IDA - Department of Computer and Information Science
Linköping University

Performance Comparison for Lyrics Classification in Traditional and Deep Learning Methods

732A92 - Text Mining - Project Report

Zuxiang Li

LiU ID: zuxli371

Linköping, Sweden

March 16,2021

Abstract

Text classification is an important part of Natural Language Processing(NLP). Various models, including traditional machine learning methods and new deep learning methods, have been discovered to help people speed up development in NLP both in research and industry. In this report, such models were applied to music's genre classification based on lyrics and generate prediction results. We analyze performance based on accuracy, model complexity, fitting time. The result showed randomforest had the best performance in accuracy, around 63%, but not in fitting time and model complexity, SVM balanced accuracy, and fitting time. Deep learning methods didn't reach an expected outcome, accuracy reached around 50%. The analysis has its limitation due to limited computational power and time.

Content

1 Introduction

Music has an important role in people's daily life, people in a different culture, ages have a various taste of music, such as Hip hop, Rock, Country, Pop, R&B and extra. Nowadays the database of music continues growing, people spend more time and effort than before to discover the music they might love, therefore, basically all music platform presented their music recommendation solution to meet user's expectations. To understand the user's preference on music type, the recommendation system requires all music information that users have played before, such as vocals, lyrics and genre. If the user played a lot of rock music before, the system would try to learn that the user loves the rock genre, then it might recommend more rock music to the user. Therefore, the genre of a song is important information to build a successful recommendation system. However, the genre still needed to be marked manually, in addition to that, the information of a song might not be complete, it could be missing from the beginning. Thus, automatically classifying music genres using lyrics could be a way to solve it. Similar research has been done before, Lyric TextMining in Music Mood Classification. American music, 2009.[4]

2 Theory

2. 1 Decision tree

The idea of decision trees is to split the domain of feature set into different hypercubes and define the target value to be constant within each hypercube. A decision tree is assembled by nodes, which brings the problem of the optimal tree length, if the tree is too large the result would be overfitting, if the tree is too small then it would be underfitting, so find an optimal tree length is also part of our mission.

2.2 Random forest

Implemented by using random forest module from sklearn, as a bagging method, it generates a number of trees instead of a single tree. The training procedure showed below.

1. Draw n_{trees} bootstrap samples.
2. Build an unpruned tree for each sample we obtained in step1. Sample m_{try} predictors and select the best split.
3. Aggregate the prediction of n_{trees}, in this project use majority voting.

2.3 SVM

Support vector machine could be an ideal model for our problem. Comparing to other problems, lyrics classifying has a high dimensional input space, which SVM is immune to that. Different Kernels may have an impact on our result, so it is part of the job to find out which kernel is the most suitable for this project..

Linear	$K(x_i, x_j) = x_i^T x_j$
Polynomial	$K(x_i, x_j) = (\gamma x_i^Y x_j + r)^d, d > 1$
RBF	$K(x_i, x_j) = exp\left(-\gamma \left\|\left\|x_i - x_j\right\|\right\|^2\right), \gamma > 0$
Sigmoid	$K(x_i, x_j) = tanh(-\gamma\, x_i^T x_j + r), \gamma > 0, r < 0$

Table 1 Common Kernels

In this project, we will deal with multiple class classification. One possible approach is one-against-all method. Consider M classes with N training samples,$\{x_1, y_1, \dots, x_n, y_n\}$, x_i is M-dimensional vector, y_i is the class label, M here corresponds to M-classes. This approach build M SVM classifiers, each classifier classify it as one class or not. The procedure showed as follow.[7]

The decision function

$$f_i(x) = w_i^T \phi(x) + b_i$$

First, we minimize the following equation,

$$L(w_i, \xi_j^i) = \frac{1}{2}\left||w_i|\right|^2 + C\sum_{j=1}^{N} \xi_j^i$$

With subject to

$$\tilde{y_j}\left(w_i^T \phi(x_j) + b_i\right) \geq 1 - \xi_j^i, \xi_j^i \geq 0$$

Where $\tilde{y_j} = 1$ if $y_j = i$ and $\tilde{y_j} = -1$ otherwise.

