Music Genre Classification using Song Lyrics

Stanford CS224N Custom Project

Mentor: Megan Leszczynski

Anna Boonyanit
Department of Computer Science
Stanford University
annaboon@stanford.edu

Andrea Dahl
Department of Computer Science
Stanford University
ahdahl@stanford.edu

Abstract

In this project, we aim to classify songs into genres using their lyrics. It is challenging for humans to perform this task, and there often is debate where a song fits since boundaries are not clearly defined and genres are overlapping. After preprocessing our data, we trained our own GloVe embeddings of the song lyrics and created different visualizations to better understand our data. As a baseline, we used our GloVe embeddings in two logistic regression models to classify them into genres. Then, we balanced our dataset so that there was a very similar number of lyrics for each of the genres. Finally, using our GloVe embeddings, we trained an LSTM model and bidirectional LSTM model. Our best LSTM model achieved an accuracy of 68%.

1 Introduction

Music genres organize music into collections by indicating similarities between songs. Often times, songs belong to multiple genres, suggesting that genre isn't always well-defined. Technologies like Spotify have high incentive to automate this categorization process since some estimate they have 60,000 songs added to their site everyday [1]. Music genres could be used to suggest new songs of the same genre to users. Spotify and other music streaming services use metadata such as acoustics and emotional tone for this genre classification [2]. They currently ignore lyrics, since it is challenging to collect large amounts of lyrics. However, utilizing lyrics could be useful in genre classification. Although it would seem that audio files are more useful than song lyrics to classify the genre of a song, due to the high dimensionality of audio data and low dimensionality of lyrics, we aim to see how well we can classify songs into genres using just their lyrics.

Current methods involve classifying genre using musical features as well as feeding lyrics into neural networks. Some of these approaches fail to take into account word order which we think can be really important. This has motivated us to use an LSTM, which performed at an accuracy of 68% for our best model. Current methods also utilize word embedding techniques such as GloVe and Word2Vec. We want to continue using word embedding methods as we think word similarity can be super useful.

2 Related Work

Many papers have focused on classifying music genre using rhythm, timbre, and pitch as opposed to lyrics. Other papers even use techniques such as album customer reviews. A paper which fed in audio features and timbre to their model used a two-layer neural network and achieved an accuracy of up to 39% [3]. It seems like using lyrics could increase the accuracy of the models.

More recently, multiple papers have utilized lyrics for automatic genre classification. When embedding the lyrics, the papers we found used a variety of techniques such as GloVe embeddings, Word2Vec with TFIDF, and Bag-Of-Words with TFIDF. They also then ran their models through various neural networks and achieved a range of accuracies.

Tsaptsinos's paper implemented a hierarchical attention network with an input of lyrics on a 20-genre dataset and achieved accuracies close to 50% [4]. Kumar et al's paper that achieved very high precision of 80.16% fed in lyrics to classify 4 genres using a XGBoost [2]. In Rajanna et al's paper [3], they utilized Word2Vec with TFIDF on a three layer neural network and achieved a high accuracy of 74%. We wanted to build off the neural networks employed and use an LSTM because we think that word order without a limited window size could improve performance.

3 Approach

3.1 Word Embeddings

Simpler techniques like Bag-Of-words fail to capture position in text, semantics, and frequency of co-occurrences. We decided to create GloVe embeddings since that allows us to take in word similarity as opposed to just word count and use our own lyric corpus because the distribution of text for lyrics is different than GloVe's default Wikipedia text.

3.2 Baselines

For our baseline, we decided to implement 2 logistic regression models because they are often used with categorical outputs. We wanted to start with a simpler model to get an understanding of how difficult this task would be and to gauge what kind of performance we expect form our final LSTM. Essentially, a logistic regression model uses the Logistic function to model the conditional probability for each of the outputted classes. For each song, and each genre, this method will return a probability that the song fits in that class. This could be helpful for artists who fit in multiple genres, as we may see that two genres have very close output probabilities using logistic regression.

We used our GloVe embeddings in different ways for the models. For one model, we embedded each word for each lyric using our GloVe embeddings and then concatenated those embeddings together. Because all the lyrics had to be the same length, we padded the lyrics with a pad token (a 100 dimensional vector of 0's). While the longest lyric was 4571 words, we only padded lyrics up to 500 words because the average song length was 278 words. We were worried that the model performance would degrade if we padded up to 4571 tokens, as for many of the lyrics, most of the embedding would be dedicated to just pad tokens. This could lead the model to predict genre based on number of pad tokens. Therefore, for our second model, we averaged all the GloVe vectors for all the words in each lyric.

