Music Genre Classification Based on Lyrics

Wengie Wang
jwang862@wisc.edu

Yezhou Li
yli967@wisc.edu

Abstract

We use machine learning algorithms to classify music genres based on lyrics. By concentrating our research on music lyrics, we hope to understand the importance of lyrics in shaping the type of music we listen to. In this project, we examine several classification algorithms to predict the genre for songs from the Million Song Dataset (MSD) [3] using genre labels extracted from the Last.FM dataset for ground true labels. The models we use include Logistic Regression, Doc2Vec Logistic Regression, Random Forest, Bag-of-Word with Multilayer Perceptron (MLP), and Recurrent Neural Networks (RNNs). Our analysis using the balanced MSD yielded promising results of 54.57% test accuracy using simple Logistic Regression for classification of six genres. We improve the test accuracy to 63.76% using Doc2Vec Logistic Regression model because instead of analyzing independent words, this model extracts lyrical features from sentences and documents. We also try the Random Forest model on our balanced dataset for classification of both six output classes and binary output classes, with test accuracies of 14.86% and 57.91% respectively. Then we try more complex models like MLP and RNN. Though MLP with weighted loss yields 54.71% test accuracy, the loss function for the validation set doesn't converge, indicating an overfitting and random guessing problem. As we tried RNN-LSTM, we encounter the same problem and are not able to get the loss converged. We surmise that the extremely imbalanced dataset, similarities in genre, lack of colloquial vocabulary in the word model, and weak association between lyrics and genre hindered the performance of genre prediction with lyrics.

1. Introduction

With the widespread availability of music online, genre classification has become crucial to help listeners find songs from their favorite genres. Music genres are often used as one of the criteria for ranking songs and sorting music preferences in various charts like Billboard and on music streaming services like Spotify. According to Hypebot [2] , a music industry and techonology news blog, a thousand songs are uploaded to Spotify, Apple, Google Music, Napster, and other streamers every hour, which is 24,000 songs every day and 1 million tracks every six weeks. With this increasingly large number, classifying songs in any given playlists by genres is vital for any music streaming services. In our project, we seek to create a model to classify songs based on only songs' lyrics, excluding additional features, like audios, so we can analyze the role of lyrics in determining a song's genre.

Apart from music streaming industry, music genre-related research has also become an increasingly popular field. Health researchers, for instance, are analyzing how songs of different genres may have therapeutic benefits for patients suffering from neurological related diseases and are examining how music can improve one's concentration [13]. According to Kumar et al., other researchers are seeking to use feature extractions to improve genre classification that can be useful for music streaming sites [10]. Therefore, lyrics and genre-classification are proving to be useful for listeners, streaming services, and current research projects.

2. Related Work

Classification of music genre based on lyrics is a Natural Language Processing (NLP) problem, where lyrics of songs can be regarded as sentences. In recent years, it has been shown that deep neural networks have made a breakthrough in the performance of NLP problems. For example, a new architecture called Dynamic Convolutional Neural Network (DCNN) [9] is developed to model sentences. In the field of Music Information Retrieval (MIR), deep neural networks have made significant improvements in model performances. Specifically, Convolutional Neural Network (CNN) is used to create automatic tagging [6] and musical instrument recognition [11].

For genre classification problem, deep neural networks also have superior performance on audios and lyrics. To be detailed, Sigtia and Dixon [14] proposed three ways to improve feature learning for audio data using neural networks, reporting an accuracy of 83% among 10 genres. Besides, with results obtained by hand-selected features and Support Vector Machines (SVM), Costa et al. [7] compared the

performances of CNN in Music Genre Recognition problems. Moreover, in 2017, Tsaptsinos [15] applied a Hierarchical Attention Network (HAN) on genre classication task on lyrics. Apart from the previous works, Dammann et al. [8] tackled the problem of genre classification of music from three angles: song previews, album artwork, and lyrics. These three features are used to train three models– Recurrent Neural Network, k-Nearest Neighbors and Naive Bayes respectively. Then outputs of these three models are combined together to classify music genres. What is worth mentioning is that the combined model was able to achieve a relatively high accuracy of 91.75% on the test set.