A sample x is classified as in

$$i^* = argmax_{i=1,\dots,M}\, f_i(x) = argmax_{i=1,\dots,M}(w_i^T \phi(x) + b_i)$$

2.4 LSTM

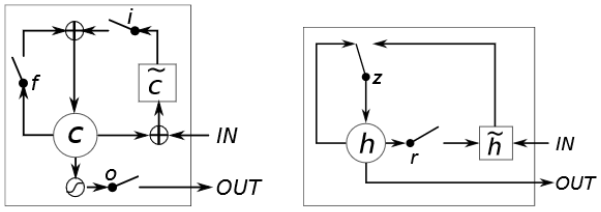
Introduced by S. Hochreiter and J. Schmidhuber in 1997. Long short-term memory (LSTM), a traditional neural network cannot work with data have persistence, especially like text data, to understand every word in a sentence you need to understand the meaning of previous words, hence, RNN was introduced to solve this problem, however, it still has its flaws, as the information continues to feed into RNN model, it becomes harder to learn. LSTM solved this challenge by designing an explicit architecture, a cell that contains INPUT, FORGET and READ gate.[1]

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

$x_t \in \mathbb{R}^{\boldsymbol{d}}$	input vector to the LSTM unit
$f_t \in \mathbb{R}^{\boldsymbol{h}}$	forget gate's activation vector
$i_t \in \mathbb{R}^{\boldsymbol{h}}$	input/update gate's activation vector
$o_t \in \mathbb{R}^{\boldsymbol{h}}$	output gate's activation vector
$h_t \in \mathbb{R}^{\boldsymbol{h}}$	hidden state vector also known as output vector of the LSTM unit
$\widetilde{c_t} \in \mathbb{R}^{\boldsymbol{h}}$	cell input activation vector
$c_t \in \mathbb{R}^{\boldsymbol{h}}$	cell state vector
$W \in \mathbb{R}^{(\boldsymbol{h}*\boldsymbol{d})}, \boldsymbol{U} \in \mathbb{R}^{(\boldsymbol{h}*\boldsymbol{h})}, b \in \mathbb{R}^{\boldsymbol{h}}$	weight matrices and bias vector parameters which need to be learned during training

2.5 GRU

Gated Recurrent Unit (GRU) was proposed by Cho et al. [2014]. It is a mutation of LSTM, it combined FOTGET gate and INPUT gate into a single UPDATE gate.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

From above it is clear to see the similarity between LSTM and GRU, but it is still difficult to tell which model perform better.[5]

2.6 TextCNN

TextCNN[8] is a simple model achieves excellent result on multiple benchmarks. A n words sentence can be represented as

$$X_{1:n} = X1 \oplus X2 \oplus \ldots \oplus X_n$$

By applying filter $w \in \mathbb{R}^{hk}$, new features can be produced from a window of words $X_{i:i+h-1}$ by

$$c_i = f(w \times X_{i+i+h-1} + b)$$

Each window in the sentence will have a filter to create a feature map

$$c = [c_1, c_2, \ldots, c_{n-h+1}]$$

Next step is applying a max-over-time pooling and take the maximum value as feature for this filter. Multiple features require multiple filters, once the features got extracted, it will be passed to next layer and softmax layer to produce output result.

