Tailored Music Recommendations from Natural Language Prompts

Final Report

Issac Blumhoefer, Elijah Carlson, Lucas Harrison, Charles Hart
GorillaChow
blumh017@umn.edu, carl5411@umn.edu, harr2512@umn.edu, hart1299@umn.edu

1 Introduction

Music is personal, unique, and everywhere. For many, it follows us everywhere we go, and discovering new music is a process we hold in tender regard. In the current era, much of this discovery is handled by recommender systems proprietary to companies like SoundCloud or Spotify. These opaque algorithms can at times be disappointing, lacking variation or novelty in the songs they serve.

The motivation for our project comes from the desire to create custom playlists simply by inputting text. Not only would this simplify the process of creating playlists, but allow users to conveniently find music that fits the now, interpreting a certain mood or theme from a user prompt and recommend songs in that space. As opposed to manual exploration, we hope this system could interpret a user's request as naturally as they would express it to another person. Very recently, attempts have been made to accomplish this task in industy. Namely, with Spotify's new "AI" playlist generation feature. We find these current products may fail to fully embody the nuances and complexities of human emotion when recommending songs from user prompts.

Our approach attempts to create a more personalized and new experience for the user, addressing the need created by other lackluster systems. For example, given a prompt like "songs for a sunset," our goal is to recommend songs that represent this idea, creating a playlist in seconds compared to sifting through countless songs, ultimately saving a time and providing a new tool to probe into the vast realm of music.

Current music platforms recommend music and playlists based on listening history, popularity, and song titles. While this could be effective for generating playlists tied to past user preferences, these systems struggle with abstract or highly creative prompts. Their algorithms often miss the general "vibe" of such prompts due to limited integration of lyrical sentiment and audio feature analysis. Examples of these issues include things like an overreliance of song titles for mood assignment and a heavy influence of past preferences for new suggestions. Spotify specifically has been one of these platforms that we have compared our model against, and the listed issues are present in their recommender.

In addition, while there have been studies on music recommendation systems, gaps remain. Few systems focus on content-based prompts, and most fail to combine lyrical feeling with the audio of any given song in meaningful ways. Moreover, music recommendation is incredibly subjective—one person's perfect match may not resonate with another, adding a lot of difficulties with how to properly evaluate. Lastly, the immense amount of music that exists poses significant computational challenges, making it difficult to ensure coverage of all potential options.

Our project can appeal to anyone exploring new music, artists looking to reach new audiences in modern ways, and researchers interested in advancing recommendation algorithms. If successful, this system could revolutionize discovery of music (and possibly other media) by enabling users to find songs that resonate with complex emotions or unique contexts, such as creating playlists for specific moments or capturing the essence of ideas that could not be properly organized until now. This opens new possibilities for how people experience music, allowing unfamiliar and once unknown songs and artists to be discovered through sentiment and environment rather than a more traditional exploration of songs and artists.

2 Methods

2.1 Overall Approach

After reviewing the current landscape of products and research on this topic, we were able to formulate a game plan for tackling the problem. First, we acquired a HuggingFace dataset from Spotify that includes various features for numerous songs on the platform, most of which were newer music. These features included things such as the danceability or energy of a song, as determined on a scale by Spotify. This dataset then served as a baseline from which we ended up comparing our own final product. We handled user prompt evaluation with Gemini using zero-shot and few-shot prompting methods. Audio analysis was incorporated with the QWEN Audio LM, and lyrical sentiment analysis was performed. Together, these elements were orchestrated to create a system architecture that leverages the current SoTA in musical understanding.

2.2 Prompt Engineering

The next step in our process was defining how to set up the user end of our pipeline. Specifically, we had to figure out a way to go from user input defining a scenario to a mapped representation of the mood or emotion that could then be mapped to a playlist of songs. The technique for this user side of things was to directly use Google's Gemini 1.5 Flash to output adjectives and feature scores describing the user input. In order to improve accuracy of metric scoring, we utilized a few-shot prompting technique to contextualize the model's prompts. This consisted of giving the chatbot 20 different input examples (such as "Songs for a Night Drive") along with pre-defined scores for a number of features describing the inputs, for example features like valence or liveness. The scores for each of the 20 inputs were obtained through direct human evaluation of the inputs.

Four different google forms with 5 inputs were filled out by participants to create the labels for this training set. The human evaluation scores were also cross-checked with Gemini's un-prompted evaluation of the inputs. To ensure the most accurate results, we used the 20 inputs as our few-shot examples while prompting.

