©2020
Ishank Sharma
ALL RIGHTS RESERVED

HIT SONG CLASSIFICATION WITH AUDIO DESCRIPTORS AND LYRICS

BY

ISHANK SHARMA

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Dr. Maria Striki

And approved by-

_____

_____

New Brunswick , New Jersey

October, 2020

ABSTRACT OF THE THESIS

Hit Song Classification using Audio and Lyrics

by ISHANK SHARMA

Thesis Director:
Dr. Maria Striki

Hit Song Science aims to predict a songs popularity based on song structure and external features. To help provide an efficient and accurate tool for Annual Top-100 Billboard Song Classification, we apply fine-tuned BERT transformer and a joint learning approach. We also explore different audio descriptors and lyrics based embedding for modeling hit song classification task on an innovative western song dataset created by us. We address class imbalance and data scarcity issues associated with traditional datasets by employing shared layer architecture and penalizing loss. We highlight a comparison study of three distinctly designed neural network architectures. All models yield high overall accuracy with relatively low training cost.

Acknowledgements

I would like to thank my supervisor Dr. Maria Striki at Rutgers Unniversity, New Brunswick for her guidance and support in building this project together. Furthermore I would like to thank Dr. Zoran Gajic, my Graduate Programme Director for all his help and for always being available throughout the project process.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Hit Song Science (HSS) aims to infer whether a song's commercial success full scale release of the song. It makes the assumption that audio and lyrical features define the underlying structure of any song and can be used to predict its chart topping behavior. Hit Song Science is a widely researched topic in Music Retrieval as it can be used to identify key musical elements pivotal for success and predict commercial popularity of the song [**4**]. Needless to say this method is particularly interesting for songwriters and composers, as they can use these techniques to allocate their time, energy and resources for targeted song promotion and production appealing to a broad audience. Therefore, a natural next step in HSS would be utilizing the identified crucial features for the automatic generation of musical pieces. For hit song prediction task people have been relying on musical features extracted solely from the audio or an external ecosystem, for instance incorporating social media or market data [**5**]. Natural language processing and Deep Learning[**6**] has been one of the most influential tools in last decade to extract, analyze, and classify essential audio and lyrical features in a song.

In this work, I will explore accurate and efficient modeling of hit song prediction task from three aspects- using audio features representing underlying low level features of a song, lyrical features, a high level representation of song and a combined technique for prediction using both low and high level features. For the task, I have also created a modern innovative database of Top-100 annual billboard songs from 1960-2019. This database has audio features extracted from 30 second Spotify preview with corresponding lyrics and is openly available for research community.

Previous work in HSS has primarily focused on using large commercial datasets or traditional music datasets such as instance Million Song Dataset or Lakh MIDI dataset[**7**]. There a few

problems associated with modeling using these datasets as they have only weekly Top-100 chart songs from 1922-2011 and with missing or incomplete lyrics or audio feature meta data. Moreover, other datasets- KKBOX[**8**], Earwormery [**9**] are commercially available and not available for open public use. Therefore, we created a new dataset, first of its kind, with annual Top-100 Billboard songs from 1960-2019 taken from Wikipedia. Each song has associated rhythmic and chord features extracted with Essentia framework from a 30 second audio preview using Spotify API. Its corresponding lyrics were downloaded using Genius API. In all, there are 6938 songs with hit and non-hit label category. We will be building our deep learning classification model and monitor its performance base on this developed dataset.

According to our study, currently on this kind of dataset; none of the previous work have utilized joint learning of audio features and lyrics for hit song prediction task. Researchers have either used deep learning methods with low level audio features such rhythm, spectral information, chords [**10**] [**11**] [**12**] or lyrics alone[**13**]. Some studies have combined both and low level audio features for joint learning. Middlebrook and Sheikh [**14**] used Spotify API computed high level features such as danceability, mood, energy, mode etc on a large dataset showing promising classification results. Zangerle et al. combined joint learning high and low level audio features. Following a similar approach we will be exploring three methods- (i) audio only, (ii) lyrics only, and (iii) combined audio and lyrics joint learning for hit song prediction task.