3. Proposed Methods

Lyrics-based classification has to address the problem of Natural Language Processing (NLP) which can be handled by several machine learning algorithms. In this project, we fit several models to classify music genres based on lyrics and compare their performances. First, we use a simple Logistic Regression as a baseline model. Then, we apply a Doc2Vec architecture during the text processing phase and fit a Logistic Regression model to see how much it can improve compared to our baseline model. We also build a Random Forest model to see if this model has better performances. Next, we apply Bag-of-Word model to the lyrics, which characterizes the lyrics by getting the term frequencies, before fitting them to a Multilayer Perceptron (MLP) model. Finally, we train more complex RNN models so that text features can be best absorbed by the models.

Accuracy is the quantitative performance metric we plan to use on our model. Specifically, the accuracy is defined as:

$$Accuracy = \frac{\#Correct\ Predictions}{\#Total\ Predictions}$$

Our main way of visualizing the performance of our model is through the confusion matrix of accuracy, with predicted and true labels (genres) on each axis. To be detailed, the confusion matrix of the prediction accuracy gives us a clear visualization about how the models perform.

3.1. Logistic Regression

We select the multiclass Logistic Regression as the baseline model, setting a benchmark for more complicated models. The multiclass Logistic Regression model calculates the probability of how likely each genre is given the lyrics. After we split the dataset into 70% of training and 30% of test set, we implement feature engineering: We convert the pre-processed lyrics to a matrix of token counts via CountVectorizer and transform the count matrix to a normalized TF-IDF representation using TF-IDF transformer. Then we train our multiclass Logistic Regression model using the Logistic Regression classifier from Scikit-Learn library [5].

3.2. Doc2Vec Logistic Regression

As simple as the regular Logistic Regression is, it doesn't consider the relationship between words or sentences. That is, the previous multiclass Logistic Regression model treats each word independently as a feature instead of analyzing the lyrics as a whole. Considering this limitation, we use a text pre-processing method named Doc2Vec, which captures the relationship between documents and not just words. Specifically, we first label the sentences "Train_i" and "Test_i", where i is the index of the lyrics via Gensim's Doc2Vec using the TaggedDocument method. After we apply Doc2Vec to the entire dataset, we extract vectors from the trained Doc2Vec model. Finally, we train a multiclass Logistic Regression model using the features obtained by Doc2Vec model.

3.3. Random Forest

Known as a flexible and easy-to-use machine learning algorithm that can be applied to both classification and regression tasks, Random Forest can produce good results even without hyper parameter tuning [4]. We build Random Forest models for both binary ("rock" vs. "non-rock") and six output classes. Similar to the procedure for building the multiclass Logistic Regression model, after vectorizing the pre-processed lyrics to a count matrix and transforming the matrix to a normalized TF-IDF representation, we train our Random Forest model using the Random Forest classifier from Scikit-Learn library.

3.4. Bag-of-Word Multilayer Perceptron

As a simple deep network model, Multilayer Perceptron model is built using Bag-of-Word. For the text preprocess part, after removing stop words and converting all lyrics to lowercases, we use spaCy to completely tokenize lyrics. Three methods are used to transform lyrics into Bag-of-Word vectors, which are Binary, Count and Term frequency-inverse document frequency. To train the MLP model, we use 3 fully connected layers with activation functions of relu. Node dropouts are also used to avoid overfitting.

