Lyric-based Playlist Continuation with Transformers

Dhruthick Gowda Mohan
dgmohan@ucsd.edu

Suhas Hebbur Eshwar
shebbureshwar@ucsd.edu

1 What we proposed vs. what we accomplished

Our aim was to investigate the effectiveness of incorporating lyrics into music recommendation systems. Specifically, we proposed using a transformer architecture for mood classification based on lyrics, whose results are used for subsequent music recommendation. We were able to accomplish all the tasks we set out to to in the proposal and were also able to obtain good results, better than the cited papers in some aspects. We list our accomplishments below.

- ~~Gather and preprocess lyrics for MoodyLyrics dataset (Çano, 2017)~~: DONE.
- ~~Build and train a baseline classification model (Naive Bayes) for mood classification on the collected dataset and examine its performance~~: DONE
- ~~Initialize and fine-tune transformer-based architectures for mood classification and examine their performance~~: DONE
- ~~Perform error-analysis of the best-performing model (BERT) on miss-classified instances~~ : DONE
- ~~Gather lyrics for tracks in Spotify's Million Playlist Dataset (Chen et al., 2018)~~ : DONE
- ~~Design a baseline algorithm for recommendation (basic collaborative filtering) and evaluate its performance~~: DONE
- ~~Design a recommendation architecture that incorporates the mood classifier fine-tuned earlier and evaluated its performance~~: DONE
- ~~Analyse the importance of mood as a feature in the recommendation system~~: DONE

2 Related work

Music emotion classification using lyrics has been performed on traditional lexicons (Hu and Downie, 2010) (Hu et al., 2009). Not only do the lexicons possess a highly restricted vocabulary, but the values must also be combined without incorporating any contextual information.

Several previous studies have explored the recognition of emotions through the analysis of English song lyrics. The work by (An et al., 2017) utilized Naïve Bayes Classifiers to classify lyrics into emotions of four distinct categories, (Happy, Angry, Relaxed, and Sad) achieving an accuracy rate of 68%.

In (Akella and Moh, 2019), multiple deep learning models were employed for mood classification using lyrics, including the Convolutional Neural Network (CNN), Bi-LSTM, and Convolutional Recurrent Neural Network (CRNN). The results indicated that the CNN model achieved an accuracy of 71%, while the Bi-LSTM model achieved 69.01% accuracy, and the CRNN model achieved 67.04% accuracy. However, in (Abdillah et al., 2020) the utilization of dropout parameters and activity regularization in the Bi-LSTM model yields a noteworthy accuracy of 91.08% in the task of emotion classification based on song lyrics. This achievement is particularly remarkable when compared to the performance of traditional machine learning methods for the same task.

When it comes to transformers, the work done by (Edmonds and Sedoc, 2021) reveals certain challenges associated with the emotion classification of song lyrics using state-of-the-art techniques when utilizing out-of-domain data. BERT models trained on extensive collections of tweets and dialogue fail to generalize effectively to lyrical data, except for emotions related to joy and sadness. Conversely, models fine-tuned specifically

on song lyrics achieve comparable levels of accuracy to models trained on out-of-domain data. Remarkably, these song-specific models maintain their performance even when working with significantly smaller lyrical datasets, aggregated from line to song level, annotated from diverse perspectives, and comprising various music genres. These findings emphasize the criticality of employing in-domain data for the accurate classification of emotions in song lyrics.

The study by (Agrawal et al., 2021), showcased the resilience of transformer-based methodology with fine-tuning for recognizing music emotions through lyrics across various datasets, surpassing previously employed approaches. In our work, we plan to employ a similar methodology to see how fine-tuned transformers on lyric data can help improve song recommendation for the task of playlist continuation.