CHAPTER 2

Methods

In this section we will emphasize on illustrating our choices methods and give a detailed explanation on the general methods that are utilised in in later sections. We will also elaborate on our choice of model and model-specific components such as activation functions and convex optimizers.

2.1 Artificial Neural Networks

“Artifical neural networks” (ANN) or “Neural networks” (NN) is a class of pattern recognition systems inspired by the design of human brain. ANNs have been widely successful in modeling classification and regression networks. Due to its popularity in wide array of machine learning tasks, a separate field called "Deep Learning" emerged to study stacked neural networks with many layers [**15**]. Many deep learning architectures have proven to outperform humans in several structured learning tasks. Such network architectures are referred as Deep Neural Networks or DNNs. A neural network consists of connected or neurons. A node is a computational representation of biological neurons and connections in the human brain. Each node is defined by an activation function and a collection of such nodes is called a hidden layer. In DNN models, there are several such neuron layers stacked together. In general, DNNs can be trained with unsupervised and supervised learning techniques. In supervised learning, we use labeled data to train the DNNs and learn the weights that minimize the loss function for classification or regression task. Refer Figure 2.1 for an illustration of a Deep Neural Network.

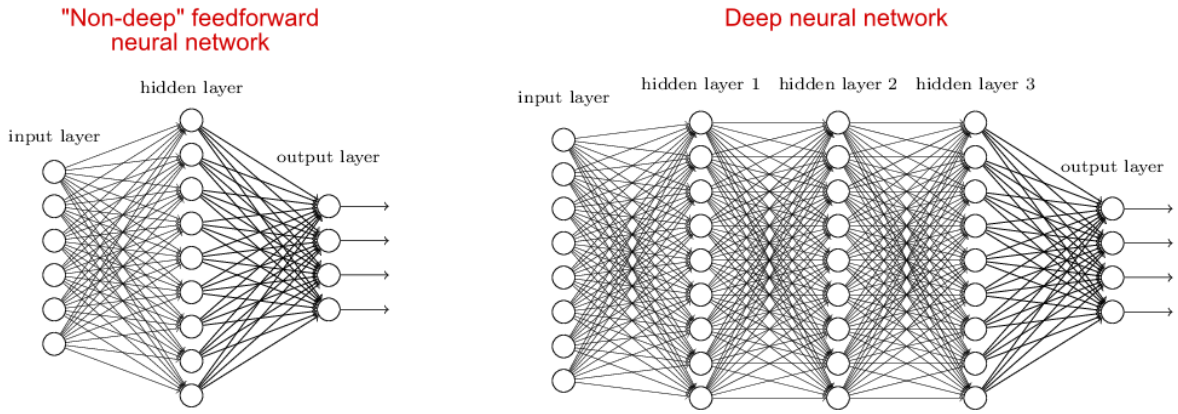

FIGURE 2.1: Illustration of a Deep Neural Network. Adapted from [**1**]

Backpropogation algorithm or Backprop efficiently computes the gradient of the weight w.r.t to the underlying loss objective of the problem. Although, there are many variants of Backprop, for our task we selected 'Adam' optimizer based on our experiments.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

FIGURE 2.2: Overview of Adam algorithm. Adapted from [**2**]

The Adam optimization algorithm [**2**] is an extension of the general gradient descent algorithm. It is a combination of the two backprop methods- "RMSProp" and "Adagrad" and is derived from the concept of Adaptive moment estimation, hence it is also known as Adam. It uses 1st and 2nd moment approximation as moving averages of the gradient and of the squared gradient exponentially decaying w.r.t. the iteration using decay rates $\beta 1$ and $\beta 2$ as hyper-parameters. Refer figure 2.2 for overview of algorithm.