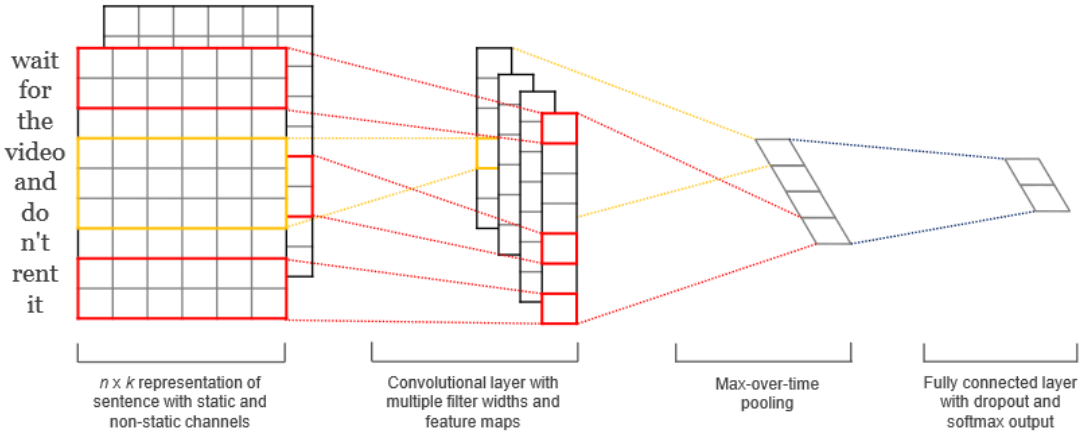


Figure 1 Model architecture Example[8]

2.7 Bag of words and TFIDF

Bag of words model is usually used in text classification; it takes the frequency of each word in the text as a feature to train the model. In this project, the lyrics of each song are represented as the bag of its words. After we create the BOW model, we apply the TFIDF scheme.

3 Data

3.1 Raw Data

In this report, song data was scarped from the Internet using a spider program. In playback.fm, it provided top100 music charts from 1900 to 2015 with four kinds of music including rock&roll, country, R&B. To add one more class as our target, I have collected hip-hop songs' data from billboard charts including from 2002 to 2020. I scarped song's information containing artist, song's name, release year, genre, in total 13143 songs retrieved using spider program. To get lyrics for saved songs, I used Genius as lyrics search engine, Genius provided full lyrics for most of the songs that were saved. Using API from Genuis.com, more than 11000 songs have successfully retrieved lyrics from it.

3.2 Data Cleaning

After merging the data collected from the Internet, a data frame containing song name, artist name, released date, lyrics, and genre was generated and saved for later use. In preprocessing, first, some special characters and musical terms were eliminated using regular expression, for example, [Verse], [Chorus],[Instrumental] and etc. Then remove common stop words. Then we got the cleaned lyrics.

3.3 Data Example

Lyrics	Genre
Inseparable Inseparable wonderful know Incredible Incredible bring woman style love…	rnb
go to shout uhh go to go to go to go to people people audience people sit mind…	rock
think young Got life ahead Hell kid go raise dream go smoke ditch town hangin…	country
help wait Oh help wait Check uh be right trip stand lie turn right round forgive face…	hiphop

Table 2 Data Example

4 Method

In traditional methods, first, build our classifiers using built-in classes in sklearn. To prove that our classifiers are working properly, we build a dummy classifier using most frequent scheme. Sklearn provided a simple way to interact with vectorizer and transformer. Using class countvectorizor and tfidftransformer, we can easily translate our data set into a normalized tf-idf matrix. Then we can build a pipeline combining vectorizer, transformer, and classifiers. Pipeline sequentially assembles a list of steps in data vectorizing and transforming. It enables a much more convenient way for parameter tuning in later steps. To get the maximized use of classifiers, we create a grid search for each classifier, using GridSearch class in sklearn, for each classifier, we set different parameter values then try to find out the best combination among all of them.

For SVM, we tuning parameters for kernels and C values. C values must be strictly positive hence we set our range from 1 to 10 to search for the best C values

Parameter	Value
C value	1,2,3,4,5,6,7,8,9,10
Kernel	RBF, Sigmoid, Linear, Poly

The step between C values is still quite big, therefore after we locate the best C value above, we will shrink the gap to find out a more precise best C value.

In decision tree, we compare different criterion in Gini and entropy, with max depth from 3 to 10, another one we will run our model without max depth limitation.

In randomforest, we use Gini as our default criterion, to get the best parameters, we will try some parameters, including max_depth, n_estimators, these two parameters controls the number of trees in the forest and the maximum depth of the tree, we will set the parameters to 8 to 48 and 10 to 180, another experiment we will set max depth to none so that nodes will expand until all leaves are pure or until all leaves contain less than min samples split samples.