To generate descriptors from user prompts, Gemini is used in a zero-shot context. In order to extract descriptors that serve as a bridge between prompt and musical description, we prompt Gemini to associate the user's input with adjectives that could also be used to describe songs. These adjectives are then extracted as a list of keywords for later use in our search process.

In our deployed interface, a user can input the given mood or situation they want to generate a playlist for. Then, after the user hits enter, the few shot prompting with the 20 training points is inputted into Gemini with the new user input at the end. Gemini learns how to interpret the new input from the training prompts and then outputs features and adjectives that describe the new user prompt.

2.3 Audio Analysis

The other side of our pipeline involves turning songs into features that can then be mapped back to the user prompts. It's a similar challenge, but demands a different approach to fully capture all elements of a song. The first avenue we took was using the actual audio of songs to extractor information about them.

A recent review of the Music Understanding Language Model (Liu et al., 2023) space cited the QWEN-Audio Model as the current state-of-the-art across most music understanding tasks. In combination with its ready availability, this made it our choice for the direct audio analysis portion of our pipeline. We also tried MuLLaMA, which anecdotally was less promising and involved less convenient setup. The QWEN-Audio Model (Chu et al., 2024) analyzes various formats of audio files and accommodates natural language prompting and question-answering regarding the piece's analysis. Our pipeline compiles a preprocessed library of song descriptions by analyzing musical audio with QWEN, and prompting it for a list of descriptors that capture the most distinctive and important characteristics of the piece, including genre, mood, and standout musical elements.

2.4 Lyrical Sentiment Analysis

Another feature extraction tool we used for songs was lyrical sentiment analysis. We first acquired an already labeled dataset from ((Cano and Morisio, 2017)) and used this as our training data. This dataset includes songs, artists, lyrics, and finally an emotional state (happy, sad, relaxed, angry) as the label. This data was fed into a pre-trained BERT model from HuggingFace and fine-tuned on the lyrics. Creating a sentiment analysis model from lyrical text is critical (Edmonds and Sedoc, 2021) because the meaning behind lyrics is often different from the meaning of just everyday text/speech. The results of the final model on the 20% test data was an accuracy of 90% for all 4 classes after 5 epochs of training. These high results gave us confidence in the model's ability to analyze lyrical sentiments. After the model was determined, it was then applied to our database of potential songs. We did not initially have the lyrical data for these songs, but we were able to download and combine them from Kaggle. The end result was a new label for each of the potential songs that bucketed them into an emotional state based on their lyrics. This label served as yet another feature that mapped back to the user input to serve as that link between a user's desired situation and the songs that match it.

2.5 Search

The result of our preprocessing pipeline is a library of songs labeled with three key features: the eleven Spotify metric ratings (floats), up to thirty musical adjectives generated by QWEN (strings), and a label of the lyrical sentiment where available.

The collation of song labels ahead of time was a primary design choice in our system's architecture. This was done in order to reduce the recommendation process to a search task, as any sort of on-the-fly song analysis would invoke prohibitive computation. This means, however, that the scope of possible recommendations is limited to the confines of our preprocessed library.

Once the user's prompt is parsed by Gemini into its best guess of corresponding Spotify metric scores and in-domain descriptors, we compute match scores for each song in the library based on their labels. Spotify metric matches are computed via MSE (mean-squared error), while descriptor match scores are simple keyword-search (the number of matching adjectives). Interestingly, efforts to accommodate semantic proximity of musical descriptors via a cosine similarity search of descriptor embeddings yield worse anecdotal performance than keyword matching, and was abandoned in favor of the 'quick and dirty' approach.

The sentiment analysis of song lyrics unfortunately was limited in its applicability to our library, owing to a difference in origin of the data.

Once we had song scores for each of these metrics based on the user prompt, we calculated a final score based on a weighted sum of our submetrics and serve a playlist curation via a top-k sample of songs. For the purposes of the demo used for user feedback, this was the highest-scored 15 songs.

2.6 Success and Hypothesis

We believed our methods would be successful because of the human feedback we incorporated on the user side and the established language processing methods on the song side. Using direct human evaluation with a sample size of over 20 meant that the prompting of Gemini would be grounded in reality. We could have simply let Gemini predict the adjectives of a prompt by itself, but by choosing to include real-life input we made it more accurate and more likely to succeed. On the song side, we incorporated pre-defined methods with past success on both the audio and the lyrics. We knew these models would work because they have worked in the past on other research projects, whether that be QWEN on other audio data or BERT on any sentiment analysis data.