3.3 Main Approach

In Rajanna's paper [2] , which also performed lyric classification, they found that their three layer deep neural network had the highest accuracy compared to their other non-neural network approaches. We want to build further on their work with neural networks and use an LSTM model because unlike many other neural networks which have a limited window when taking into account prior words, the LSTM has no limit as it can process any length input. The LSTM also uses the same weight matrix regardless of position in text and has a hidden representation that encodes previous words seen. Additionally, we found that a lot of the same words most frequently appeared in pop and rock, so we think that word order will be especially important beyond word count. Then, we also used a bidirectional LSTM because it increases the amount of input information available to the network by running over the input forwards and backward simultaneously.

4 Experiments

4.1 Data

Our dataset from Kaggle [5] contains 160,000 songs in multiple languages from 6 genres and includes its lyrics and its corresponding genre. The artists of some songs are marked under multiple genres. The dataset comes as two different tables, one for artists and their genre and one for songs, their artist, and their lyrics. Our input for the model is the lyrics embedded with our GloVe embeddings and our output is the one-hot-encoded genre.

4.2 Preprocessing

Using part of the code from Kaggle [5], we merged the two tables, dropped all non-English songs (which led to dropping 3 out of 6 genres), and removed duplicate songs. In total there were 7580 songs in our dataset that were labelled under more than one class. In one method, we decided to add multiple genres for lyrics marked under multiple genres, since if we had picked only one genre, it would have been unclear which genre was more accurate for an artist and our model would be less accurate. In one LSTM model, we counted a song genre prediction as correct if the model predicted any of the genres that a song fell under. For the rest of the models, we considered whatever the labeled genre in the test set was as the correct genre. Because Pytorch doesn't take in categorical data for the loss function we are using, we also one-hot-encoded the 3 output genres. Also, after realizing that the model was performing with extremely high accuracy for rock compared to the other 2 genres, possibly because there were 3 times as many rock lyrics as compared to pop lyrics and 2 times as many rock lyrics as compared to hip hop lyrics, we oversampled pop and hip hop to generate new datasets such that the three genres had a roughly equal number of lyrics. Additionally, we tokenized the punctuation so that punctuation would be recognized as its own word when we performed GloVe embedding. Then, we generated our own GloVe embeddings trained on our lyric corpus using code from Stanford NLP's GitHub [6].

We split our dataset into a training, validation, and test dataset (80-20-20 split). The oversampled training set had 123,428 rows.

4.3 Evaluation method

After creating word embeddings using GloVe, we created a visualization of the word vectors of the top 200 most frequently appearing words in the lyrics to gauge if the GloVe embeddings made sense. In order to understand different the words used in the genres were, we plotted commons words from each genre. Beyond calculating accuracy of the model, we also created confusion matrices in order to see which genres were more likely to be confused with one another.

4.4 Experimental details

Using the GloVe vectors generated, we converted the song lyrics into the vectors and padded them appropriately and concatenated them for one model and took the average of the vectors for another model. Very small amounts of this code was based off of Assignment 4. Using the these vectorized lyrics, we implemented a logistic regression model on PyTorch from scratch. We used a learning rate of 0.001, used cross entropy loss and the Adam optimizer and ran both models on 200 epochs. We also coded an LSTM and ran it on a combination of various epochs (we generated training loss graphs to determine when to stop running the model), learning rates, and batch size and ultimately found that our best model used a learning rate of 0.001, batch size of 128, and 60 epochs. Running the LSTM with those parameters took approximately two hours. We used cross entropy loss and the Adam optimizer. For the logistic regression models, we only used our unbalanced dataset and for the LSTM, we used both the unbalanced and balanced dataset. Because some of the songs in the dataset were under multiple genres, we classified the prediction from the model as correct if it predicted any of the genres the song belongs to.

4.5 Results

The GloVe embedding plot (Figure 13) seemed to make sense overall (similar words tended to be near one another). In order to determine which models or word embedding techniques might work best, we also analyzed word counts for each genre and found significant overlap between the 3 genres. Only 27 of the rock words and 10 of the pop words out of the 300 most frequently appearing words for those genres were unique to those genres, suggesting that a bag-of-words embedding or a machine learning model that didn't utilize the order of the words would not be very effective. This led us to want to use GloVe and an LSTM for our final model.