Parameter	Value
Max depth	8,16,24,32,40,48

n_estimators	10-180

In deep learning methods, we tried different models including LSTM, GRU, and TextCNN. First, we split our dataset into train and test set with a test size equal to 0.2, so that 80% of our all data would be used for training and the rest for testing. Then we use tokenizer from keras to vectorize our text corpus, translate each lyric into a sequence of numbers. Since we have 4 genres to predict so our problem is a categorical classification with 4 targets. To encode our label for train and test, we used labelencoder to transformer our labels into numerical targets, then use onehotencoder to encode it to an array of 1 and 0 so that our model would take it in as a parameter.

The next step is building our embedding matrix, we train a 100-dimensional glove word embedding using our lyrics corpus.
Each model has its own structure, the detailed information is attached in Appendix.

5 Results

First, we will have a look at the distribution of different genres in our data set. Then we run the majority baseline, we can see that the accuracy is 30% in Figure 2.

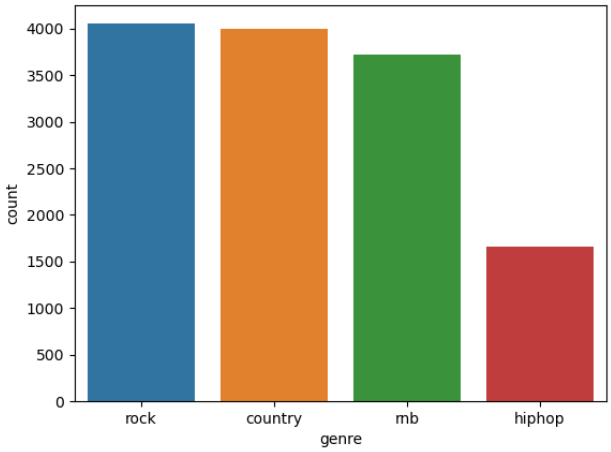


Figure 2 Genre distributions

Then we dive into different traditional methods, we run 5-fold cross-validation for each model in parameter tuning.
The first model is SVM, the results show below, we can see that rbf kernel performed the best among all kernels, to retrieve a more precise C value, we fixed rbf kernel and shrink the range, and discovered that 1.4 is the optimal C value.

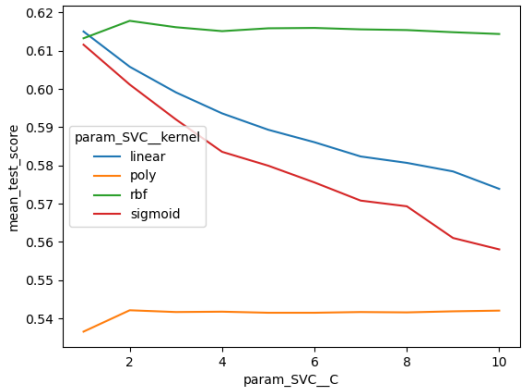


Figure 3 Grid search with kernel and C

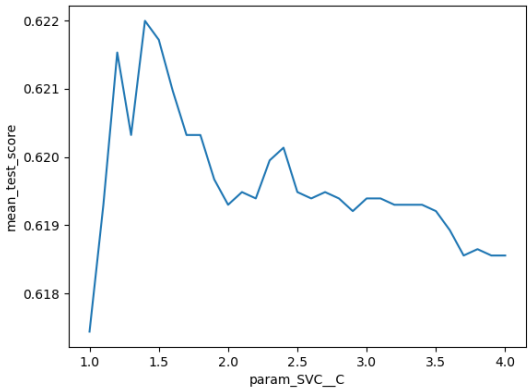


Figure 4 2nd search

The next model is decision tree, we set max depth and criterion as variable, at max depth equal to 9 with Gini criterion it reached maximal mean test score. We also have tested max depth set to none, the result got worse. Hence the optimal parameter for decision tree is depth 9 with gini criterion.

