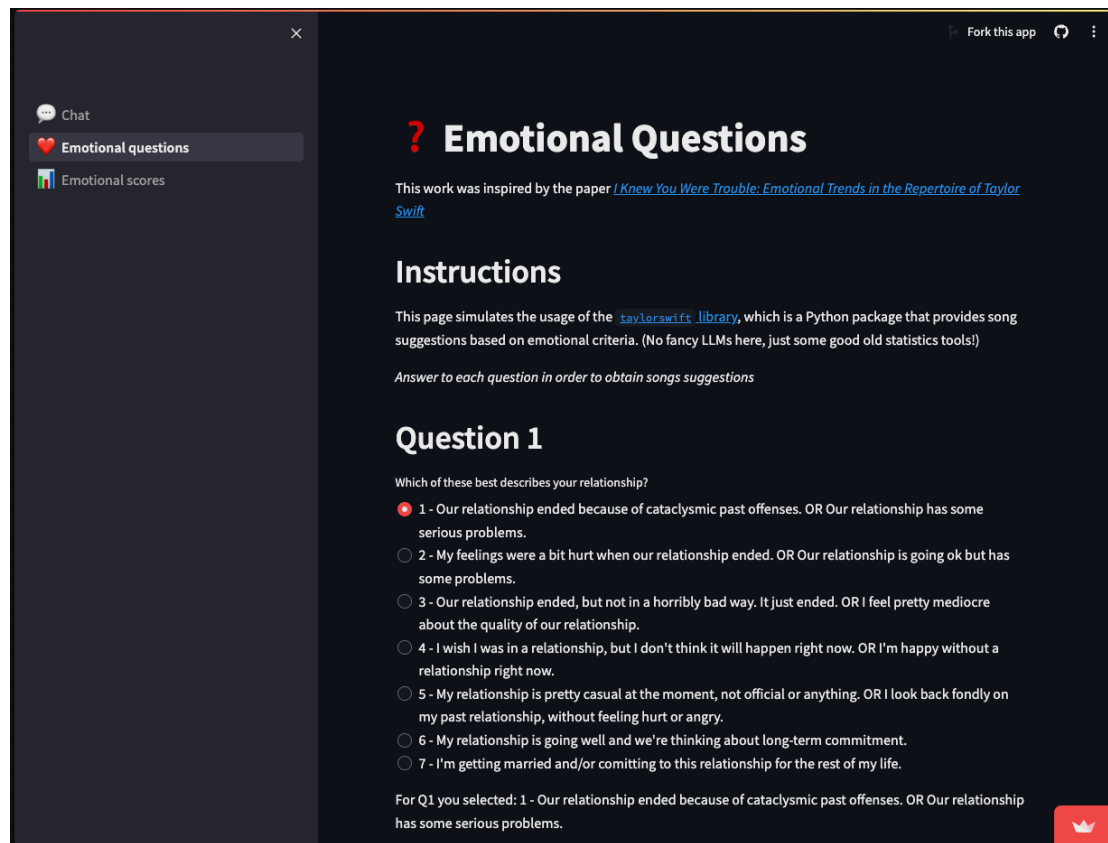


Figure 16: Emotional question window to run inference on original `taylorswift` library [3]