As shown in the table below, the LSTM using the correct genres according to the test label had an equal accuracy with our baseline GloVe Average model (0.64). The GloVe Average model outperformed the GloVe Concatenation model by 0.09 likely because many of the lyrics probably had a high amount of padding which could cause the model to learn things like song length or to ignore

the tokens at the end of a long song. The confusion matricies are quite similar for both baselines; rock clearly performed the best, potentially because there were more rock lyrics in our corpus. Our confusion matrices were more balanced in the LSTM models compared to the logistic regression model. The LSTM models all had a higher accuracy for genres besides rock, suggesting that the logistic regression performance was actually poorer even though the overall accuracies were almost the same. While the rock category's performance went down, the performance of hip hop improved significantly and the performance of pop improved somewhat in all LSTM models, especially the bidirectional model.

We are not surprised that rock performed the best in the models with the unbalanced dataset - because it had the most samples in the dataset, we think our model potentially could have just been predicting rock mostly when it wasn't sure what genre it was.

We expected hip hop to perform slightly better considering that it had the most unique words in its lyrics as shown in figure 8. It also had the longest average lyrics out of the genres (480).

When we expanded our definition of a correct classification to include a prediction of any class that the song belonged to, we also saw an increase in accuracy (0.68) as there were 7580 total songs with multiple genres.

Overall, we would have expected the LSTM models to perform better than the logistic regression model because they can remember information for long periods of time (some lyrics can be several hundreds of words long). We also would have expected the bi-directional LSTM to outperform the normal LSTM because the bi-directional LSTM preserves information from both past and future.

Model	Accuracy
Baseline with GloVe Concatenation	0.55
Baseline with GloVe Average	0.64
LSTM with Unbalanced Dataset	0.64
LSTM with Balanced Dataset	0.57
LSTM with Balanced Dataset and Multiple Correct Genres	0.68
Bi-directional LSTM with Balanced Dataset	0.55

Figure 1: This table documents the accuracy of the models we have implemented.

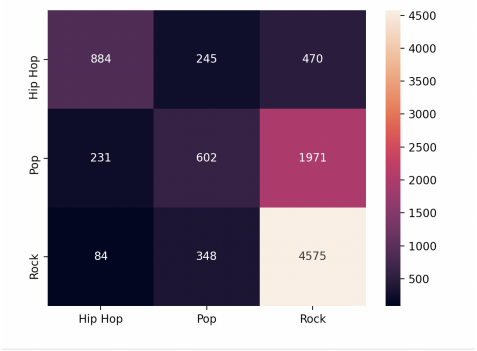

Figure 2: This is a confusion matrix for the logistic regression model using average GloVe embeddings. The rows represent true genre while the columns represent predicted genre.

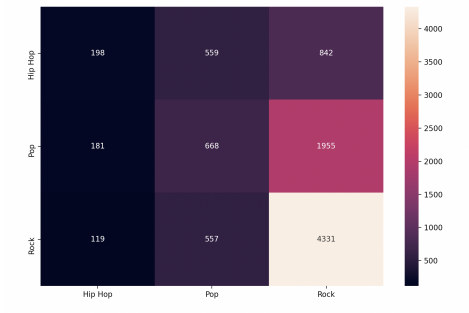

Figure 3: This is a confusion matrix for the logistic regression model using concatenated GloVe embeddings. The rows represent true genre while the columns represent predicted genre.

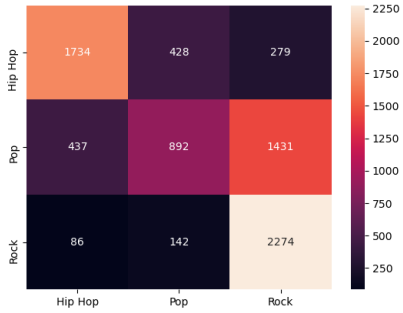

Figure 4: This is a confusion matrix for the LSTM model using an unbalanced (the original) dataset. The rows represent true genre while the columns represent predicted genre.

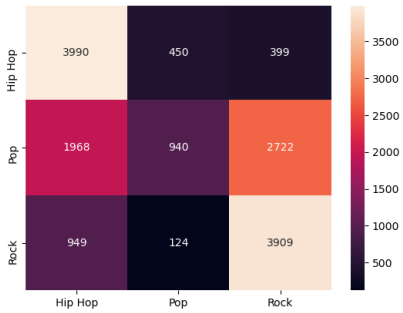

Figure 5: This is a confusion matrix for the LSTM model using a balanced dataset. The rows represent true genre while the columns represent predicted genre.