In music recommendation, specifically, the task of automatic playlist continuation, the work by (Monti et al., 2018) took an ensemble approach by combining multiple recurrent neural networks, specifically Long-Short Term Memory (LSTM) cells, to predict the next track based on a sequence of tracks. By utilizing lyric metadata from the WASABI lyric corpus (Buffa et al., 2021), they developed lyric features that described different stylistic and linguistic dimensions of a song text, such as emotion and vocabulary. This highlights the importance of incorporating lyrics as a valuable source of information for understanding the content and emotional aspects of songs.

While LSTMs are one way to approach the task, most recommendation architectures often rely on matrix factorization as it has proved to be effective in many different domains. (Rubtsov et al., 2018) incorporate matrix factorization along with XGBoost in their approach for the task of automatic playlist continuation. Their work showed that models based on matrix factorization are capable to play the role of a base model for candidate selection. The XGBoost model assigns recommendation scores to the generated candidates based on the candidate's several features. We base our approach on their work and include mood-based statistics generated by our mood classifier as part of the candidate's features.

3 Our Datasets

We used 2 different datasets for classification and recommendation - MoodyLyrics for the classification of mood and a subset of the Spotify Million Playlist dataset for track recommendation.

MoodyLyrics (Çano, 2017): This dataset contains 2595 songs that are evenly distributed among the four quadrants of Russell's Valence-Arousal (V-A) circumplex model as shown in Figure 1 (Russell, 1980), which is a recognized framework for capturing musical emotions in a two-dimensional continuous space. In this model, emotions are represented as points in this space, where Valence indicates pleasantness and Arousal represents the energy content. To assign the V-A values at a word level, the authors utilized various existing lexicons such as ANEW, WordNet, and WordNet-Affect. These values were then averaged at the song level. The validity of these assignments was further confirmed by incorporating subjective human judgment of mood tags from the AllMusic Dataset (Malheiro et al., 2018). To ensure high representativeness of each category, the authors only included songs in each quadrant that surpassed specific thresholds for both Valence and Arousal values.

We collected the lyrics of the dataset using the Genius lyrics API. For the 2595 songs in the dataset, we were able to get lyrics for 2523 tracks. We also did some basic analysis of the dataset. Table 1 shows a sample of tracks for different moods. The column of relevant words in the lyrics was manually added. Figure 2 shows the distribution of moods among all the tracks in the dataset.

Spotify Million Playlist Dataset (Chen et al., 2018): Sampled from the over 4 billion public playlists on Spotify, this dataset of 1 million playlists consisting of over 2 million unique tracks by nearly 300,000 artists and represents the largest public dataset of music playlists in the world. The dataset includes public playlists created by US Spotify users between January 2010 and November 2017.

Due to computational constraints, we sampled over 140,000 tracks from the dataset and collected the lyrics using Genius API similarly as we did for the MoodyLyrics dataset. Out of this subset of tracks, we were able to obtain lyrics for over 113,000 tracks. Figure 3 shows the analysis of the dataset where we have calculated the number of songs in playlists and plotted the ranges. The

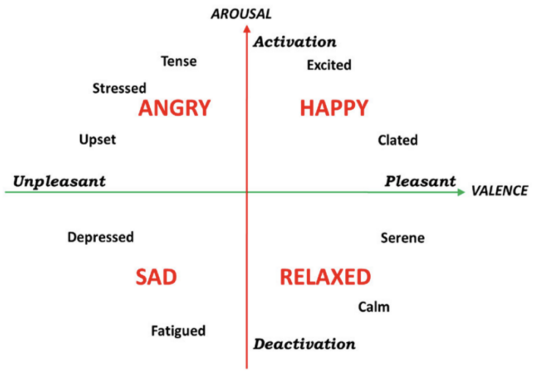

Figure 1: A Circumplex Model of Emotions (Russell, 1980).