Our final hypothesis was that our final pipeline from prompt to playlist generation would have better results than the Spotify playlist generator, as defined by a human survey of both methods, scoring agreement on matching prompts to adjectives and then comparing the results of our playlist versus Spotify's for certain prompts. This hypothesis stems from the fact that our model incorporates more advanced features than Spotify, making it a more robust mapping tool.

2.7 Challenges in Development

One of the biggest challenges of this project was determining the exact mapping from a prompt to a playlist. We ended up settling on adjective and song feature matching, but that wasn't our first idea. Rather, our first idea was to build out a sentiment/emotion analysis on the prompt itself. This idea would have involved fine-tuning a multi-classifier HuggingFace model to deal with short expressions, but we eventually determined that this process would not have been as effective due to the brevity of most prompts. Instead, we pivoted to Gemini to allow for more creativity on the mapping side of things.

Another challenge we encountered was our database of songs. Firstly, web scraping song lyrics is a legal gray area and is often frowned upon due to copyright restrictions. Hence, for the lyrical emotion analysis we used a pre-approved lyrics database from Kaggle. The issue is that some songs that were used in the QWEN audio adjective assignment were not included in this database. And since the QWEN method is the bottleneck in our project due to time and space intensive issues, any particular song not found in the Kaggle database was a big loss. This lack of overlap between the lyrical and audio databases served to be an issue when it came to actual playlist generation because it limited our choice of songs considerably, and we actually were forced to not even consider the emotion label of a song's lyrics at times due to our small sample of songs. Having no central database of songs was a setback in its own right, and also put our group at a disadvantage towards Spotify, which has the ability to sift through so many songs within their platform.

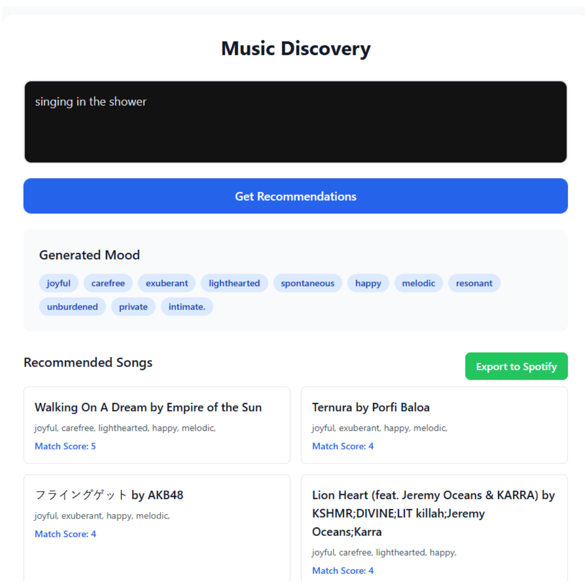
2.8 Novelty

While Spotify does have a working model that does a similar thing to ours, the model we have created gives much more emotionally based responses, along with being separated from the Spotify User. Spotify's model tends to give many results based solely off the name of the song – e.g. if you gave a prompt of "Sunny day", many of the resulting suggestions will be songs with the word "sunny" somewhere in the title. Furthermore, the results of the Spotify suggestions are very clearly biased towards songs that the user has already listened to, which is not very helpful when trying to find new songs. We do not have access to the code that Spotify uses for its model unfortunately, so these are simply observations from using the Spotify model ourselves. This is an important caveat to keep in mind, as we can only make guesses at the architectural differences between our systems and theirs.

Our model, on the other hand, takes a more in-depth approach to finding which songs to suggest. Using sentiment analysis in conjunction with Spotify metrics, we are able to capture the emotions that the prompt gives. Furthermore, we leverage analysis of the song's audio itself, generating a depth of description that is not apparent in Spotify's system. Additionally, since our suggestions are not based on listening history, our model can be much more helpful for suggesting new music to the listener.

3 Results

3.1 Demo



singing in the shower

joyful carefree exuberant lighthearted spontaneous happy melodic resonant unburdened private intimate.