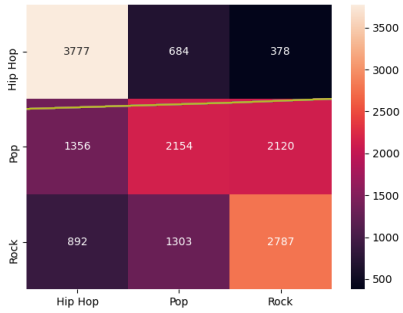

Figure 6: This is a confusion matrix for the bidirectional LSTM model using a balanced dataset. The rows represent true genre while the columns represent predicted genre.

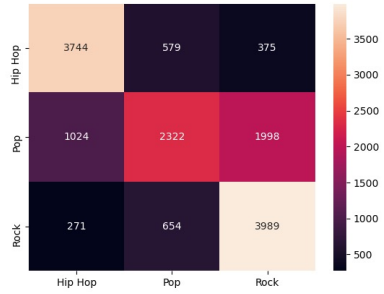

Figure 7: This is a confusion matrix for the LSTM model using a balanced dataset where the prediction was correct if the predicted class fell under any of the song's genres. The rows represent true genre while the columns represent predicted genre.

Per class accuracy					
	Learning Rate	**0.001**	**0.0001**	**0.00001**	**0.000001**
Batch Size	**Batch Size**				
64	**64**	[0.8462, 0.2426, 0.5229]	[0.8171, 0.2462, 0.7732]	[0.8233, 0.2501, 0.7545]	[0.8264, 0.2501, 0.7415]
128	**128**	[0.8215, 0.2497, 0.7788]	[0.8153, 0.2877, 0.7310]	[0.8208, 0.2561, 0.7469]	[0.8246, 0.2554, 0.7445]
Validation accuracy					
	Learning Rate	**0.001**	**0.0001**	**0.00001**	**0.000001**
Batch Size	**Batch Size**				
64	**64**	0.522	0.595	0.592	0.589
128	**128**	0.599	0.596	0.591	0.591

Figure 8: These are the results of our hyperparameter tuning experiments with the LSTM model

Genre	Average Lyric length
All genres	278
Hip Hop	480
Pop	287
Rock	208

Figure 9: This table documents the average lyric lengths for each of the genres.

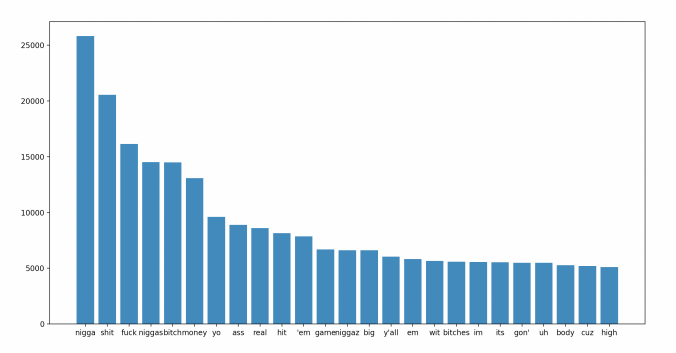

Figure 10: This plots the words and their respective counts in the hip hop genre that are not part of the top 300 words for pop or rock. While there were 97 total unique top words for hip hop, we chose to plot only 25 so that their words would be large enough on the bar graph.

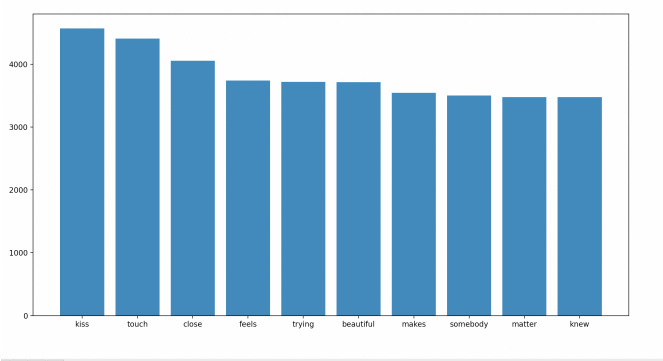

Figure 11: This plots the words and their respective counts in the pop genre that are not part of the top 300 words for hip hop or rock.