3.5. Recurrent Neural Network

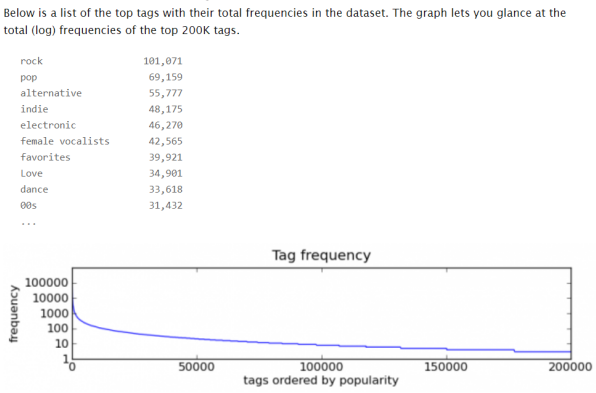
To better classify lyrics by genre, we use a more complicated deep neural network–RNN with LSTM (Table 3). Torchtext is used to tokenize lyrics, to build vocabulary and to pad inputs to equal sizes. We use glove.6B.100d to initiate embedding layers such that words of similar meaning are mapped closely. The RNN-LSTM layer consists of 3 bidirectional-RNN hidden layers. We use 3 linear layers after the RNN layer with 0.5 dropout rate as regularization.

4. Experiments

4.1. Dataset

Data Cleaning We base our research on the data collected from Million Song Dataset (MSD), which is one of the largest music datasets containing music features and listeners' data collected from a million contemporary songs. In particular, we use the MusixXMatch subset, which contains lyrics of specific song-IDs. We also use the Last.FM dataset to predict a corresponding genre-based tag that we use to assess the accuracy of our proposed model. Some of the most popular genres found in the latter subset are illustrated in the following figure (Figure 1).

Figure 1. Music Genres



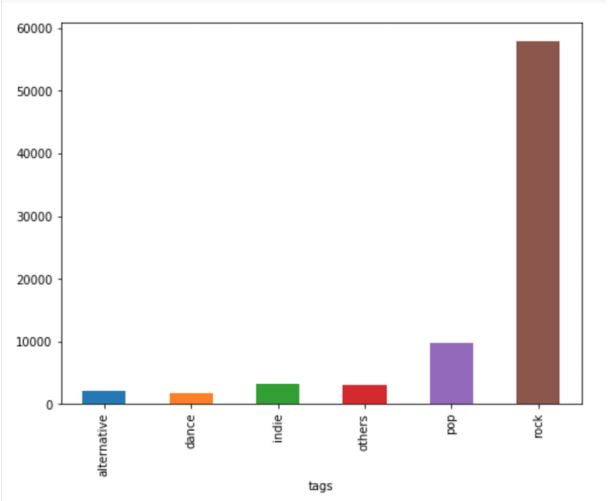
After merging the two subsets, we get a new dataset of one million songs with headers of artists, title, language, lyrics, trackID, and tags (Table 1).

Table 1. Before Cleaning

	artist	title	language	lyrics	track_id	tags
8	Joseph Locke	Goodbye	0.0	NaN	TRMMMHY12903CB53F1	NaN
9	The Sun Harbor's Chorus-Documentary Recordings	Mama_ mama can't you see ?	0.0	NaN	TRCKJCQ128F42810CE	handclaps
10	3 Gars Su'l Sofa	L'antarctique	0.0	NaN	TRPHJJA12903CE133C	strange,francophone,3gs
11	Jorge Negrete	El hijo del pueblo	2.0	Es mi orgullo haber nacido en el barrio mßs hu...	TRBIEKG12903D01567	mexicanisima,Lo Mejor de Jorge Negrete,Vozbril...
12	Danny Diablo	Cold Beer feat. Prince Metropolitan	0.0	NaN	TRRFBRO128F930C00A	NaN
13	Tiger Lou	Pilots	0.0	NaN	TRMMMBW128F4260CAE	getragen
14	Waldemar Bastos	N Gana	0.0	NaN	TRZJLPD128F428B79B	africa,world fusion,angola
15	Lena Philipsson	006	1.0	I had come in the name of love\nWith a mission...	TRMMMKI128F931D80D	swedish,pop