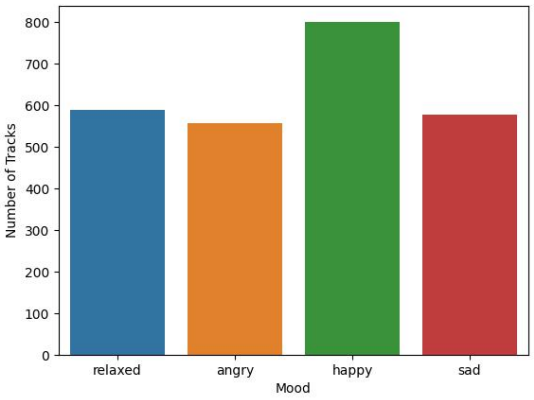

Figure 2: Tracks for each mood.

higher the number of tracks in a single playlist, the better the data available to make predictions for other similar tracks.

3.1 Data preprocessing

Since we are using BERT tokenizer we do several preprocessing steps to convert raw lyrics text that can be ingested by the BERT model. Here are some of the main steps:

1. **Tokenization**: The text is split into individual words, subwords, or characters depending on the specific tokenizer used. BERT uses a WordPiece tokenizer, which splits words into subword units based on a predefined vocabulary. For example, the word "running" might be split into "run" and "##ning" if "running" is not present in the vocabulary.

2. **Adding special Tokens**: Special tokens like "[CLS]" and "[SEP]" are added at the beginning of the input sequence to mark the separation between sentences respectively.

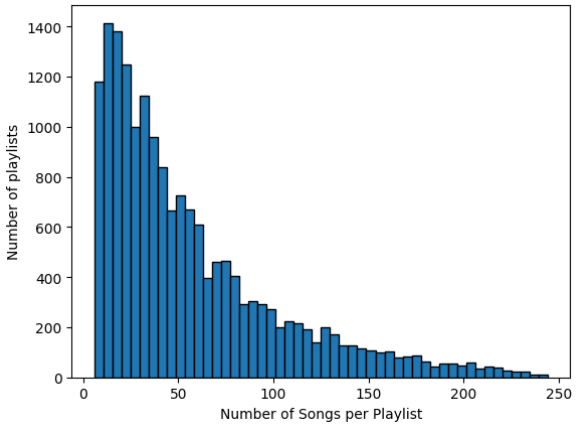

Figure 3: Histogram of songs per playlist

3. **Padding and Truncation**: Padding tokens such as "[PAD]" are used to make the input sequence lengths consistent, removing tokens from the end if the text is longer than the maximum length. If the input text is shorter, padding tokens are added.

4. **Token IDs**: Each token is mapped to a unique ID from the vocabulary which are used as input to BERT model during training and inference.

4 Baselines

For the task of mood classification, we chose the Naive Bayes classifier, which is a probabilistic machine learning model that assumes that the features are independent and makes predictions based on Bayes' theorem. The model takes a vector representing the term frequency of each token in the lyric as input and predicts one of the four emotion classes - happy, relaxed, sad, and angry. The 2523 track lyrics in our dataset were split into the train (2018 tracks, 80% split), validation (253 tracks, 10% split), and test (252 tracks, 10% split) sets, all of which contained equal proportions of examples in each emotion class. The Naive Bayes classifier resulted in a training accuracy of 87% and a validation accuracy of 72%.

For the task of track recommendation for automatic playlist continuation, we worked with a subset of the Spotify Millions Playlist due to memory constraints. Our subset consisted of 997,890 interactions, where each interaction is a unique playlist-track (playlist_id-track_id) pair, with a total of 17500 playlists and 113,434 unique tracks. We split this subset in a stratified manner based on the number of tracks associated with each playlist

Artist	Track name	Relevent words in lyrics	Mood
Beyonce	Dangerously in love	Love, happiest, proud, bloom, smile	Happy
Guns N Roses	Civil War	Fighting, crying, war, Vietnam, power hungry, soldiers	Angry
Inspiral carpets	Party in the sky	Loneliness, Stupid, Bad, Stupid	Sad
The Beatles	Two of us	Sunday, driving, home, writing, love, honey	Relaxed