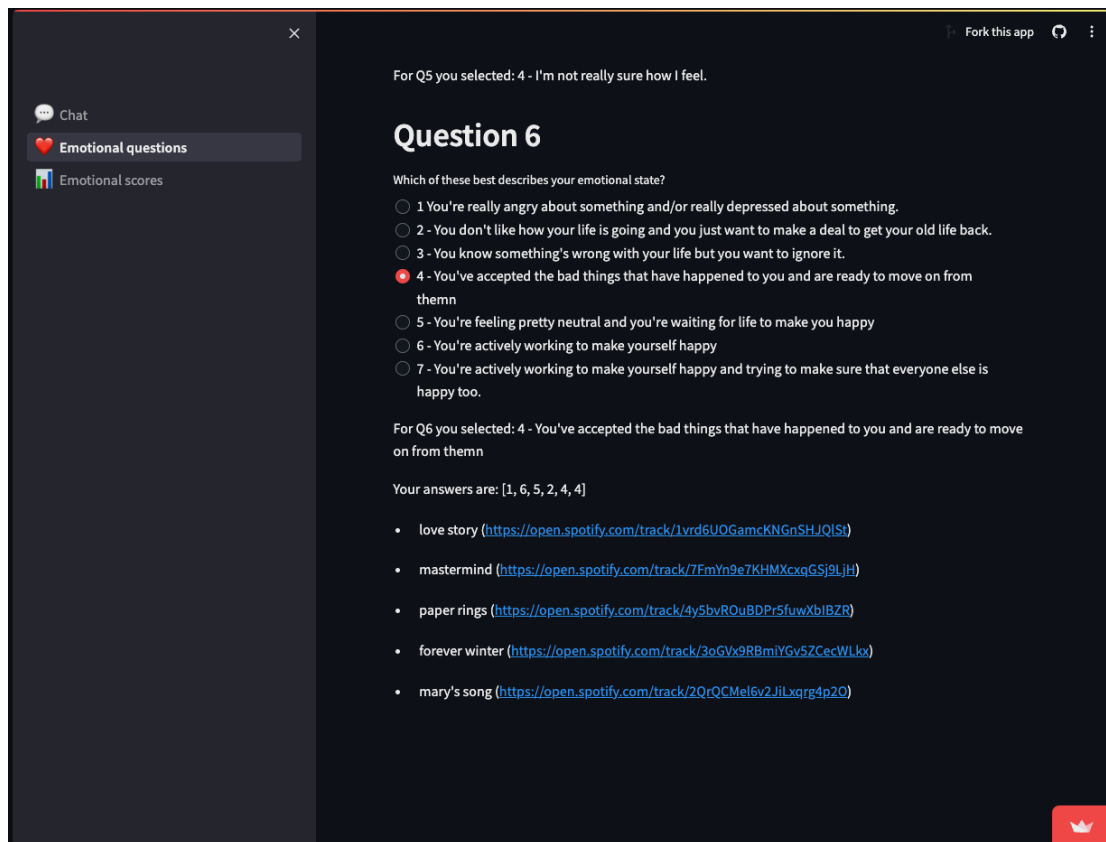


Figure 17: Results provide by the original taylorswift library

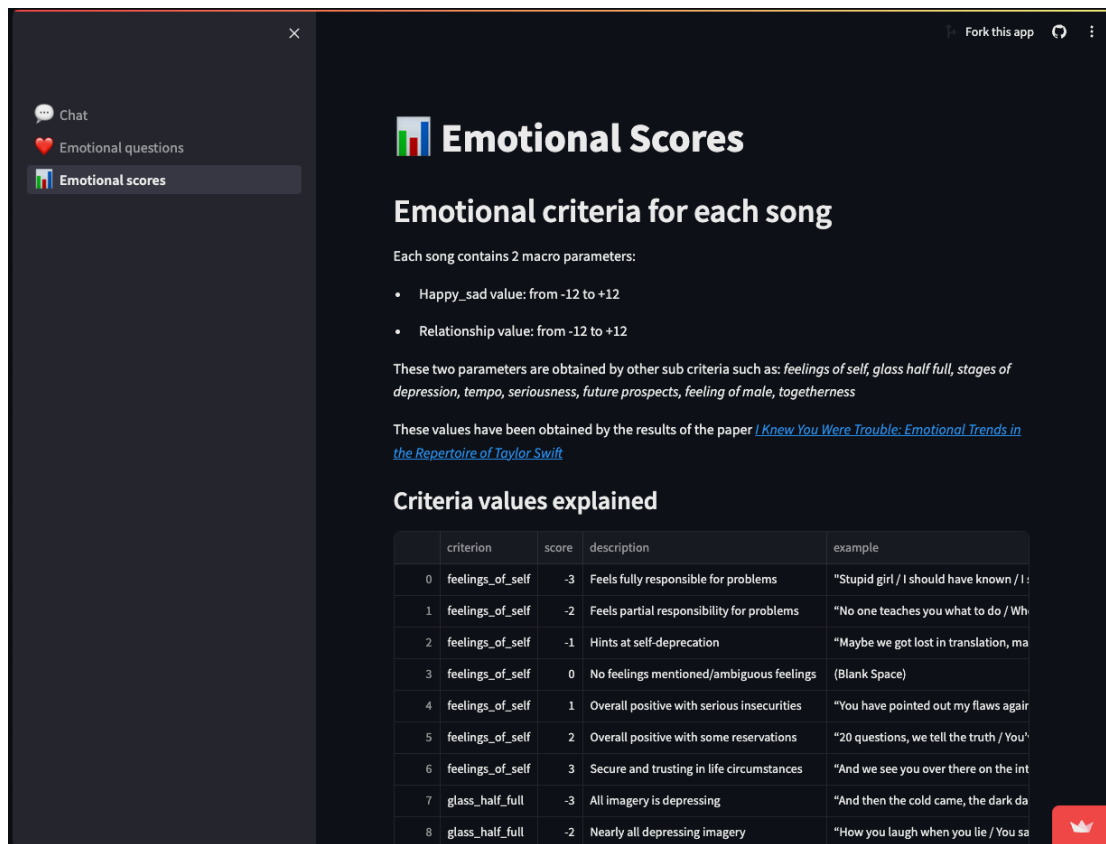


Figure 18: Database used window

Chat		Emotional questions		Emotional scores	
39	feelings_of_male	1	He expressed casual interest in a relationship	"I say 'I've heard that you've been out	
40	feelings_of_male	2	They are dating but not that seriously (she ha	"But you pull me in and I'm a little me	
41	feelings_of_male	3	Public declaration of love/commitment	"I never wanna see you walk away" (P	
42	togetherness	-3	Barriers to joint actions	Distance: "But you're in London and I	
43	togetherness	-2	No joint actions	(The Man)	
44	togetherness	-1	More things apart than together	(Holy Ground, You Belong With Me)	
45	togetherness	0	Equal amounts of time together and apart	(willow, Treacherous)	
46	togetherness	1	More things together than apart	(Stay Beautiful, New Year's Day)	
47	togetherness	2	They do everything together	(Paper Rings)	
48	togetherness	3	No identity as an individual	"I'm only up when you're not down /	

### Emotional criteria for each song

	song_name	album	happy_sad	relationship	feelings_of_self	glass_half_full
0	cold as you	Taylor Swift	-10	-8	-1	-3
1	i'm only me when i'm with you	Taylor Swift	9	10	3	3
2	invisible	Taylor Swift	-1	-4	0	-2
3	mary's song	Taylor Swift	5	12	0	2
4	our song	Taylor Swift	5	6	2	2
5	the outside	Taylor Swift	-5	-2	-2	-3
6	a perfectly good heart	Taylor Swift	-4	-8	-2	-3
7	picture to burn	Taylor Swift	4	-9	3	1
8	a place in this world	Taylor Swift	2	-2	1	0
9	should've said no	Taylor Swift	-5	-6	0	-2

Figure 19: Database used window (contd.)