In the language column, "language = 1" indicates the song is English. Since we only focus on English songs, we remove non-English songs and songs with NaN tags and lyrics. At this point, we only have 77,885 songs to work with, which is only around 7.8% of MSD. Originally, in the tag column, the descriptions include several different genres. In order to get a one-to-one song and genre tag, for a specific song, we count the occurrences of the most popular label in MSD ("rock", "pop", "alternative", "indie", "electronic", "electronic", "female vocalists", "favorites", "love", "dance", and "00s") and label the song with the most occurred labels. For example, if the original tag is *"hard rock, rock,classic rock, aerosmith, 70s, guitar, heavy, music, 70s rock, car, hard rock faves,tasty guitar licks and riffs, seen live"*, we check the occurrences of each of the most popular labels—"rock", "pop", "alternative",... etc. and we get the tag that returns the highest number of occurrence. For this song, the tag is "rock", since it returns 12. If there are multiple tags that have the same occurrences, then we use the first label in the list. We then apply this selection procedure to the entire dataset to get a unique label for a specific song. Once we have the unique label for each song, we count the occurrences of each tag ("rock", "pop", "alternative", "indie", "electronic", "electronic", "female vocalists", "favorites", "love", "dance", and "00s"), and get the five most occurred tags and label all other songs as "others". As a result, we get our six output classes as "rock", "pop", "alternative", "dance", and "others". We also created a new column named "category" to map genre tags to numeric values 0-5. However, the new dataset we have is quite imbalanced, with more than 70% of the songs are labeled "rock"(Figue 2).

Figure 2. Genre Distribution after Cleaning



Data Pre-processing To clean the lyrics, we define a function to first decode the HTML file using Beautiful Soup [1], convert the lyrics to lower-case and then strip punctuation and special characters. Then we remove stop words, which are commonly used word (such as "the", "a", and "an"), using a list of stop words via Natural Language Toolkit (NLTK) corpus. At this point, we get a cleaner dataset to work with (Table 2).

Table 2. After Cleaning

	artist	title	language	lyrics	track_id	tags	category
8	The Maytals	Night And Day	1.0	night day night wonder time come night day nig...	TRNGNXD128F92EC53B	rock	1
9	Billy Idol	Scream	1.0	ooh yeah fill cup fill bowl train ready roll e...	TRLBEKJ128F4250887	rock	1
10	Diana Krall	Dancing In The Dark	1.0	dancing dark til tune ends dancing dark soon e...	TRGKFSV128F14808A2	dance	5
11	Joe Diffie	Ships That Don't Come In	1.0	could tell hed tough life way sat stared id co...	TRBFZQH128F42656D9	rock	1
12	Aerosmith	Remember	1.0	aerosmith miscellaneous remember walking san a...	TRBJIDH128F4278108	rock	1
13	Chasen	You And I	1.0	molded everything say lovers code grow closer ...	TRYBUNR12903CCA8C6	rock	1
14	Warren Zevon	The Rest of the Night	1.0	stop lets party rest night seven oclock eight ...	TRQSCXV128F931B428	rock	1
15	Write This Down	Redemption	1.0	weve made costly mistake tonight fed lions den...	TRMMWQF12903CC0CAC	rock	1

Table 3. RNN Architecture

Layer	Detail
Embedding	size=(25_000,100) initiated by glove.6B.100d
RNN-LSTM	size=(100, 256) bidirectional
Dropout	prob=0.5
RNN-LSTM	size=(100, 256) bidirectional
Dropout	prob=0.5
Fully Connected	size=(2*256, 6)

4.2. Logistic Regression

In practice, we use Pipeline class in Schkit-Learn library as a compound classifier to make the *Vectorizer* $\rightarrow$ *Transformer* $\rightarrow$ *Classifier* easier to work with. To train a multiclass Logistic Regression model, we need to specify several parameters, including inverse of regularization strength (`C`), `multi_class`, `solver`, and `class_weight`. We first set the number of We apply L2 regularization to our model because we presume that all input features impact the output. Afterwards, we set `solver` to "sag" since "sag" support L2 regularization. Then we set `multi_class` to "multinomial" since we have six output classes and `class_weight` to "balanced" since our cleaned dataset is quite imbalanced. The only hyper parameter we tune is the inverse regularization strength (`C`), for which smaller values specify stronger regularization. We try different values of `C` from 0.001 to 10^5.

