Sentiment Analysis from Lyrics

Abstract

This project aims to develop and evaluate a BERT-based sentiment analysis model for lyrics. The results will provide valuable insights into the emotional aspects of music and enable various applications in music recommendation, playlist generation, and cultural analysis.

Introduction

The project's goal is to analyze emotions from music lyrics. This project can be used for many purposes, such as analyzing communities' music habits or new music recommendations. To solve this problem, we used a dataset called "MuSe" [1] and trained a BERT model to predict the emotions of given lyrics. Due to "MuSe" not being suitable for this task, we used additional approaches, such as getting help from the "Genius API" [2], to create our customized dataset. Then, we used the library "Optuna" [3] for hyperparameter tuning. Finally, we implemented inference by utilizing the trained model to make predictions of new lyrics data.

Related work

Several studies have been conducted on emotion prediction from music lyrics.

In the study LyEmoBERT: Classification of lyrics' emotion and recommendation using a pre-trained model, by Revathy V. R et al. [4], the authors developed a multi-class BERT model for the four important human emotions - happy, angry, relaxed, and sad. This work seems similar to ours, but our goal is to predict a single explanatory label rather than 4 generalized emotions. They also mentioned the possibility of using such work for prediction from audio files.

In another study, Sentiment Analysis on lyrics of popular music artists, by Karanveer Singh [5], instead of BERT the author used NLTK's Vader Sentiment analyzer [6] for the analysis. The author only used 3 labels, negative, neutral and positive. We wanted to develop a model that predicts more detailed labels, therefore we took a different approach in terms of labels. But, we got the inspiration of using "Genius API" from this work.

Method

Dataset

In our project, we created our customized dataset because there was no suitable dataset for our goals. To achieve this, we devised the solution of using a baseline dataset, "MuSe" [1]. The dataset contains different combinations of 279 unique mood labels. In this project, we only took the rows that has a single emotion.

79392 unique values	26012 unique values	['sleazy'] 1% ['lazy'] 1% Other (88228) 98%
'Till I Collapse	Eminem	['aggressive']
St. Anger	Metallica	['aggressive']
Speedin'	Rick Ross	['aggressive']
Bamboo Banga	M.I.A.	['aggressive', 'fun', 'sexy', 'energetic']
Die MF Die	Dope	['aggressive']

Figure 1: MuSe Dataset

Muse contains several columns, including the artist's name, track name, and the song's emotions. This dataset is available on "Kaggle" and contains 90.000 songs. Muse did not contain lyrics data, so we came up with the idea of using an API of a website called "Genius.com." We used this API to fetch lyrics using the artists' and their tracks' names. Fetching was the most challenging part of the project because the API fetches data as HTML files and the search for the lyrics data can only be done by the user. Thankfully, we found a library

called "LyricsGenius" [7] that does this process automatically for you.

```
Iterating over row 5 out of 8021
Iterating over row 6 out of 8021
Iterating over row 7 out of 8021
Iterating over row 8 out of 8021
Iterating over row 9 out of 8021
Could not find lyrics for 04-Have You Ever Loved A Woman by Eric Clapton
```

Figure 2: LyricsGenius while executing

	track	artist	seeds	lyrics
0	Now	Wonder Girls	2	i wish i wasn't alive i really hate my life i ...
1	I Want A Little Girl	Solomon Burke	4	ext. north bachman road day (harry awakens in...
2	Pump It	The Black Eyed Peas	1	i don't get breaks, i just make time i don't g...
3	Talk to Your Daughter	Bulletboys	8	mama, papa, please talk to you daughter for me...
4	Oh How It Feels	:Wumpscut:	3	None
...	...	...	...	...
8016	Should Be Loved	Blue October	5	NaN
8017	I Got My Education (Bootleggers Response)	Uncanny Alliance	2	NaN
8018	Kelvin K - Ancestral Moon (Groovenauts Remix)	Kelvin K	7	NaN
8019	I'm Wild About That Thing	Steinski	0	NaN
8020	Hot Show	Prozzäk	5	NaN

8021 rows × 4 columns

Figure 3: Our dataset after applying lyrics

We skipped punctuations and non-ASCII characters in the fetching process. After dropping the rows with Nan values, we got our dataset with 5752 rows. We used the top 10 emotion labels from the final dataset, which are "sleazy," "lazy," "exotic," "martial," "fierce," "organic," "lyrical," "gritty," "erotic," and "technical."