Walking On A Dream by Empire of the Sun
joyful, carefree, lighthearted, happy, melodic,
Match Score: 5

Ternura by Porfi Baloa
joyful, exuberant, happy, melodic,
Match Score: 4

フライングゲット by AKB48
joyful, exuberant, happy, melodic,
Match Score: 4

Lion Heart (feat. Jeremy Oceans & KARRA) by KSHMR;DIVINE;LIT killah;Jeremy Oceans;Karra
joyful, carefree, lighthearted, happy,
Match Score: 4

Figure 1: Demo of working final product for an example prompt.

The first point of success for our project was to deliver a working prototype of the system. As of 12/12/2024, a live demo is available linked from our project website. For members of our group and participants at our poster session, this demo provided a functional demonstration of our system and accommodated a wide variety of untested prompts for playlist curation. Free of catastrophic failure, crucially during the demo on our poster day, we consider this a huge success and were able to deliver our system to users, even adding an "Export to Spotify" feature to move the playlist to a place you can listen.

3.2 Evaluation Metrics

The ability to create a wide variety of playlists from natural language mapped to moods and complex feelings is very subjective, and needs a lot of feedback to become a legitimate product. Strong agreement among users that even complex prompts were able to capture direct adjectives and make good use of our prompting model. Having a working demo that gives a quick result of songs. These were the key criteria by which we defined a successful project going into the first stages of development, and guided us in the evaluation of our completed prototype. Due to the subjective nature of our task, we looked to user feedback to quantify the system's performance.

3.2.1 User Prompt Processing

A foundational component of our music recommendation pipeline is the decomposition of user prompts into a set of in-domain adjectives. The result of this step in the pipeline determines the musical descriptors that are used to search song descriptions generated by the QWEN audio LM, and therefore must be accurate in order to serve fitting recommendations. Therefore, this was the first component of our system we evaluated via human feedback.

To get useful feedback on prompt descriptions, we designed a survey to see how users would agree to the adjectives that the model returned for the given prompt. To obtain these prompts, we first chose ones that had relatively straightforward ideas of what adjectives and songs could be derived from, and then as the list went down, the prompts got more abstract and nuanced, essentially trying to confuse the model. In total, 27 users responded to the survey, rating descriptions of 6 different prompts on a Likert scale, from strongly agree to strongly disagree.

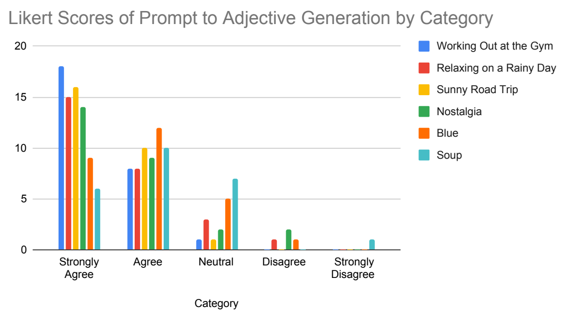


Figure 2: Likert Scale Visualization for Several Prompts

Visible in figure (2), the prompt descriptions were overwhelmingly rated as neutral or better, with a negative correlation between description rating and prompt opacity.

3.2.2 End-to-End Performance

Of course, the most fundamental result to evaluate was the end-to-end performance of our system–how users rate the playlists it curates. A survey to this end was designed with six prompts and their corresponding playlist recommendations. Each prompt-playlist pair was rated 1-5 on questions regarding fit, consistency, and prompt comprehensibility (Appendix, User Survey *1*). Finally, participants were asked to select one of the six prompts and use it to generate a playlist with Spotify, to compare to the GorillaChow option.

Question	**Avg. Score**
The songs in this playlist do a good job fitting the prompt	3.97
This playlist maintains a consistent theme	3.77
The prompt is understandable and could define a musical theme	4.42

Table 1: Mean Scores for Playlist Evaluation Questions

Participants generated 30 ratings of each question, with an average score of 3.97 / 5 for playlist fit (Table *1*), and 3.77 / 5 for playlist consistency. Users were also asked to rate the comprehensibility of each prompt, and we observed a moderate correlation of 0.77 between prompt comprehensibility and playlist consistency. Interestingly, prompt comprehensibility and playlist fit were not strongly correlated.

Finally, four participants generated playlists on Spotify and compared the results to the GorillaChow result. Three users preferred Spotify's playlist, in two instances citing higher numbers of songs they were already familiar with, and in one a more consistent theme. One user was ambivalent toward the comparison, elaborating that GorillaChow created good variety while Spotify produced a lot of the same songs, having more one-dimensional results than that of GorillaChow's output.