4.3. Doc2Vec Logistic Regression

We create a function to implement Doc2Vec from Gensim to process our lyrics for the cleanned dataset. Since this implementation requires each lyric to have a label associated with it, we label the lyrics for each song with "Train_i" and "Test_i" using TaggedDocument method from Gensim's Doc2Vec module. When training the Doc2Vec model, we specify the following parameters: `dm`, `vector_size`, `negative` (how many "noise words" should be drawn), `min_count`, and `alpha` (initial learning rate). Specifically, we set the dm to 0 since we use distributed bag of words (DBOW), set vector_size to 300 since we use 300 vector dimensional feature vectors, and initialize the learning rate `alpha` to 0.5. Then we draw five "noise words" and ignores all words with total frequency lower than 1 by setting `negative` and `min_count` to 5 and 1 respectively. Then, we initialize the Doc2Vec model and train for 30 epochs. Finally, we implement a Logistic Regression model trained by the Doc2Vec features.

4.4. Random Forest

We build two Random Forest models with six output classes and binary output classes using the Random Forest Classifier imported from Scikit-Learn library. Similar to multiclass Logistic Regression model, we use Pipeline class to combine Count Vectorizer and TF-IDF Transformer and Random Forest Classifer modules. In particular, in Random Forest Classifier, we initialize the number of trees in the forest and the maximum depth of the tree by setting `n_estimator` to 200 and setting `max_depth` to 3. We also set `class_weight` to "balanced" to address the problem of imbalanced dataset.

4.5. Bag-of-Word-Multilayer Perceptron

Despite that our Bag-of-Word MLP first obtains 75% test accuracy, it is achieved only in the first epoch. In the following epochs, training and validation losses remain constant with occasional decreases. Thus, it is reasonable to believe that the model is not learning at all. As we further take a look into the predictions on test data, we notice that the model only predicts the dominant label–"rock". To solve this problem, we use weighted loss in our model. With weighted loss, the model starts to predict all class labels. To find a reasonable model, we try different optimization algorithms like Stochastic Gradient Descent (SGD)and ADAM and implement the MLP model with 2, 3, 4 or more layers. We also train the model by adding stop words and by using different bag-of-word vectors like binary, count and TF-IDF. However, training and validation accuracy keeps wondering around 20% or lower.

4.6. Recurrent Neural Network

Though RNN is a more complicated and usually more reasonable model for text classification, the same problem happens. To be detailed, the RNN-LSTM model gets meaningless test accuracy of 75% without weighted loss and 15% with weighted loss. We implement our model by creating glove embedding, adding bidirectional RNN-layer and more hidden RNN layers, and increasing the dropout rate. Despite all these experiments, we are not able to obtain bet-

ter test accuracy than 15%. (See Table 3 for RNN detailed architecture.)

Preprocess We will use `Torchtext` to handle the preprocess part. Choose tokenizer to be `spaCy`, remove stop words, use `glove.6B.100d`.

Embedding Layer Weight is initialized by **glove.6B.100d**. Words with close meaning are mapped close.

RNN-LSTM Layer With 2 bidirectional-RNN, 0.5 dropout rate as regularization.

Linear Layer This is just a linear layer to transform output of RNN layer to the form we can use in prediction.