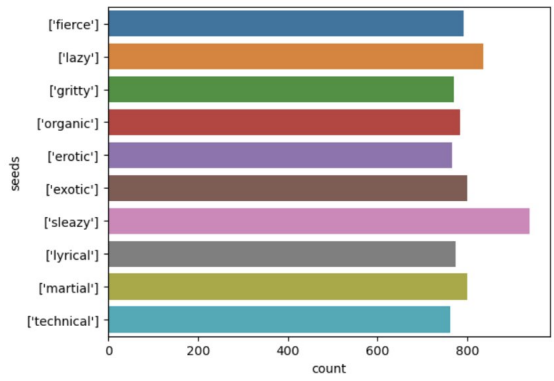

Figure 4: Label counts

Preprocessing

We were unsure about using stopwords for preprocessing since they can affect the context. However, after our trials, we found that it does not affect accuracy, so we decided to use it to make the training faster. Next, we removed numbers and special characters from the lyrics. We also changed the label name from "seeds" to "labels" to make it suitable for the BERT training. After this, we dropped rows that contained non-English lyrics that belonged to other languages. As a result, our total row number dropped to 4844.

After checking if the dataset is balanced, we saw some overrepresentations for some labels. The most used label has 807 rows, while the least has 278 in the dataset. To eliminate this, we used undersampling by equalizing every label to have 278 rows, and we achieved this by dropping random rows from the labels. Next, we dropped the column representing row indexes.

Finally, we used BERT uncased pre-trained auto tokenizer for tokenization. Ultimately, our row number dropped to 2780, and our dataset was ready.

	labels	lyrics
0	2	wish aliv realli hate life realli want die hat...
1	4	ext north bachman road day harri awaken driver...
2	1	get break make time get award get deadlin toke...
3	8	mama papa pleas talk daughter done made love g...
4	3	oh babe wanna lie gonna take give caus know wi...
...	...	...
212	5	tonight sail radio song rescu go stay long tro...
213	2	uh uh uh huh huh oh snap look go slow babi bab...
214	4	surasiu tave bomba yeah choru surasiu tave su ...
215	9	cling level ground balanc abil side posit prog...
216	4	gonna pick could get matter hair mess caus gon...

Figure 5: Final version of our dataset

Training

For training, we split our dataset into 3 parts, training, validation and test. We kept 80 percent of the data for the training set, and kept 10 percent each for both of the validation and test sets.

We used an uncased BERT model for training. We configured the model and added a 20 percent hidden dropout probability. Then, we unfroze all the layers, because our dataset is big enough and we didn't want our model to miss information from the dataset, rather than relying mostly on the pre-trained model.

We got the help of a library called "Optuna" to try different hyperparameter combinations in training. Optuna tries to find the optimum combination by observing test accuracy results instead of trying every combination. Our code tries different hyperparameters; if the new model has the best accuracy in the test split, it gets saved. We were planning to try the following hyperparameters for training:

- Learning rate: 3e-4, 1e-4, 5e-5, 3e-5
- Epochs: from 2 to 10

- Gradient accumulation steps: 1, 2
- Warmup step: from 1e-4 to 1e-2

Unfortunately, Optuna did not work as we expected. It was training with the same hyperparameters in a loop. Therefore, we used the following hyperparameters for training:

- Learning rate: 0.0003
- Epochs: from 2
- Gradient accumulation steps: 2
- Warmup step: 500

We used accuracy from the "evaluate" library of "HuggingFace" as a metric, and Cross Entropy from "sklearn" for loss. We only saved the best model through all epochs. The model had a validation accuracy of around 18-22 percent. We could not understand why, but we will continue to work on it.

[142/142 07:30, Epoch 2/2]

Epoch	Training Loss	Validation Loss	Accuracy
1	2.080800	2.152168	0.222615
2	1.943900	2.184342	0.197880

Figure 6: Training the model

For evaluating the training, we used class-specific precision, recall, f1-score and support on test dataset.

```
Current model is the new best model with accuracy 0.2297

                precision    recall  f1-score   support

    ['erotic']       0.72      0.21      0.33       109
    ['exotic']       0.05      0.04      0.04        23
    ['fierce']       0.00      1.00      0.00         0
    ['gritty']       0.03      0.25      0.05         4
      ['lazy']       0.21      0.40      0.27        15
   ['lyrical']       0.06      0.15      0.09        13
   ['martial']       0.59      0.35      0.44        49
   ['organic']       0.09      0.12      0.10        26
    ['sleazy']       0.09      0.25      0.13         8
['technical']       0.48      0.28      0.35        36

     accuracy                           0.23       283
    macro avg       0.23      0.30      0.18       283
 weighted avg       0.47      0.23      0.28       283
```

Figure 7: Inference Example

From the report, we can see that the model does a good job in predicting the label "Erotic", while it is very bad at predicting "Gritty" and "Lyrical". This may be because of the random splitted test dataset.