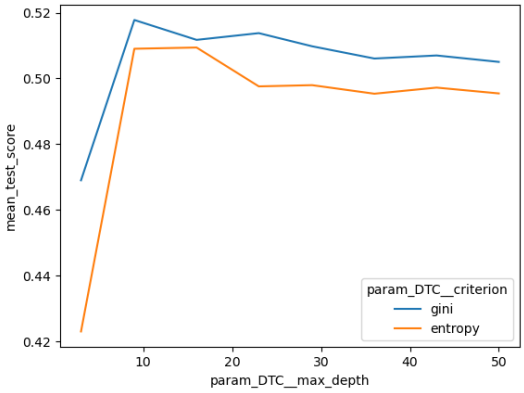


Figure 5 Decision tree grid search

The third model is Randomforest, In the first experiment, we have two parameters for tuning, max depth, and number of estimators. We can observe that as the number of estimators increases, the test score increases until it reaches around 80 it becomes stable. Then we have a look at the max depth, the higher max depth, the higher the test score.

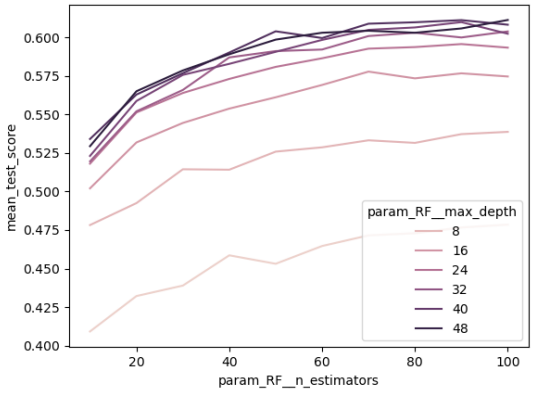


Figure 6 Randomforest grid search

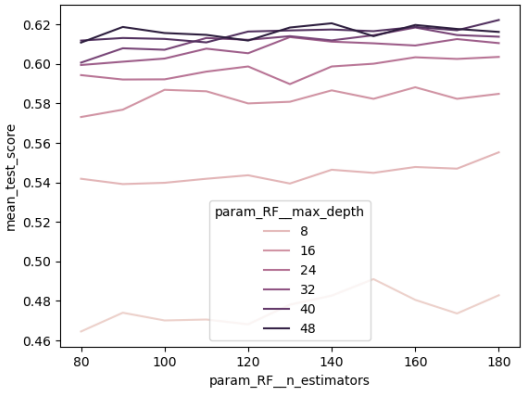


Figure 7 Randomforest grid search

Second experiment we will fix max depth to none, set number of estimators as variable, it reached 0.6325 at 800 estimators at Figure 8.

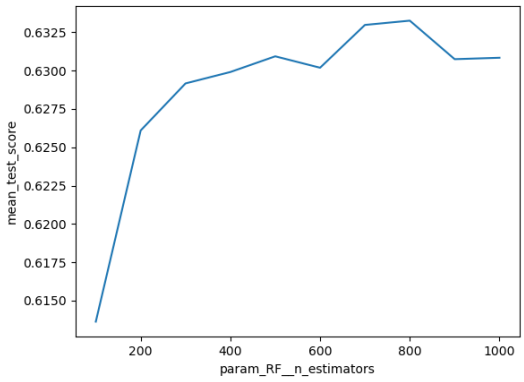


Figure 8 Randomforest

Next, we will have a look at the detailed score in searching for the optimal parameter in cross-validation. We can see that randomforest obtained the highest test score among the other classifiers, while decision tree only obtained 0.51. As for time-consuming in fitting model, decision tree has the lowest fit time, while randomforest took more than 3500 seconds to train the model.

model	mean_fit _time	split0_test _score	split1_test _score	split2_test _score	split3_test _score	split4_test _score	mean_test _score
randomf orest	3511.46	0.63907	0.627734	0.638436	0.627268	0.633783	0.633258
decision tree	8.255802	0.504651	0.524895	0.53141	0.506747	0.521173	0.517775
svm	140.3636	0.632558	0.612378	0.603071	0.616566	0.624477	0.61781

Table 3 Detailed Result

Next, we move on to the deep learning classification methods result, for each classifier, we trained it with 32 batch size and 30 epochs, with accuracy and loss plotted for each classifier in training and testing, we can see that for TextCNN the model performed poorly with only 50% accuracy on the test set and it begins to overfitting around 3rd epochs, for LSTM and

GRU, both performed slightly better than TextCNN, the maximal accuracy on the test set is around 60%, after 10th epochs it starts to overfitting.