## References

- [1] T. Schäfer, P. Sedlmeier, C. Städtler, and D. Huron, “The psychological functions of music listening,” *Frontiers in psychology*, vol. 4, p. 54458, 2013.
- [2] D. L. Tommaso Canova, Hieu Pham, “Taylor’s tune,” <https://github.com/cannox227/Taylor-s-Tune>, 2024.
- [3] M. Mansfield and D. Seligman, “I knew you were trouble: Emotional trends in the repertoire of taylor swift,” *arXiv preprint arXiv:2103.16737*, 2021.
- [4] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” 2019.
- [5] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter,” 2020.
- [6] S. M. Mohammad and P. D. Turney, “Crowdsourcing a word-emotion association lexicon,” *Computational Intelligence*, vol. 29, no. 3, pp. 436–465, 2013.
- [7] R. Egger, “Text representations and word embeddings: Vectorizing textual data,” in *Applied data science in tourism: Interdisciplinary approaches, methodologies, and applications*. Springer, 2022, pp. 335–361.
- [8] T. Taipalus, “Vector database management systems: Fundamental concepts, use-cases, and current challenges,” *Cognitive Systems Research*, p. 101216, 2024.
- [9] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel et al., “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [10] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, “Language models as knowledge bases?” *arXiv preprint arXiv:1909.01066*, 2019.
- [11] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, “Survey of hallucination in natural language generation,” *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.
- [12] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang, “Retrieval-augmented generation for large language models: A survey,” *arXiv preprint arXiv:2312.10997*, 2023.
- [13] S. M. Bsharat, A. Myrzakhan, and Z. Shen, “Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4,” 2024.

- [14] X. Wang, X. Li, Z. Yin, Y. Wu, and L. Jia, “Emotional intelligence of large language models,” 2023.
- [15] S. Baron-Cohen, A. M. Leslie, and U. Frith, “Does the autistic child have a “theory of mind” ?” *Cognition*, vol. 21, no. 1, pp. 37–46, 1985. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0010027785900228>
- [16] D. Sperber and D. Wilson, “Pragmatics, modularity and mind-reading,” *Mind & Language*, vol. 17, no. 1-2, pp. 3–23, 2002. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/1468-0017.00186>
- [17] J. D. Mayer, D. R. Caruso, and P. Salovey, “The ability model of emotional intelligence: Principles and updates,” *Emotion Review*, vol. 8, no. 4, pp. 290–300, 2016. [Online]. Available: <https://doi.org/10.1177/1754073916639667>
- [18] S. M. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, “Semeval-2018 Task 1: Affect in tweets,” in *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA, 2018.
- [19] N. Yewithana, “Emotions dataset,” <https://www.kaggle.com/datasets/nelgiriyeewithana/emotions>, 2024.
- [20] M. Fink, “Object classification from a single example utilizing class relevance metrics,” *Advances in neural information processing systems*, vol. 17, 2004.
- [21] L. Fei-Fei, R. Fergus, and P. Perona, “One-shot learning of object categories,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.
- [22] W. Yaqing, Y. Quanming, T. Kwok James, and M. Ni Lionel, “Generalizing from a few examples: A survey on few-shot learning,” *ACM computing surveys*, vol. 53, no. 3, pp. 1–34, 2020.
- [23] S. J. Semnani, V. Z. Yao, H. C. Zhang, and M. S. Lam, “Wikichat: A few-shot llm-based chatbot grounded with wikipedia,” *arXiv preprint arXiv:2305.14292*, 2023.
- [24] T. Bansal, R. Jha, and A. McCallum, “Learning to few-shot learn across diverse natural language classification tasks,” *arXiv preprint arXiv:1911.03863*, 2019.
- [25] Q. Ye, B. Y. Lin, and X. Ren, “Crossfit: A few-shot learning challenge for cross-task generalization in nlp,” *arXiv preprint arXiv:2104.08835*, 2021.
- [26] A. Madotto, Z. Lin, G. I. Winata, and P. Fung, “Few-shot bot: Prompt-based learning for dialogue systems,” *arXiv preprint arXiv:2110.08118*, 2021.

- [27] B. Peng, C. Zhu, C. Li, X. Li, J. Li, M. Zeng, and J. Gao, “Few-shot natural language generation for task-oriented dialog,” arXiv preprint arXiv:2002.12328, 2020.
- [28] M. Ahmed, H. U. Khan, and E. U. Munir, “Conversational ai: An explication of few-shot learning problem in transformers-based chatbot systems,” IEEE Transactions on Computational Social Systems, 2023.