2.2 Batch Normalization

Batch Normalization or BatchNorm is a widely adopted technique that enables efficient and faster training of deep neural networks[**3**]. It has not only sped up modern deep neural architecture, but increased the baseline accuracy and regularization capability in a network. Batch norm works by eliminating the internal covariance shift of layer inputs. It basically standardises the each layer input before propagating the input forward. In our model the input and its subsequent feedforward layer is followed by BathcNorm and Dropout layer. We use the BatchNorm before Dropout since it has been proven to be effective in many regression and classification network studies[**16**]. Refer figure 2.3 for a summary of BatchNorm algorithm.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

FIGURE 2.3: Batch Normalization transform algorithm. Adapted from [**3**]

2.3 Contextual Embeddings

Traditional word level embeddings like Word2vec, Glove, FastText [**17**] incorporates possible meanings of a words derived from surrounding sentence structure. However, paragraph texts consists of multiple sentences which are coherent units of natural languages that convey information at a pragmatic or discourse level. Word embeddings give unsatisfactory results in a scenario where the global context is differs from the local sentence structure. For this

study we employ BERT embeddings to get underlying sentence vector of lyrics. Bidirectional Encoder Representations from Transformer (BERT) is a masked language model which encoder decoder network of transformers for bidirectional learning[**18**]. Additionally, it can be easily fine-tuned for any downstream task with few additional layers. BERT ans its variants have outperformed traditional language models is variety of tasks such as question answering and language inference[**19, 20**]. It can accurately learn the distribution of out of vocabulary words without substantial modifications [**21**]. BERT encodes the sentence structure with three types of embeddings- (i) Positional (ii) Token (iii) Segment, to induce temporal and contextual information. Unlike traditional natural language models BERT can be easily parallelised and trained with low computational overhead making it highly scalable and efficient for downstream performance. For our task we will generate BERT based lyrics embedding and utilise it for hit-song classification task. We prove that lyrics embedding can be used to successfully disambiguate hit or non-hit songs as well as it offers insight on high level abstraction of songs. Figure 2.4 shows overview of BERT architecture. We give detail of our implementation design choices in section 3.4