	Rock	Hiphop	Country	R&B
Original Data	30.41%	12.13%	29.10%	28.36%
SVM	31.48%	8.49%	30.18%	29.85%
Decision Tree	30.37%	13.21%	30.11%	26.31%
Randomforest	28.88%	8.56%	35.13%	27.43%

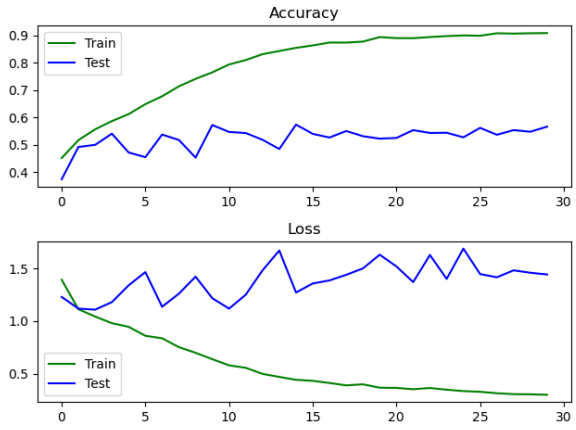


Figure 9 TextCNN

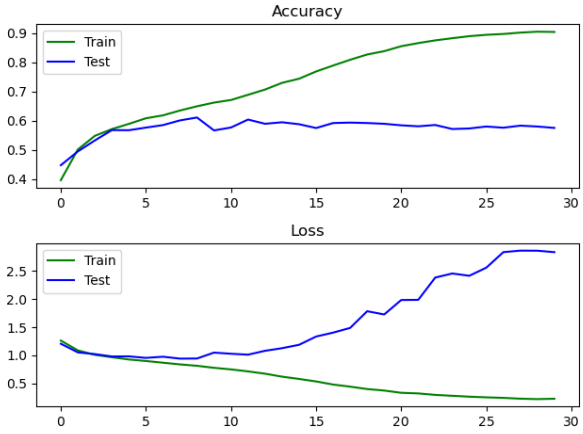


Figure 10 GRU

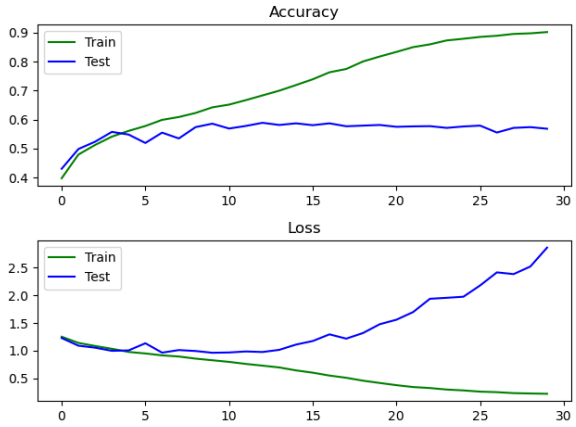
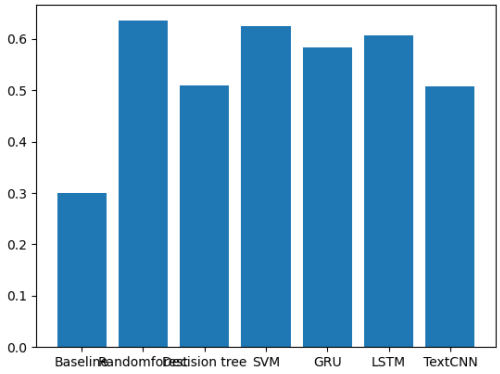


Figure 11 LSTM

Next, we have a look at the predicted class distribution of deep learning methods, judging by the first look, all classifier except LSTM performed poorly on prediction.