Table 1: Sample of tracks and associated moods

into the train (598,734 interactions, 60% split), validation (199,578 interactions, 20% split), and test (199,578 interactions, 20% split) sets. We implemented a basic collaborative filtering-based approach for recommending new tracks for each of the playlists. In this approach, given a playlist of tracks, a list of similar tracks is retrieved for each track in the given playlist. Here, the similarity between two tracks is calculated using Jaccard similarity. Suppose t is a track in the given playlist, P_t is a set of all the playlists containing t, and t_c is a candidate track. The jaccard similarity is calculated as follows,

$$\text{Jaccard Similarity} = \frac{|P_t \cap P_{t_c}|}{|P_t \cup P_{t_c}|}.$$

The candidate tracks are all the other unique tracks from all the playlists in P_t. The similarity scores are averaged for all the tracks in the playlist and the candidate tracks with the highest similarity scores are returned as recommendations.

For evaluating the quality of the recommendation, we use the R-Precision@K metric, which is the fraction of relevant tracks that are present in the top-K recommendations. Here, relevant tracks are tracks in the validation set. The baseline approach resulted in a R-Precision@500 of 0.51 on the validation set.

5 Our approach

5.1 Methodology

In our approach for mood classification, we try two different transformer architectures - BERT and XLNet. While BERT (Bi-directional Encoder Representations from Transformers) (Devlin et al., 2019) is a masked language model where only a fraction of tokens in a fixed-length input are masked and predicted, XLNet (Yang et al., 2020) introduces permutation modeling where all tokens are predicted but in random order. XLNet also uses Transformer-XL (Dai et al., 2019) as its base architecture, enabling it to learn long-term dependencies beyond a fixed length. Both models are fine-tuned with the processed lyrics and emotion labels from our dataset.

For recommendation, we follow a similar approach as (Rubtsov et al., 2018). We first train a candidate selection model based on matrix factorization and then a final recommendation model that uses XGBoost. An interaction matrix between playlists and tracks is constructed using the training data. The candidate selection model performs matrix factorization on the interaction matrix, to learn the latent factors for playlists and tracks. Given a playlist-track pair (p, t) as input, the candidate selection model generates a score as follows:

$$\text{score}(p, t) = b_p + b_t + l_p \cdot l_t,$$

where, $b_p, b_t \in \mathbf{R}$ are the biases associated with the playlist and track, $l_p, l_t \in \mathbf{R}^n$ are the latent factors of the playlist and track, respectively. The dimension of the latent factors, n, can be chosen beforehand. Given a playlist of tracks, we sort the scores generated by the candidate selection model on all the other tracks that are not in the playlist in descending order and return the top 1000 tracks for final recommendation. This is done for all the playlists and their tracks in the train set.

In the final stage, we treat the problem as a classification task. For each of the 1000 candidate tracks generated for a given playlist, we assign a label of 1 if the playlist-track pair is present in the validation set, otherwise, we assign a label of 0. We also generate the features described in Table 2 for each playlist-track pair and train an XGBoost model with the binary logistic-loss and the hyperparameters specified by (Rubtsov et al., 2018). We include parameters generated by the candidate selection model, co-occurrence statis-

Features	Names	Description
Playlist and Track Bias	playlist_bias, track_bias	The biases associated with the playlist and track (b_p, b_t) generated by the candidate selection model.
Dot Product of Latent Factors	dot_product	The dot product between playlist latent factor and track latent factor $(l_p \cdot l_t)$ generated by the candidate selection model.
Candidate Selection Score	score	score(p, t) generated by the candidate selection model.
Rank	rank	The rank of t based on the scores generated by the candidate selection model for the top-1000 candidate tracks.
Co-Occurrence and Normalized Co-Occurrence Features	max_coo, min_coo, mean_coo, median_coo, max_ncoo, min_ncoo, mean_ncoo, median_coo	The minimum, maximum, mean, and median of co-occurrence and normalized co-occurrence values.
Playlist-Track Features	track_artist_count, track_album_count, track_artist_share, track_album_share	Number and share of tracks in p with the same artist and album as t.
Track Features	global_track_count, global_artist_count, global_album_count	Global frequency of t, and t's artist and album.
Playlist Features	unique_artist_count, unique_album_count, num_tracks_in_playlist	Number of unique artists and albums in p and length of p.
Mood-based Features	mood, playlist_mood	Mood of t and most common mood among tracks in p.