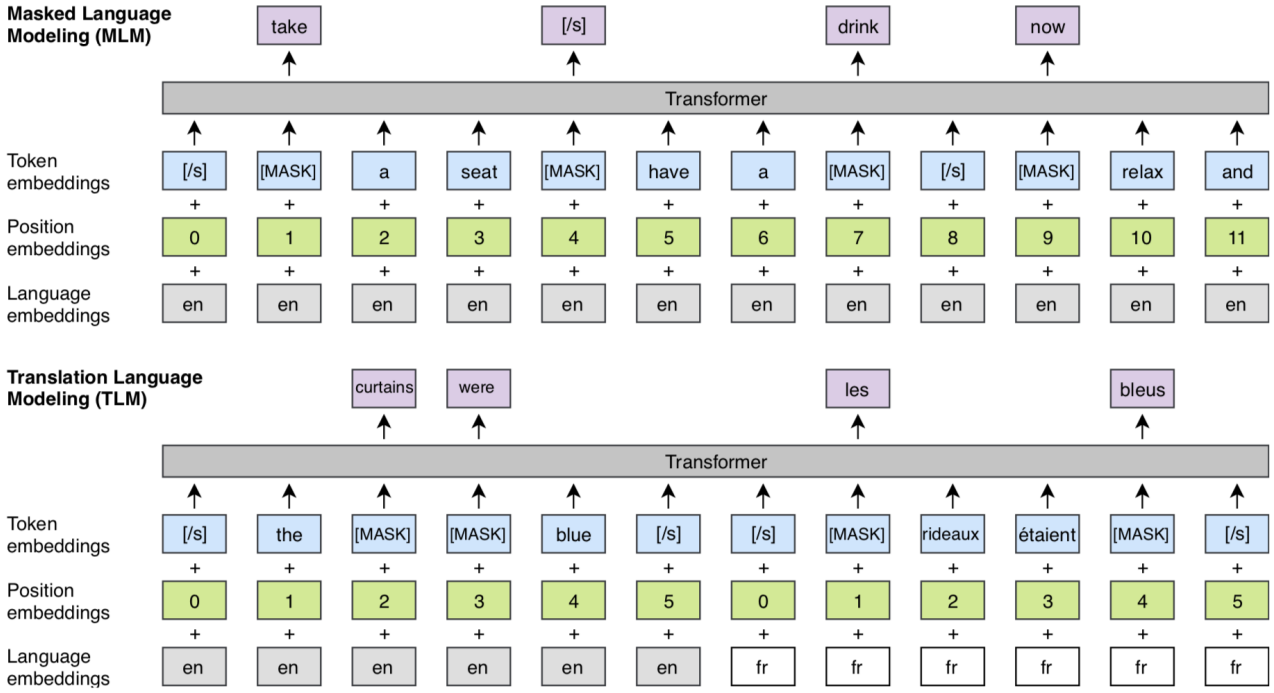

FIGURE 2.4: Representation of BERT architecture

2.4 Joint Objective Learning

Joint objective learning is the concept of having a system learn tasks simultaneously with different representations. The notion behind the idea is, that many tasks require a similar basic understanding of data at a lower level, for example lower layers of two different networks can have the same distribution of weights, hence the lower part might be improved by sharing hidden layer weights to learn a shared representation of multiple tasks. This approach can be further extended to multi-modal learning where in two separate networks can have different objective but can a shared layer for joint knowledge representation. In our model, we experiment with three different model architectures and a joint approach of learning shared representation of a song with lyrics and audio descriptors. Figure 2.5 shows our multitask learning approach for hit song classification

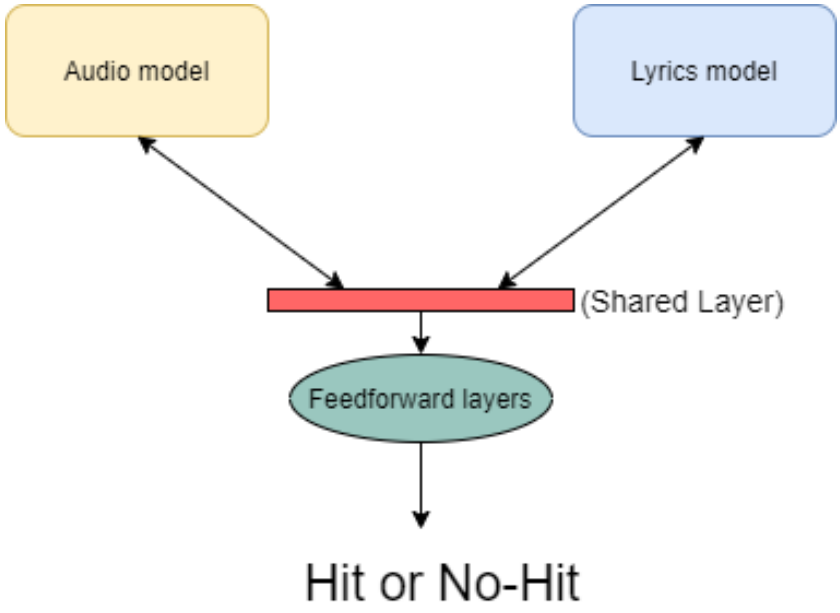

FIGURE 2.5: Joint learning of Lyrics embedding and Audio descriptor features

2.5 Precision and Recall

In addition, to calculating classification score for our models, we will compute Precision and Recall scores which gives a more cross sectional view of model performances. Precision, is

defined as $\frac{TP}{TP+FP}$ wherein TP stands for true positive rate, and FP is false positive rate. This score calculates the ratio of correct classification from all classifications. Whereas, Recall is calculated as $\frac{TP}{TP+FN}$, where FN stands for the false negative rate. This measure is also known as "Sensitivity" and shows the classifier's efficiency to classify true positive results.