4.7. Software

- Python
  - ML: PyTorch, TorchText, Scikit-Learn, Keras, Gensim, nltk, BeautifulSoup
  - Visualization: Matplotlib, Seaborn
  - Others: Pandas, NumPy
  - Environment: Google Colab, JupyterLab

5. Results and Discussion

5.1. Logistic regression

In the Logistic regression model, we achieved 54.57% test accuracy (Confusion matrix in Figure 3). As a relatively simple model, it has several problems. Logistic regression model assumes that the words are independent of each other, which does not take the word order into consideration. This is a strong assumption and might not be appropriate in our dataset, given that lyrics have sequential patterns.

To improve the logistic regression model, we use Doc2Vec preprocess method such that lyrics are treated as sentences instead of independent words. This method improves our test accuracy to 63.73% (Confusion matrix in Figure 4). However, this model has a problem of overfitting. Because, after using L-2 regularization, the test accuracy falls to 14%, which is worse than a random guess. Hence, we conclude that our Doc2Vec Logistic Regression model with L-2 regularization is not learning.

5.2. Random Forest

In the random forest model, the test accuracy is 14.86%. Given the confusion matrix (Figure 5) of this model, we surmise that this model is simply taking random guesses, especially in the ”rock” and ”pop” classes which are the major classes in this dataset. Therefore, random forest in not an appropriate model in the setting.

Figure 3. Confusion Matrix of Regular Logistic Regression

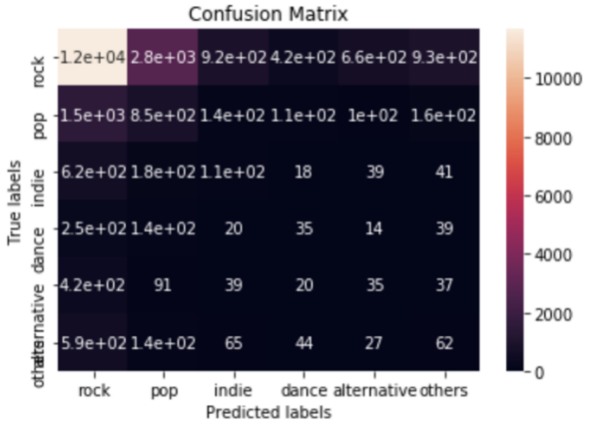


Figure 4. Confusion Matrix of Doc2Vec Logistic Regression

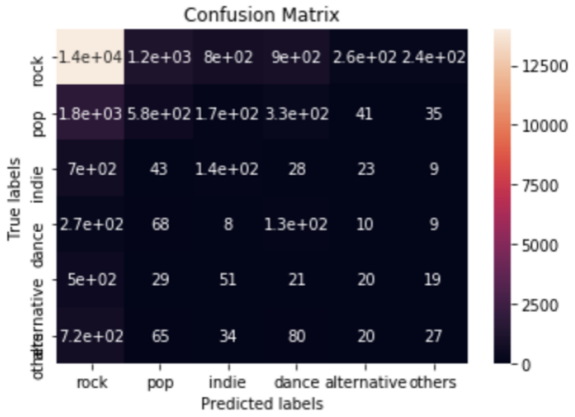
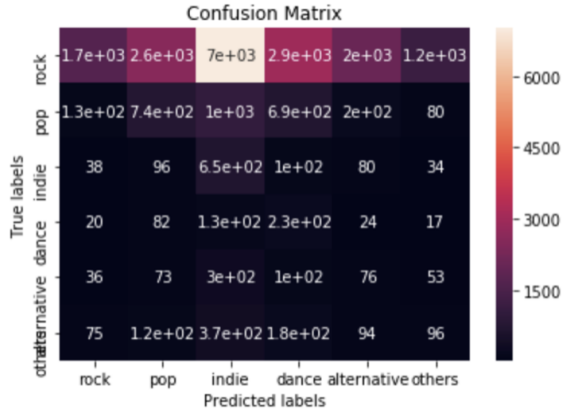


Figure 5. Confusion Matrix of Random Forest



5.3. Bag-of-Word MLP

In the Bag-of-word MLP model, the test accuracy is 75.11%. However, this model is meaningless because it only predicts rock. With weighted loss, the model can predict all class labels. However, the test accuracy is stuck around 15%. We tried to tune all kinds of hyper-parameters,

but the model can not be improved. Therefore, we conclude that this Bag-of-word model can not be trained to perform better than 15%.