Table 2: Features associated with each playlist-track pair (p, t)

tics, playlist, and track statistics as features. For a given playlist-track pair (p, t_c), co-occurrence, n_{t_c,t_p}, is defined as the number of playlists that contain both t_c and t_p, where t_c is the candidate track and t_p is a track already present in playlist p. The normalized co-occurrence is given by $\frac{n_{t_c,t_p}}{n_{t_p}}$. These values are calculated for every track in the playlist. Finally, we also incorporate our lyric-based mood classifier to generate mood-based features. The candidate tracks with the highest prediction scores assigned by the XGBoost model are returned as the final recommended tracks.

5.2 Implementation

We fine-tuned both BERT and XLNet for mood classification of lyrics with pre-trained weights (bert-base-uncased) with the help of HuggingFace's transformers library and PyTorch. After hyperparameter tuning on the validation set, the best results were achieved by BERT when the model was trained for 2 epochs with a batch size of 32 and maximum input length of 256 while using Adam with weight decay for optimization with a learning rate of 3×10^{-5}. Due to memory constraints in our work environment, we weren't able to experiment with higher values of the maximum input length.

For recommendation, we referred to the implementation provided by (Rubtsov et al., 2018), which can be found at: https://github.com/VasiliyRubtsov/recsys2018. We used the LightFM library, which is a Python implementation of a number of popular recommendation algorithms and provides support for matrix factorization. The dimension of latent factors was set to 200 and the LightFM model for candidate selection was trained for 300 epochs with Weighted Approximate-Rank Pairwise (WARP)

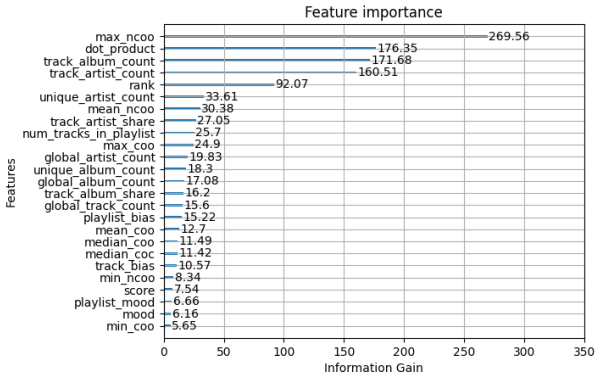

Figure 4: Importance of features included in the XGBoost model for recommendation

loss and learning rate of 0.02. We evaluated the model on only part of the validation set every five epochs since validation on the entire set consumed a considerable amount of time. The best model with respect to R-Precision@500 was obtained after 45 epochs. We then proceeded to generate candidates for the final stage of recommendation with XGBoost along with the features describes in Table 2. The XGBoost model was implemented with the same hyperparameters as described in (Rubtsov et al., 2018) with a binary logistic objective, learning rate of 0.1, decision tree boosting, and a maximum tree depth of 7 and evaluates performance using the AUC metric, while training for a maximum of 300 boosting rounds with early stopping after 30 rounds without improvement.

All of the above implementations were done on Google Colab, with 12 GB memory. We utilized the free GPU resource for training, tuning, and evaluating our transformer-based architectures, while the recommendation models were trained on the CPU. The data collection was done locally on Jupyter Notebook since it involved numerous API calls and consumed an excessive amount of time. Our final BERT model for mood classification took about 7 minutes to train, while the LightFM model and XGBoost model for recommendation, took approximately 72 and 122 minutes respectively. Our implementation can be found at `https://github.com/dhruthick/cse256project`, in which all the models in the folder `models` along with the `baseline_collaborative_filtering.ipynb` were implemented by us.