4 Discussion

Our model addresses challenges in data limitations and metric expansion by integrating analysis of audio features, lyrical sentiment, and user-defined prompts. While the Spotify dataset provided a useful baseline, its metrics alone were insufficient for capturing nuanced relationships between musical attributes for the scope of our project. By expanding these metrics, we achieved a more accurate understanding of sentiment, potentially reducing errors and improving accuracy. Combining different analyses worked well, as each individually often fell short in capturing the song's full context. The expanded metrics offered a more holistic approach, ensuring reliable recommendations. The overall subjectivity makes it hard to judge, but in our opinion, the current model improves upon the

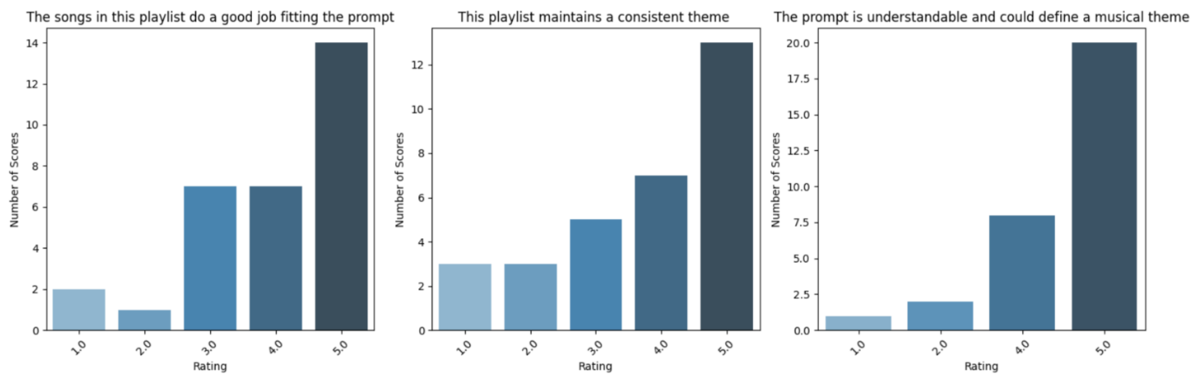


Figure 3: User Evaluation on playlists created by the model.

baseline.

4.1 Demo

The working product created is very easy to use and has simple results to understand. Once you input the prompt, there will be two different sets of results. The first is the generated adjectives that represent the mood of the prompt, and then the second is the songs that match the given prompt. You can see how similar the songs are to the prompt by how well they "match", and it will show both the title of the song and the artist. There is also a button titled "Export to Spotify" that will take the given collection of songs and create a Spotify playlist from it. From all the steps listed above, the final product came together very nicely and encompasses the prompt engineering, audio analysis, lyrical sentiment analysis, user feedback, and public datasets to incorporate the best song recommendations possible given the prompt.

Further development of the application housing our demo would involve streamlining the Spotify Account Integration Flow, which is a known issue at present.

4.2 User Feedback on the Model

4.2.1 Likert Scores on Prompt Adjectives

Looking at Figure 1, we can see that most users strongly agreed or agreed with the given prompts, especially the ones considered more straightforward. Even the prompts like "Blue" and "Soup" had strong agreement scores, with very few users disagreeing with any of the adjectives provided by the prompt that was supposed to gather mixed results for being so abstract. The average score of each prompt was above 3.8, and increased as the prompt became more straightforward, which was what we were looking for in addition to strong agreement results. Overall, the human evaluation showed that the model does a good job of describing certain prompts, especially towards more nuanced topics that can confuse other models and can normally be described as edge cases.

4.2.2 Playlist Generation Ratings

Overall, the six sample playlists presented in our user survey were rated fairly well. An average fit score of 3.97, "agree" on the Likert Scale, and average consistency score of 3.77 indicate that our system shows promise in generating quality playlists. A major limitation of our survey was the amount of feedback we recorded. Only six participants recorded responses, which doesn't represent the statistical rigor we may have wished to present.

In fact, survey turnout was a primary point of concern when this form was developed. Because evaluating a playlist involves manually listening to a number of songs, the task quickly can become laborious and hamper willingness to participate. In designing our survey, we tried to deliver the most painless process we could, and for this reason chose to provide pre-packaged prompts and generated playlists, instead of collecting user prompts, which may have led to less biased data. Perhaps the best option would have been to deliver surveys via an interface that allowed embedded audio playback of songs from our playlists, in order to streamline users' abilities to assess their quality. This was simply out of scope, however, as it would constitute the development of an entirely new application.