"Fierce" has a recall value of 1.00 because we set the recall to be set to 1.00 if there are no examples from this label in the test dataset.

Overall, "Exotic", "Gritty", and "Lyrical" labels had pretty bad scores. While "Erotic", "Martial", "Technical" labels' scores were much better.

Results

Evaluation

We used a confusion matrix to evaluate the model on the test dataset.

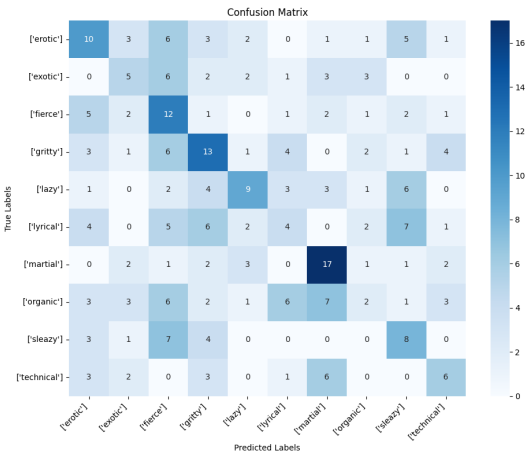

Figure 8: Confusion matrix on the test dataset

The model got 24 percent accuracy on test dataset. The model most accurately predicts the label "Martial". This label is also the label that predicted the most by the model. The model also seems to get confused on the labels "Exotic", "Lyrical", "Organic", compared to other labels.

Inference

We implemented an inference to predict emotions from lyrics data that are not from our training dataset. For the test, we used Mariah Carey's song "You're All I Want For Christmas". After evaluation, the model predicted the song's emotion as "Erotic". This may seem incorrect at first, but the song is about love and desiring someone. Therefore, this prediction is not irrelevant, but "Lyrical" could be more suitable for this song .

```
# Example inference
input_text = 'I dont want a lot for Christmas There is just one thing I need I
predicted_label = classify_sentiment(input_text, tokenizer, model)

Input is: I dont want a lot for Christmas There is just one thing I need I dont
Predicted label is: ['erotic']
```

Figure 9: Inference Example

Discussion

We expected the model to perform better due to applying all these preprocessing and using a pre-trained model. We will learn so much from this project about training once we have better training performance.

In addition, we learned that APIs can help in creating the datasets you desire.

Conclusion

In this project, we aimed to develop a BERT-based sentiment analysis model for music lyrics. We aimed to analyze emotions within song lyrics and explore potential applications in music recommendation, playlist generation, and cultural analysis.

In conclusion, our current BERT-based model did not achieve the desired accuracy. Preprocessing methods and dataset preparation can be reviewed again to increase the accuracy. Number of labels can be increased for more robust inferences, as 10 labels may not be enough in real-life predictions.

The repository of this project is accessible from: https://github.com/mertbkts/Lyrics_Sentiment_Analysis

References

1. MuSe: The Musical Sentiment Dataset https://www.kaggle.com/datasets/cakiki/muse-the-musical-sentiment-dataset https://www.researchgate.net/publication/353100695_MuSe_The_Musical_Sentiment_Dataset

2. Optuna: A hyperparameter optimization framework https://optuna.org

3. Genius API https://docs.genius.com/

4. LyEmoBERT: Classification of lyrics' emotion and recommendation using a pre-trained model https://www.sciencedirect.com/science/article/pii/S1877050923000984

5. Sentiment Analysis on lyrics of popular music artists https://kvsingh.github.io/lyrics-sentiment-analysis.html

6. VADER (Valence Aware Dictionary and sEntiment Reasoner) https://github.com/cjhutto/vaderSentiment

7. LyricsGenius: a Python client for the Genius.com API https://lyricsgenius.readthedocs.io