Figure 6. Training Validation Loss of RNN with Weighted Loss

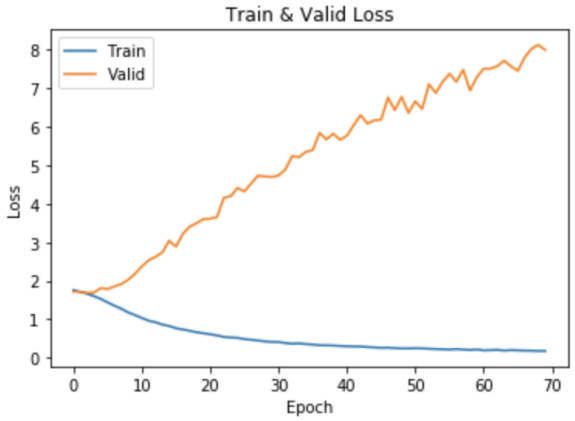
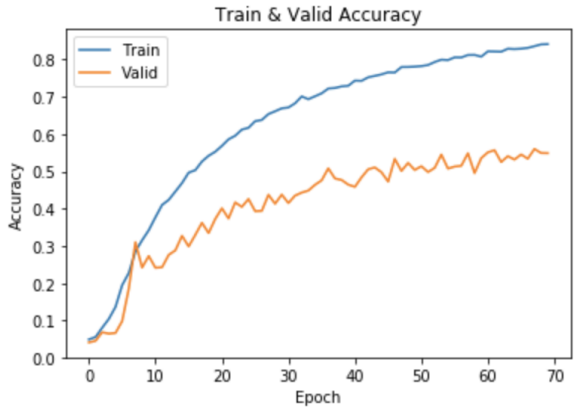


Figure 7. Training Validation Accuracy of RNN with Weighted Loss



5.4. RNN

In our RNN-LSTM model, the test accuracy is 75.11% without weighted loss and 13.12% with wighted loss. Given the loss plot (Figure 6) and accuracy plot (Figure 7), we find that after epoch 50, validation accuracy has the tendency to increase above 50%. This is simply because the model starts to realize that the majority of class label is rock. Moreover, validation loss keeps increasing after epoch 3, indicating that the model is already overfitted. After we try different RNN architectures and hyper-parameters, the loss and accuracy shows the similar pattern. In conclusion, the RNN model can not be trained to perform well on this imbalanced dataset.

The summary of test accuracy of models we developed is shown in Table 4.

Table 4. Model performance

Model	Detail	Test Accuracy
Logistic Regression	Regular	54.57%
	Doc2Vec	63.73%
	Doc2Vec-L2	14%
Random Forest	Regular	14.86%
Bag-of-Word	Regular	75.11%
	Weighted Loss	15.43%
RNN-LSTM	Regular	75.89%
	Weighted Loss	13.12%

6. Conclusions

After we try Logistic Regression, Random Forest, MLP using Bag-of-Word and RNN, we are not able to obtain a good model for this classification task. Our models are either meaninglessly predicting only one class label or useless because the test accuracy is below 15%.

6.1. Limitations

Given that the models we use are able to achieve good performances on similar topics like sentiment analysis, we suppose that there are two main reasons for our low test accuracy. On one hand, our dataset is extremely imbalanced, with more than 70% of the songs are labeled ”rock” in our dataset 2. On the other hand, since we still get poor performances of our model even after we train our models with weighted loss and after we add parameters to balance our dataset, it is reasonable for us to assume that there is no strong association between lyrics and genre. This does not contradicts common sense, considering that the same lyrics can be played by musician of different genre and exhibits different style. Therefore, we argue that more information like soundtrack is needed to classify music genre by lyrics.