Furthermore, only four respondents took the time to compare our system head-to-head with Spotify, which is nowhere near the volume of feedback desirable to draw generalizable conclusions. In conjunction with limited turnout for our pre-generated playlist ratings, this response rate severely limits the confidence with which we can come to conclusions from our user study.

4.2.3 Untested Metrics

As mentioned, survey engagement was a critical limiting factor on the strength of our feedback suite. So much so that we left 'low-hanging fruit' on the table in terms of our survey design. While users rated the accuracy of the prompt description step in our pipeline, we conducted no study to determine the power of Spotify metrics in the music search process.

The evaluation metric used to search for songs in our library was a weighted sum of QWEN description match and Spotify Metrics' MSE, where the weights were chosen manually based on our own judgement. While this approach delivered a working solution, it carries several methodological limitations, namely a lack of empirical validation and potential lack of optimization. A more ideal approach would involve user feedback rating the quality of system outputs across several different scenarios, in order to analyze feedback for the sake of data-driven weight choices.

Notably, the Spotify metrics' MSE distance and QWEN descriptor match scores of songs in the library were often negatively correlated during search. This calls for further exploration into the particular utility of these metrics in defining a composite evaluation score.

4.3 Spotify vs. Our Demo

While limited in number, the responses we collected comparing our system to Spotify's head-to-head did not paint the picture of clear superiority we may have hoped for, according to Figure 3. In fact, the most positive response we received was neutral in preference between GorillaChow and Spotify's playlist. What we do get an indication of from this data, though, is one clear advantage held by Spotify's system. Two out of three responses in favor of Spotify's curation cited the inclusion of songs they were already familiar with as the reason for their preference. Spotify draws very *heavily* on a user's pre-existing library when curating playlists with its 'AI' feature. This user-data is not drawn on by our system, but could be given an extension of our work. A straightforward approach hinging on integration with the Spotify API could be to add a metric in our evaluation to bias the search process toward songs already in a user's library. Along with randomness to tune for discoverability, this could account for a need as expressed by what user feedback we do have.

The other reason users preferred Spotify to our system was the better consistency in its playlist. One reason GorillaChow could be less optimized for thematic consistency is the small library of music from which it currently draws, which we will discuss further later on.

4.4 Reproducibility

Our results are fairly simple to be reproduced assuming that others use the same models and data as we did, which is easily accessible through our GitHub. This would train a very similar model which then would produce similar results to ours. As for the conclusions and data obtained from the model, the demo on our website makes reproducing outputs very simple. The demo shows the generated mood of the prompt so that the prompt analysis can be separately tested. Then, each song shows which labels it has from the QWEN Audio Analysis and lyrical sentiment analysis. Lastly, we show a match score showing how similar the prompt descriptions and song descriptions are. All of this combined allows one to easily replicate any results we show from our demo.

4.5 Impact of Dataset

Our choice of datasets should not have hindered any other projects as all of the data we found was publicly available. The knowledge of such databases existing may have influenced others' choice of project had they known about them beforehand, but our specific choice of dataset had no influence on others research or development.

4.6 Ethics of Our Project

There is not much inherent risk to software that recommends music, though there are some things worth discussing here. Should our model become much more powerful, have a significantly larger database of songs available, and become well-known, it could potentially have a downside of some artists having difficulty starting off. If a model such as ours became used by the majority of the population to discover music, it would then be the case that most music recommended to people would be via the algorithm of that model. If an artist were to create music that the algorithm has a hard time labeling, for example, then the model would not be very likely to recommend it to other people. However, this would require an almost total shift in how music is currently discovered by the majority of people, as a lot of the time, music is discovered by recommendation from other people. Another potential issue in this scenario is that if someone were to gain unauthorized access to the database of this model, they could alter certain aspects of it to promote or obstruct the recommendation of certain songs. This could all be addressed by not trying to make the model the sole recommendation system to people, though it seems extremely unlikely that something like this would ever happen anyway.

4.7 Limitations

Given the relatively small number of songs (837) that are labeled in our database, we have a significantly smaller scope than a similar project such as Spotify's playlist maker. Because we have so few songs, we obviously cannot recommend much of a variety of songs, and even the recommendations that we do make may not be fully accurate since we just don't have the right songs to recommend. An example that shows this is if you put something like "deathcore" into the prompt. This, being a music genre, should be extremely easy to pull labels from and to match to songs. However, the best recommendation we can give has only a match score of 8/10. If you search the same thing into the basic Spotify search, hundreds of songs will appear all fitting the prompt. Thus, without more songs in our database, the recommendations are necessarily limited in their accuracy.