	Rock	Hiphop	Country	R&B
Original Data	30.41%	12.13%	29.10%	28.36%
LSTM	24.00%	10.31%	30.89%	34.80%
GRU	24.15%	2.79%	30.82%	42.24%
TextCNN	12.77%	13.51%	26.24%	47.49%

After retrieving prediction on all classifiers we have tested, we calculated the accuracy for all of them, all classifiers' accuracy have passed baseline, the highest among them all is randomforest.



6 Discussion

First, we will focus on the result of traditional methods, SVM had the second-best accuracy among all classifiers, using RBF kernel and 1.4 as C value, SVM has its natural advantage on text classification, SVM adapt well on high dimensional data.

Decision tree had the lowest accuracy in our results, only 50% of accuracy on the test set. Meanwhile, randomforest scored the highest in our experiment, as an upgrade version of decision tree, it is expected to have a better performance than itself, and it did so. We had an ideal result on adding more trees in the randomforest. The reason of difference on the two models, because as a bagging classifier, randomforest takes the advantage of multiple decision trees, so it doesn't rely on a single feature in a single tree, randomforest select features randomly during the training process, hence it doesn't rely on any specific set of features. However, it also has its drawback, it requires more training time and computational resource than other methods, in our result, though it has the highest score among all classifiers, comparing to SVM, to improve 2% of accuracy, randomforest took 25 times of fitting time than SVM, it might be unnecessary to do so if time and resource are limited. Comparing with related work that has done by Ariel Amar Hanan Benzion Elad Gershon Ruth Taboada in AI Genre Classification by Lyrics[2], we both obtained similar results, that the lower complexity model behaves fast but with low accuracy, vice versa.

The second part will focus on deep learning methods, in general, three classifiers did not perform well on our dataset, LSTM outperformed the other two classifiers, though they had a better model structure and more parameters to train, which is quite counterintuitive. In addition to that, in prediction distribution over 4 classes, LSTM also had a better performance than the other two models. The simpler model in lyrics classification has its advantage, the research from Alexandros Tsaptsinos in Music Genre Classification by Lyrics using a Hierarchical Attention Network[3] also proved that.

In this project, we have successfully tested and compared different classifiers' performance on lyrics classification. However, we noticed that the dataset downloaded from the Internet is minor imbalanced. The number of hip hop lyrics is around half of the other genres. If possible, more hip hop data should be added into our dataset. In addition to that, add more genres could also affect our result. As shown in previous research[3], an increasing number of the genre

could result in worse performance. The computer's performance was another limitation for this project, each deep learning model requires at least two hours to fit, therefore I don't have enough time and computational resource to do hyperparameter tuning in this part.

Conclusion

Though deep learning is very popular now, it has its limitation and applicable scenarios. In this project, traditional text classification methods outperformed deep learning methods in lyrics classification. Especially, SVM might be the best candidate for text classification due to its own strength. Deep learning methods still have their potential, but to obtain a better performance it may require more effort on hyperparameter tuning, comparing to traditional methods, not only more time but also more computational resource needed.

Appendix

Code Link: https://github.com/zuxiangli/Text_mining_project

LSTM		
Layer	Output Shape	Param
embedding (Embedding)	(None, 3000, 128)	76928
dense (Dense)	(None, 3000, 128)	16512
gru (GRUlstm (LSTM))	(None, 256)	394240
dense_1 (Dense)	(None, 128)	32896
flatten (Flatten)	(None, 128)	0
dropout (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 4)	516
Total params: 521,092 Trainable params: 521,092 Non-trainable params: 0		

Table 4 LSTM Structure

GRU		
Layer	Output Shape	Param
embedding (Embedding)	(None, 3000, 128)	76928
dense (Dense)	(None, 3000, 128)	16512
gru (GRU)	(None, 256)	296448
dense_1 (Dense)	(None, 128)	32896
flatten (Flatten)	(None, 128)	0
dropout (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 4)	516