5.3 Results

For mood classification, BERT performed better than XLNet on the validation set, with a final validation accuracy of 94% while XLNet obtained a validation accuracy of 92%. Both architectures outperformed the baseline Naive-Bayes approach. The better performance may have been due to the difference in their training approach or due to the limited maximum input length we were restricted to due to memory constraints. Finally, we evaluated the BERT model on our test set and observed a test accuracy of 93%.

For recommendation, the two-stage strategy with LightFM and XGBoost resulted in an R-Precision@500 of 0.5667 on the test set, which is a significant improvement over our baseline. To analyze the effectiveness of mood-based features we re-trained and evaluated the XGBoost model without any mood-based features. This version of the recommendation model resulted in an R-Precision@500 of 0.5661, i.e. we observed only a 0.0005 increase in performance when mood-based features are included. To understand the importance of features we plot the information gain of each feature as shown in Figure 4, which is the average gain of splits that use the feature. We see that the mood-based features are only very slightly effective.

6 Error Analysis

Table 3 shows a few examples of the instances where the BERT model got the predictions wrong. Analysis of some of these instances gives us the following insights into the reason why the model might have got it wrong:

- Ambiguity in lyrics: Lyrics can be open to interpretation and can contain ambiguous phrases. The model may have difficulty in capturing the intended meaning if lyrics are context dependent.
- Musical aspects: BERT model heavily relies on the textual context provided by the lyrics alone. While lyrics can provide the theme of the song they might not capture the complete picture.
- Noise in the data: A few instances in the data did not have the actual lyrics of the song but some random text since the API is not entirely reliable which contributes to a reduction in accuracy

Table 4 shows a few instances from the test data where the baseline model gets the predictions wrong whereas the BERT model gets it right. Here we can see BERT model is a far superior model since it has the contextual understanding between words in a sentence and it is not easily deceived just by the presence of relevant words in the lyrics. BERT models are also able to capture underlying semantic relationships between words thereby able to identify semantic cues enabling them to make more accurate predictions

7 Contributions of Group Members

- Dhruthick: Built and trained classification and recommendation models, evaluation of models, report writing
- Suhas: Data collection and analysis, hyperparameter tuning, error analysis, related work, report writing

8 Conclusion

In conclusion, our aim with this project was to study the effectiveness of incorporating lyric-based features in music recommendation systems. Our approach, mainly involved two stages: In the first stage we try to classify the lyrics into one of four moods, and in the second stage we incorporate the moods generated by our classifier for track recommendations. Finally, we observed that mood-based features result in only a slightly better recommendation performance.

Throughout the project, several key takeaways emerged. We observed that the data collection stage presented significant challenges and consumed most of our time. Second, given a task-specific well-labeled dataset, transformer architectures are capable of providing outstanding performance than most other approaches for text classification. Finally, we also understood that evaluating recommendation models is very different from evaluating learning algorithms generally.

When implementing our approach, we found that building a complex recommendation model can be a tedious task that involves dealing with millions of interactions, which can be very challenging when working with limited memory. Regarding the mood classification itself, we were pleasantly surprised by the overall performance of our model, especially because our BERT model resulted in a better performance than the proposed XLNet architecture by (Agrawal et al., 2021).

In the future, we could try to gather or annotate a better dataset for lyric-based recommendation ourselves, since we believe the low effectiveness of mood-based features in our approach can maybe be attributed to how most user-created playlists on streaming services only contain popular music or only music by a particular artist or from a particular album, i.e tracks with diverse emotions, which is why features such as album and artist counts seem to be more important. We could create a dataset that only includes user-created playlists that are emotion-specific, to provide better recommendations than just popular music.

References

Abdillah, J., Asror, I., and Wibowo, Y. (2020). Emotion classification of song lyrics using bidirectional lstm method with glove word representation weighting. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 4:723–729.