4.8 Future Work

As mentioned, expanding the breadth of our musical library would be a clear first-pass improvement to our system. Adding more songs to our database would improve every aspect of the recommendation algorithm, providing a greater variety of music and including more songs in the search process to potentially match a user's request.

Another point of consideration is expanding our efforts in user testing. As discussed prior, leveraging more user feedback would allow for more rigorously optimized metrics, tailored to meaningful extrinsic evaluation. Overall, having more songs to work with and user data to adjust the model, we could greatly improve the accuracy of our results based on reasonable prompts. As discussed in other parts of this paper, very abstract prompts do not really have any proper recommendation.

Finally, other sources of potential optimization include more intricate integration with user's Spotify libraries and the inclusion of other sources of song metadata. Taking user's Spotify Libraries into account in our search process would allow more personally tailored curation, while including other available song attributes not processed by our system, such as date and popularity, could accommodate logistic specific prompts.

5 Conclusion

This project was a product of hard work across all four of us. It took a long time to find our footing, but overall the final product has everything we originally intended: a pipeline straight from the user's prompt to a collection of songs that accurately fit the proper genre, both tested and with good user feedback. Despite initially mixed feedback when compared directly to our competitor's feature at Spotify, we still feel our system shows promise and with some careful optimization may have a greater upside than they do.

References

Erion Cano and Maurizio Morisio. 2017. Moodylyrics: A sentiment annotated lyrics dataset. In *Proceedings of the 2017 International Conference on Intelligent Systems, Metaheuristics Swarm Intelligence*, pages 118–124. ACM.

Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, Chang Zhou, and Jingren Zhou. 2024. Qwen2-audio technical report.

Darren Edmonds and João Sedoc. 2021. Multi-emotion classification for song lyrics. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 221–235, Online. Association for Computational Linguistics.

Shansong Liu, Atin Sakkeer Hussain, Chenshuo Sun, and Ying Shan. 2023. Music understanding llama: Advancing text-to-music generation with question answering and captioning.

6 Appendix

User Survey 1: Spotify vs. GorillaChow evaluation

Playlists tested:

- "chill instrumental playlist I can have in the background while studying"
- "Sex playlist"
- "Music to conquer lands and define eras"
- "Classy dinner party"
- "The sonic wavelength paradigm aligns with my personal entropy threshold"
- "Some upbeat songs for my morning run - something that'll keep me motivated but not too intense"

Questions asked about each prompt-playlist pair:

- "The songs in this playlist do a good job fitting the prompt"
- "This playlist maintains a consistent theme"
- "The prompt is understandable and could define a musical theme"

User Survey 2: Agreement with Prompt Adjectives These prompts are in order of most to least straightforward by us and ChatGPT. Prompts tested (adjectives in parentheses): - Working out at the gym (motivated, energetic, driving, rhythmic, powerful) - Relaxing on a rainy day (calm, soothing, mellow, peaceful, tranquil) - Sunny road trip (upbeat, joyful, carefree, optimistic, bright) - Nostalgia (sentimental, wishful, melancholic, reflective, yearning) - Blue (serene, calm, cool, tranquil, peaceful) - Soup (brothy, rich, thick, hearty, light)

In addition to this, we had a lot of trouble finding publicly accessible data. We tried our best to join datasets together to have the best shot at analyzing a lot of music, and it was a grind to find and analyze datasets upon datasets of songs. The set of songs we have is something we feel encompasses a lot of genres of music enough to properly give good results.

In addition, subjectivity is incredibly tough in the music industry. We did our best to conduct surveys in order to get proper feedback, but sometimes it is hard to see what users want and need out of their musical preferences. We took the proper steps in order to properly evaluate our model, and got good feedback on our prompting, playlists, and Spotify's approach as well compared to ours.

Last but not least, thank you to the CSCI 5541 staff for a great semester. This was an incredibly interesting class, and during the AI boom, it was incredibly beneficial in following what is going on in world AI news, and each of our individual journeys in computer science, artificial intelligence, NLP, and beyond. It was a great class that forced each one of us to work outside of our comfort zones to create a great project.