Total params: 521,092 Trainable params: 521,092 Non-trainable params: 0	

Table 5 GRU Structure

TextCNN			
Layer	Output Shape	Param	Connected to
input_1 (InputLayer)	[(None, 3000)]	0	
embedding (Embedding)	(None, 3000, 100)	9497100	input_1[0][0]
conv1d (Conv1D)	(None, 3000, 256)	77056	embedding[0][0]
conv1d_2 (Conv1D)	(None, 3000, 256)	102656	embedding[0][0]
conv1d_4 (Conv1D)	(None, 3000, 256)	128256	embedding[0][0]
batch_normalization (BatchNorma	(None, 3000, 256)	1024	conv1d[0][0]
batch_normalization_2 (BatchNor	(None, 3000, 256)	1024	conv1d_2[0][0]
batch_normalization_4 (BatchNor	(None, 3000, 256)	1024	conv1d_4[0][0]
activation (Activation)	(None, 3000, 256)	0	batch_normalization[0][0]
activation_2 (Activation)	(None, 3000, 256)	0	batch_normalization_2[0][0]
activation_4 (Activation)	(None, 3000, 256)	0	batch_normalization_4[0][0]
conv1d_1 (Conv1D)	(None, 3000, 128)	98432	activation[0][0]
conv1d_3 (Conv1D)	(None, 3000, 128)	131200	activation_2[0][0]
conv1d_5 (Conv1D)	(None, 3000, 128)	163968	activation_4[0][0]
batch_normalization_1 (BatchNor	(None, 3000, 128)	512	conv1d_1[0][0]
batch_normalization_3 (BatchNor	(None, 3000, 128)	512	conv1d_3[0][0]
batch_normalization_5 (BatchNor	(None, 3000, 128)	512	conv1d_5[0][0]
activation_1 (Activation)	(None, 3000, 128)	0	batch_normalization_1[0][0]
activation_3 (Activation)	(None, 3000, 128)	0	batch_normalization_3[0][0]
activation_5 (Activation)	(None, 3000, 128)	0	batch_normalization_5[0][0]
max_pooling1d (MaxPooling1D)	(None, 750, 128)	0	activation_1[0][0]
max_pooling1d_1 (MaxPooling1D)	(None, 750, 128)	0	activation_3[0][0]
max_pooling1d_2 (MaxPooling1D)	(None, 750, 128)	0	activation_5[0][0]
concatenate (Concatenate)	(None, 2250, 128)	0	max_pooling1d[0][0],max_pooling1d_1[0][0],max_pooling1d_2[0][0]
flatten (Flatten)	(None, 288000)	0	concatenate[0][0]

dropout (Dropout)	(None, 288000)	0	flatten[0][0]
dense (Dense)	(None, 512)	147456512	dropout[0][0]
batch_normalization_6 (BatchNor	(None, 512)	2048	dense[0][0]
dense_1 (Dense)	(None, 4)	2052	batch_normalization_6[0][0]
Total params: 157,663,888 Trainable params: 148,163,460 Non-trainable params: 9,500,428			

Table 6 TextCNN Structure

References

[1] Long short-term memory,Wikipedia,https://en.wikipedia.org/wiki/Long_short-term_memory

[2] AI Genre Classification by Lyric, Ariel Amar Hanan Benzion Elad Gershon Ruth Taboada, https://www.cs.huji.ac.il/~ai/projects/2017/decision_trees/AI_Genre_Classification_By_Lyrics/ai-genre-classification.pdf

[3] Alexandros Tsaptsinos, Music Genre Classification by Lyrics using a Hierarchical Attention Network,https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2728368.pdf

[4] Xiao Hu, J. Stephen Downie, and Andreas F. Ehmann Lyric TextMining in Music Mood Classification. American music, 2009.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho,Yoshua Bengio, Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, https://arxiv.org/pdf/1412.3555.pdf

[6] AndyLiaw, MatthewWiener, Classification and Regression by randomForest, ISSN1609-3631

[7] Yi Liu, Yuan F.Zheng, One-Against-AllMulti-Class SVM Classification Using Reliability Measures, 0-7803-9048-2/05/$20.0002005IEEE

[8] Yoon Kim, Convolutional Neural Networks for Sentence Classification, CoRR, abs/1408.5882

[9] Randomforest documentation in sklearn, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html?highlight=randomforest#sklearn.ensemble.RandomForestClassifier