Agrawal, Y., Shanker, R. G. R., and Alluri, V. (2021). Transformer-based approach towards music emotion recognition from lyrics. In *Lecture Notes in Computer Science*, pages 167–175. Springer International Publishing.

Akella, R. and Moh, T.-S. (2019). Mood classification with lyrics and convnets. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 511–514.

An, Y., Sun, S., and Wang, S. (2017). Naive bayes classifiers for music emotion classification based on lyrics. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, pages 635–638.

Buffa, M., Cabrio, E., Fell, M., Gandon, F., Giboin, A., Hennequin, R., Michel, F., Pauwels, J., Pellerin, G., Tikat, M., and Winckler, M. (2021). The WASABI Dataset: Cultural, Lyrics and Audio Analysis Metadata About 2 Million Popular Commercially Released Songs. In *The Semantic Web. ESWC 2021. Lecture Notes in Computer Science, vol 12731.*, pages 515–531.

Chen, C.-W., Lamere, P., Schedl, M., and Zamani, H. (2018). Recsys challenge 2018: Automatic music playlist continuation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 527–528, New York, NY, USA. Association for Computing Machinery.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Edmonds, D. and Sedoc, J. (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.

Track name	Artist name	Lyrics highlights	Predicted mood	True mood
It is Love	Living Colour	The glamour boys, swear, diva, always dancing, always laughing, playing the role, never have no money ...	Happy	Relaxed
Part of me	Tool	I know you well, part of me, know you better, than I know myself I know you best, better than anyone ...	Happy	Sad
Just stand back	Low	I cant decide and I cant hide, Make up my mind, waste of time, Here comes the knife,I could turn on you so fast, hit its got soul ...	Angry	Relaxed
Lady Pilot	Neko Case	Stars cant fight city lights, turned their backs on us Its true today, saw it from the plane, Aeroplanes were never built to fly down, down, ...	Relaxed	Angry

Table 3: Error Analysis of BERT model

Track name	Artist name	Lyrics highlights	Baseline output	BERT output	True output
Take me as I am	Faith Hill	Believe , you don't believe me, its forever, you dont believe me, you dream about the future? ...	Happy	Relaxed	Relaxed
When I hustle	Huey	Bad girls, sad girls, on the street at night (Walkin) Picking up strangers, spirit high...	Happy	Relaxed	Relaxed
One more empty chair	Blood red shoes	Don't wanna take it slow, take you home, watch the world explode, your glow, creep across my skull, slowly enter, crawl ...	Angry	Sad	Sad
Peace, Love and Happiness	G. Love & Special Sauce	Glad this week, get away Go home, throw my clothes, Cant wait to see you, lovely ...	Happy	Relaxed	Relaxed

Table 4: Error Analysis of Baseline vs BERT model

Hu, X. and Downie, J. S. (2010). When lyrics outperform audio for music mood classification: A feature analysis. In *International Society for Music Information Retrieval Conference*.

Hu, Y., Xiaoou, C., and Yang, D. (2009). Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. pages 123–128.

Malheiro, R., Panda, R., Gomes, P., and Paiva, R. P. (2018). Emotionally-relevant features for classification and regression of music lyrics. *IEEE Transactions on Affective Computing*, 9(2):240–254.

Monti, D., Palumbo, E., Rizzo, G., Lisena, P., Troncy, R., Fell, M., and Cabrio, E. (2018). An ensemble approach of recurrent neural networks using pre-trained embeddings for playlist completion. pages 1–6.

Rubtsov, V., Kamenshchikov, M., Valyaev, I., Leksin, V., and Ignatov, D. I. (2018). A hybrid two-stage recommender system for automatic playlist continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018*, RecSys Challenge '18, New York, NY, USA. Association for Computing Machinery.

Russell, J. A. (1980). A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2020). Xlnet: Generalized autoregressive pretraining for language understanding.

Çano, E. (2017). Moodylyrics: A sentiment annotated lyrics dataset. pages 118–124.