Music Genre Classification By Lyrics Using Deep Neural Network

Maryam Barouti, Prince Osei Aboagye

CS 6956: Deep Learning for Natural Language Processing

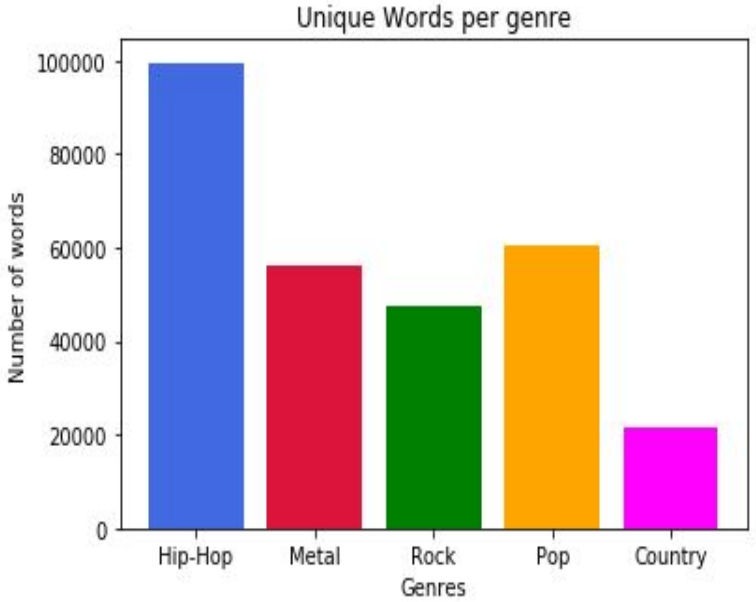
Introduction

Music genre classification based on lyrics alone is a very important and heavily researched task in Music Information Retrieval. The ever growing amount of online music databases and the ease with which people have access to music content calls for intelligent tools to help people browse and search these music databases and also categorize and organize songs they listen to. The question now is "what is it about a song that allows us to immediately determine it's genre?" The common approach would be to identify certain characteristics of the music such as rhythmic structure, harmonic content and instrumentation in other to categorize and organize songs based on the genre. However, this approach of genre classification comes naturally to individuals who have develop a musical ear. Being able to automatically classify and provide tags to the music present in a user's library, based on genre, without having to identify certain characteristics of the music would be beneficial for audio and video streaming services as Spotify, iTunes and Youtube. This project explores several machine Learning and deep learning algorithms to identify and classify the genre of a song given its lyrics.

Dataset and preprocessing

The dataset was from the MetroLyrics on Kaggle. There was more than 380,000 lyrics and there was 10 labels of music genre for the genres. We reduced the number of classes to 5 class consisting *Hip-Hop, Metal,Rock, Pop, and Country.* And for each of the genres we collected 8839 songs which length was more than 50 tokens. After extraction the dataset we then carried out a preprocessed the data. The data is read into python in the form of token. This is then followed by text normalization. After sentence tokenization various other techniques including cleaning text, case conversion, expanding contractions, correcting spellings,correcting repeating characters, removing stop words and other unnecessary terms,stemming, and lemmatization were performed in other to get the textual data

into a form that can be easily understood and interpreted for our classification task.



The chart below shows the unique words in each of the genres:

Figure 1: Unique Words per Genre

The maximum length of the lyrics we considered was 1000. Therefore if the lyrics was shorter than 1000, it was padded and if longer than 1000, we truncated it.

We used two types of embedding, one was the keras embedding which was trainable and the other one was GloVe which was not trainable.

Models

In this problem, we wanted to used the same data and compare which model will do better on this data. The models we used was *LSTM, GRU, HAN, CNN, Bi-LSTM, Bi-GRU.* And the embeddings we used were *Keras Embedding and GloVe;* the length of embedding we used was 100. Also we used SVM as our baseline.

For training the models, we used RMSprop classifier with decay. To avoid overfitting we used a drop layer of 0.5, and also the l1 regularizer of 0.001, also we used early stopping with the patience of 3.