Apart from the problem of the imbalanced dataset and the weak association between lyrics and genres, there are some other potential problems of tag cleaning that might affect our bad classification. That is, the tag cleaning process might make it harder for our models to predict. Specifically, as most lyrics in the raw dataset come with multiple tags, we decide to select the tag with highest frequency as class label for each lyrics. In this process, some lyrics might be assigned with a wrong tag. Also, our assumption that genres do not overlap with each other may also impede our models from achieving good performances.

In summary, the main limitation of this project is the quality of dataset. Because of the restriction in copy right, we can not obtain better dataset with both lyrics and accurate information about corresponding genres.

6.2. Future Work

To train a better model, the first priority is to collect a better dataset of lyrics and genre. We would also like

to try different models like Hierarchical Attention Network(HAN) [16]. This model has an advantage to minor the hierarchical structure of lyrics and would probably perform better. Another idea is to classify music genre by both lyrics and audio [12] [8], given the fact that lyrics and audio together may better determine the genre of a song.

7. Acknowledgements

We would like to express our great appreciation to our professor Dr. Sebastian Raschka for merging the subsets of the original dataset. Advice given by him has also been a great help in building our models.

8. Contributions

Wengie Wang cleaned the dataset from one million songs to 77,885 songs and pre-processed the tags and lyrics. She coded the multiclass Logistic Regression model, Doc2Vec Logistic Regression model and Random Forest model. She did data visualizations, organized the presentation, interpreted the results and wrote the final report.

Yezhou Li wrote codes for BOW-MLP model and RNN models. He was involved in finding related work and methods for our project, interpreting the results, comparing performances of our models, and figuring out future improvements and working on the project report.

References

[1] Beautiful soup. `https://www.crummy.com/software/BeautifulSoup/bs4/doc/`.

[2] Blog of hypebot. `https://www.hypebot.com/hypebot/2018/06/24000-tracks-uploaded-to-music-streamers-every-24-hours.html`.

[3] Million song dataset. `http://millionsongdataset.com/`.

[4] The random forest algorithm. `https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd`.

[5] Scikit learn. `https://scikit-learn.org/stable/`.

[6] K. Choi, G. Fazekas, and M. Sandler. Automatic tagging using deep convolutional neural networks. *arXiv preprint arXiv:1606.00298*, 2016.

[7] Y. M. Costa, L. S. Oliveira, and C. N. Silla Jr. An evaluation of convolutional neural networks for music classification using spectrograms. *Applied soft computing*, 52:28–38, 2017.

[8] T. Dammann and K. Haugh. Genre classification of spotify songs using lyrics, audio previews, and album artwork.

[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[10] A. Kumar, A. Rajpal, and D. Rathore. Genre classification using feature extraction and deep learning techniques. In *2018 10th International Conference on Knowledge and Systems Engineering (KSE)*, pages 175–180. IEEE, 2018.

[11] V. Lostanlen and C.-E. Cella. Deep convolutional networks on the pitch spiral for musical instrument recognition. *arXiv preprint arXiv:1605.06644*, 2016.

[12] R. Mayer, R. Neumayer, and A. Rauber. Combination of audio and lyrics features for genre classification in digital audio collections. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 159–168. ACM, 2008.

[13] C. O'Callaghan and D. Grocke. Lyric analysis research in music therapy: Rationales, methods and representations. *The Arts in Psychotherapy*, 36(5):320–328, 2009.

[14] S. Sigtia and S. Dixon. Improved music feature learning with deep neural networks. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6959–6963. IEEE, 2014.

[15] A. Tsaptsinos. Lyrics-based music genre classification using a hierarchical attention network. *arXiv preprint arXiv:1707.04678*, 2017.

[16] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.