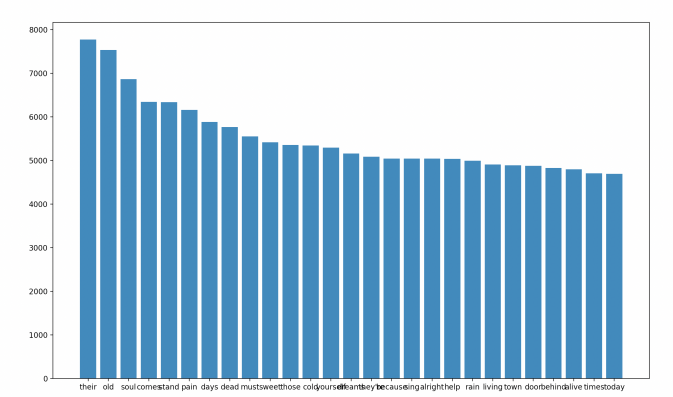

Figure 12: This plots the words and their respective counts in the rock genre that are not part of the top 300 words for pop or hip hop.

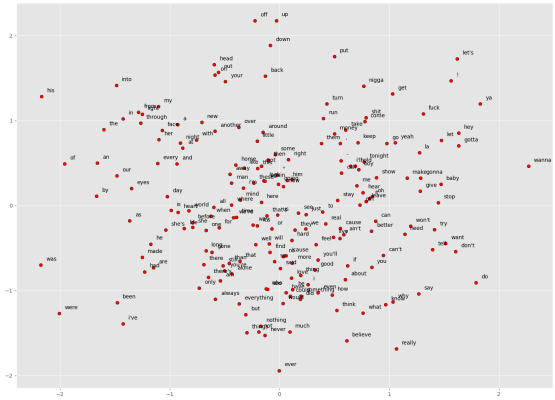

Figure 13: This plots the GloVe vectors that we generated on our lyric corpus for the top 200 most frequently appearing words in our lyrics. The code to generate this plot was based off of code from the 224n website.

5 Analysis

When inspecting our labelled dataset, we noticed multiple instances of hip hop songs that were mis-classified as pop. The data our models were using as "ground truth" had obvious mistakes. When we investigated some of the mis-classifications our model had made, we found several that were the true genre but were misidentified in the database. Our model predicted Time is Money by Akon, Bang it To The Curb by Far East Movement, and Angels by Mac Miller as Hip Hop, which is the genre we determined that these artists belong to, but the dataset labeled them as pop. It is possible that there are a lot more of instances of these misclassifications in the dataset, leading to poorer accuracy for the pop label. As shown in figure 5, there were 1968 songs that were predicted as hip hop but should have been pop. Based on the analysis we did with our outputs, it's likely that many of those songs are actually belong in the hip hop category.

As shown in figure 8, the average length of a song lyric is 278. For our LSTM model with a balanced dataset, we found that the average length of a misprediction was 306 words, suggesting that the model had a slightly harder time predicting genre for longer lyrics. This result is surprising - we would have expected that longer lyrics would have provided the model with more information and led to a better prediction.

6 Conclusion

Our main goal was to most accurately identify the genre of a song based off its lyrics. After preprocessing our data and creating GloVe embeddings, we were able to reach impressive performance with logistic regression baselines. Then, by adding memory and word order to our model in the form of an LSTM, we were able to achieve equal accuracy. When we saw unequal performance among our three classes, we rebalanced our dataset and observed more even distribution of accuracy for pop, rock, and hip hop but a lower accuracy overall. Finally, by adding both backwards memory in addition to forward in a bidirectional LSTM, we were surprised to see a drop in accuracy overall. With additional time, we could investigate why a bidirectional LSTM would have worse performance. Also because our dataset has inaccuracies, it's hard to trust the accuracy scores of our model. If we had time to go through the dataset and fix the mistakes or use a different dataset entirely, it would be interesting to run our models again to see if some of the models have bumps in performance. In the future, we also hope to try running the experiments on already trained GloVe embeddings on a Wikipedia dataset to see if our pre-trained embeddings made a difference.

References

[1] Tim Ingham. Over 60,000 tracks are now uploaded to spotify every day. that's nearly one per second., Mar 2021.

[2] Akshi Kumar, Arjun Rajpal, and Dushyant Rathore. Genre classification using word embeddings and deep learning. *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018.

[3] Arjun Raj Rajanna, Kamelia Aryafar, Ali Shokoufandeh, and Raymond Ptucha. Deep neural networks: A case study for music genre classification. *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 2015.

[4] Alexandros Tsaptsinos. Lyric-based music genre classification using a hierarchical attention network, Jul 2017.

[5] Anderson Neisse. Song lyrics from 6 musical genres, Nov 2019.

[6] Glove: Global vectors for word representation.