Here are the architectures we used; since Bi-LSTM, Bi-GRU, LSTM and GRU are mostly the same we just report the architecture of one of them and the rest would be almost the same:

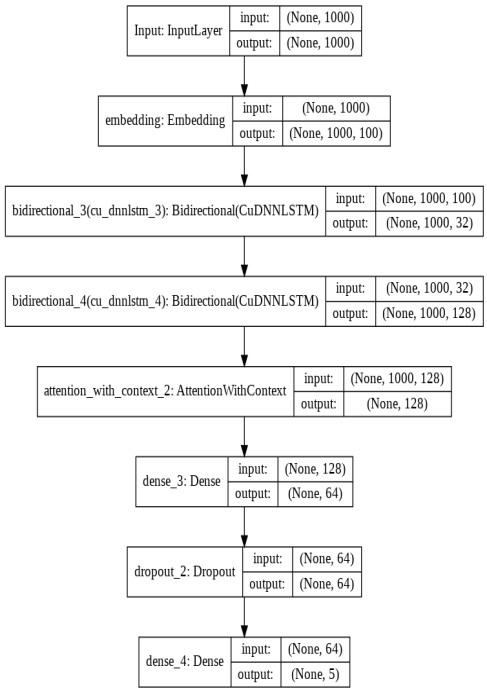


Figure 2: LSTM model

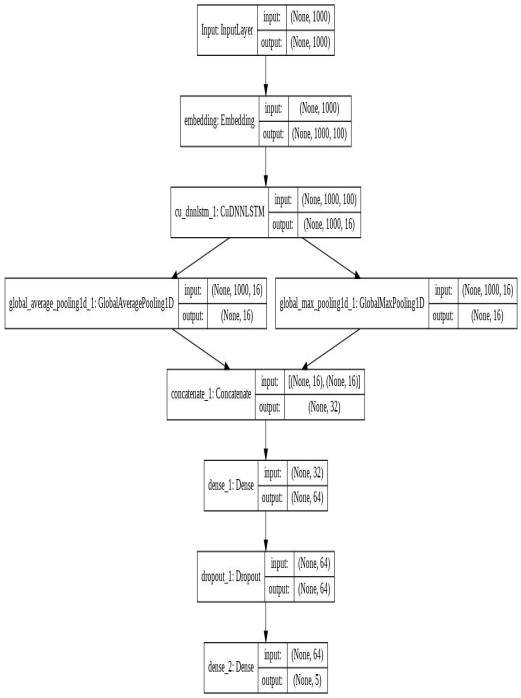


Figure 3: CNN model

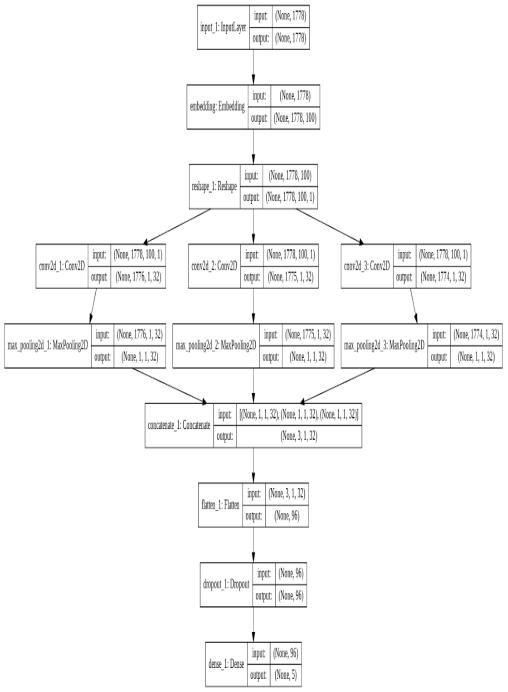


Figure 4: HAN model

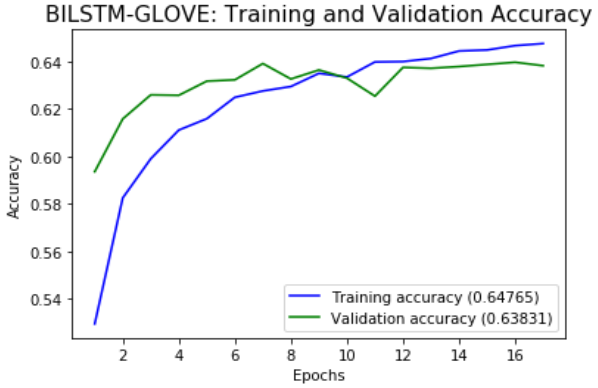
Results

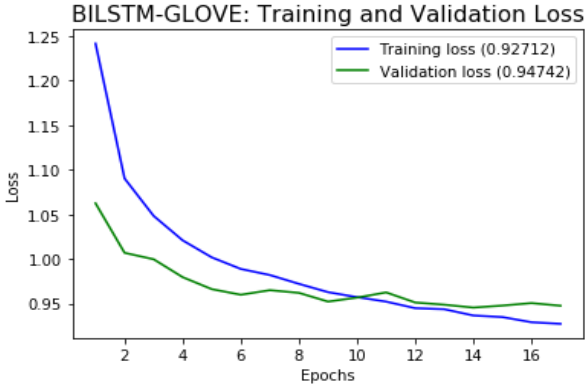
Here is a table showing all the accuracies we got for each of the models:

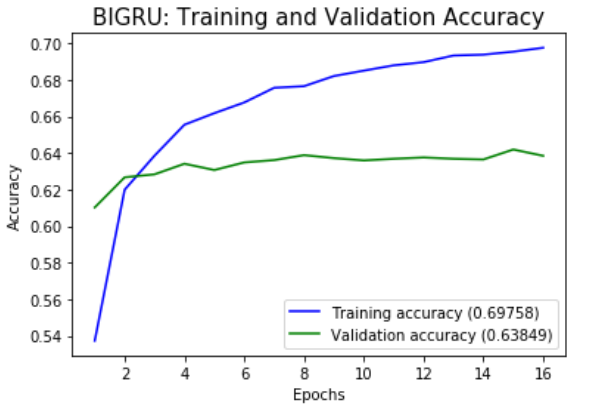
	LSTM	GRU	CNN	Bi-LSTM	Bi-GRU	HAN
Keras Embedding	63.33	62.64	**64.7**	63.67	**63.87**	63.16
GloVe	63.45	63.13	63.29	**64.85**	**64.16**	63.53

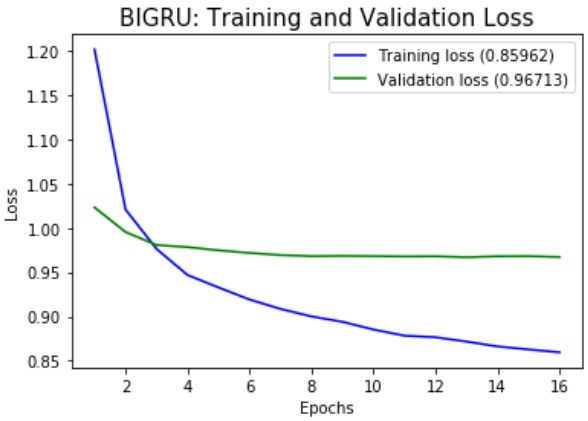
Table 1: Models accuracies for both embedding

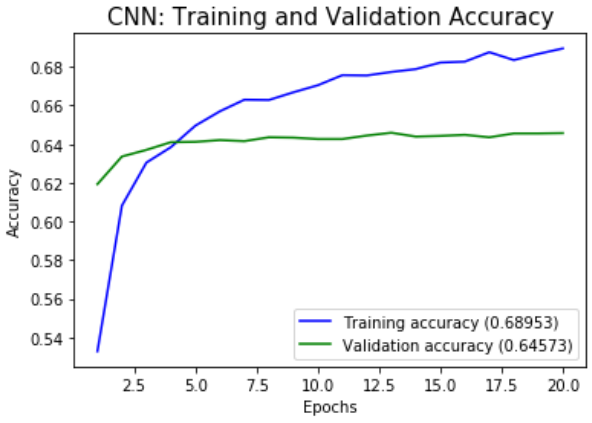
Here is the validation loss/training loss and validation accuracy/ training accuracy plots for our top four answers:

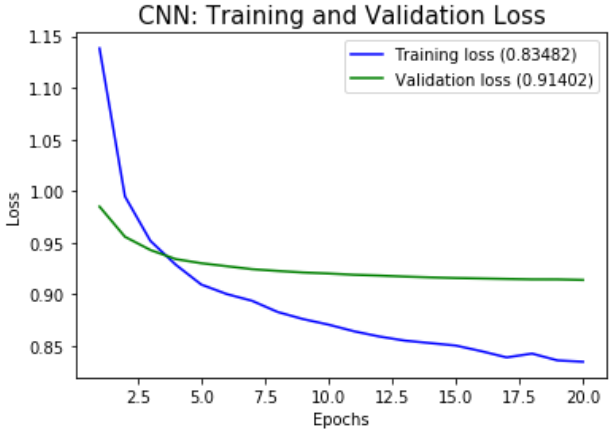


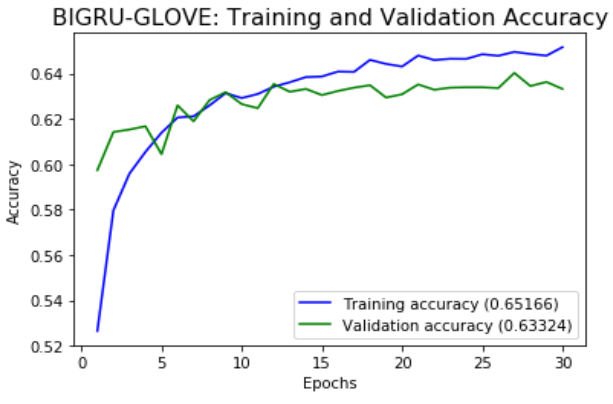


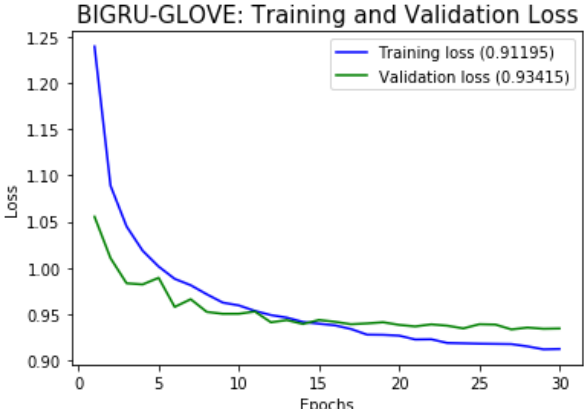












Since the task is multi-class labeling, accuracy doesn't tell too much about how good is the model. We also considered the precision, recall and F1-scores of the labels into account in purpose to not have a skewed model.

Here are the precision, recall and F1-scores for our top four models:

	Country	Hip-Hop	Metal	Pop	Rock	Micro avg
Precision	0.76	0.89	0.80	0.59	0.56	0.76
Recall	0.58	0.76	0.71	0.34	0.12	0.51
F1-score	0.66	0.82	0.75	0.43	0.2	0.61

Table 2: Precision/recall/F1-score for CNN with Keras Embedding

	Country	Hip-Hop	Metal	Pop	Rock	Micro avg
Precision	0.7	0.87	0.77	0.60	0.46	0.76
Recall	0.66	0.80	0.78	0.33	0.07	0.62
F1-score	0.68	0.83	0.78	0.43	0.12	0.66

Table 3: Precision/recall/F1-score for Bi-GRU with Keras Embedding

	Country	Hip-Hop	Metal	Pop	Rock	Micro avg
Precision	0.7	0.87	0.82	0.61	0.58	0.76
Recall	0.62	0.78	0.63	0.34	0.02	0.49
F1-score	0.66	0.82	0.75	0.44	0.05	0.6

Table 4: Precision/recall/F1-score for Bi-GRU with GloVe

	Country	Hip-Hop	Metal	Pop	Rock	Micro avg
Precision	0.72	0.86	0.82	0.63	0.59	0.76
Recall	0.61	0.78	0.71	0.37	0.01	0.5
F1-score	0.66	0.82	0.75	0.47	0.02	0.6

Table 5: Precision/ Recall/ F1-score for Bi-LSTM with GloVe

In terms of the F-Score, the best model was Bi-GRU with keras embedding with F1-score of 0.66, however the Bi-LSTM with GloVe embedding has the best accuracy of 64.85.

Future Work

One of the aspects that prevented us from improving our results was the fact that Google Colaboratory had limited RAM and Disk space, so we were forced to have an embedding of length 100 also we just considered the first 1000 words of the sentences into account. So definitely if we had more time we would consider other types of pretrained embeddings and tried to solve the problem with those embeddings and see what happens. Also, we could implement our own embedding and try to build our model on top of those word embeddings.

Conclusion

We have learnt lots from this project, we applied all the deep neural networks we have discussed in the class in this project and have done a hands on project related to all types of neural networks that can be applied to texts and some of them were mentioned in the class and some of them were ideas we got from homeworks or in presentations. We also learnt how to deal with pre-trained word embeddings or how to use keras embedding and how it exactly works. Also,tuning hyperparameters could be rough in some cases, especially when the models are prone to overfitting and we solved this problem by using other methods of preventing overfitting and by keeping the track of loss and accuracies of Validation and training set.