2.6 Learning rate decay

Modern neural networks wide, deep and nonconvex. They are powerful approaches for representation learning and serve as core components of artificial intelligence systems. Due to non-convexity and high parametric cost, we utilise regularization techniques to train these systems effectively and efficiently. Learning rate decay has been a de facto technique for training such neural networks[**22**]. We begin with large learning rate and reduce it over multiple times. For our use two types of decay methods-

- Step Decay: Reduce learning rate by a factor on every *nth* step. We define it by following equation-

$$lr = lr0 * drop^{floor(epoch/epochs_drop)} \tag{2.1}$$

- Exponential Decay: Reduce learning rate exponentially if specified performance metric doesn't improve. It is defined as follows-

$$lr = lr0 * e^{(kt)} \tag{2.2}$$

CHAPTER 3

Data Treatment

We base our study on a new created dataset for Top 100 annual Billboard charts (1960-2019). In this section we will summarize the structure of our data, issues and text processing. Finally, we build BERT embedding vectors for our lyrics based classification model. We hypothesize that these newly created features are salient in liking/disliking of song. We further choose only that categories that are proven to be effective for hit song identification in previous works.[**23**]

3.1 Data Structure

Our dataset consists of 6938 commercial English audio songs from 1960-2019 in Top-100 annual billboard charts. For all song we gathered lyrics using Genius API. To get audio, we extracted 30 second song preview for each song with Spotify Developer API. Finally, low level audio descriptors are extracted using Essentia audio processing framework[**23**]. The dataset and code is freely available in this repository. In total, there are 23 features-numerical, categorical and string based and one label category. Figure 3.1 shows some examples from our dataset. To our knowledge, there is no other open dataset which has audio descriptor features, song info, lyrics as well as 30 second song clip. Although Million Song Dataset (MSD), a widely popular dataset, contains song info and audio descriptors of one million songs that are representative for western commercial music released between 1922 and 2011 [**24**]. It is not suitable for our task since it contains old songs and incomplete data information.

zerocrossingrate_stdev	danceability	...	bpm	chords_key	chords_scale	beats_count	name	year	ranking	artist	lyrics	label
-1.084370	-0.397708	...	-1.219666	9	0	13	theme from a summer place	1965	1	percy faith	summer place \n rain storm \n safe warm \n sum...	1
-1.463847	-0.582061	...	-1.285930	2	0	12	tonight's the night (gonna be alright)	1982	1	rod stewart	Verse 1 \n stay away window \n stay away door ...	1
-0.924485	-1.020842	...	1.119001	3	0	42	stranger on the shore	1967	1	acker bilk	stand watch tide blue dream dream watch ship s...	1
-0.844275	-0.759281	...	0.868882	6	0	37	the way we were	1979	1	barbra streisand	Intro \n hmmmmm hmmmmm \n Hmm hmm hmmmmmm hmmm...	1
-0.530639	0.674443	...	-1.352548	7	0	11	tie a yellow ribbon round the ole oak tree	1978	1	tony orlando and dawn	verse 1 \n come home \n time \n get know \n re...	1

FIGURE 3.1: Samples in our Dataset.

3.2 Issues

One of the issues with our dataset is a slight class imbalance. There are in total, 3646 negative samples(non-hit) and 3292 positive (hit) samples. As explained by Ying Liu et al. [**25**] sometimes, in a situation, the high cost of human efforts required in labeling can also cause this imbalance. In our approach we can also pass data enough times (epochs) through neural network for efficient learning. Moreover, we can also use techniques such as under sampling and feature engineering to alter sample size [**26**] Although there are no missing values in our dataset. For some songs we were unable to get lyrics accurately. Thus, we impute such lyrics with Artist, Song and Release Year. We further encode label categories and standardize numerical categories.

3.3 Data Preprocessing

We identified 23 features in total features in total. Refer Table 3.1 for list of all selected features. We remove any missing data and only samples with full data are kept. All numerical features are standardized and categorical features are encoded with category label.

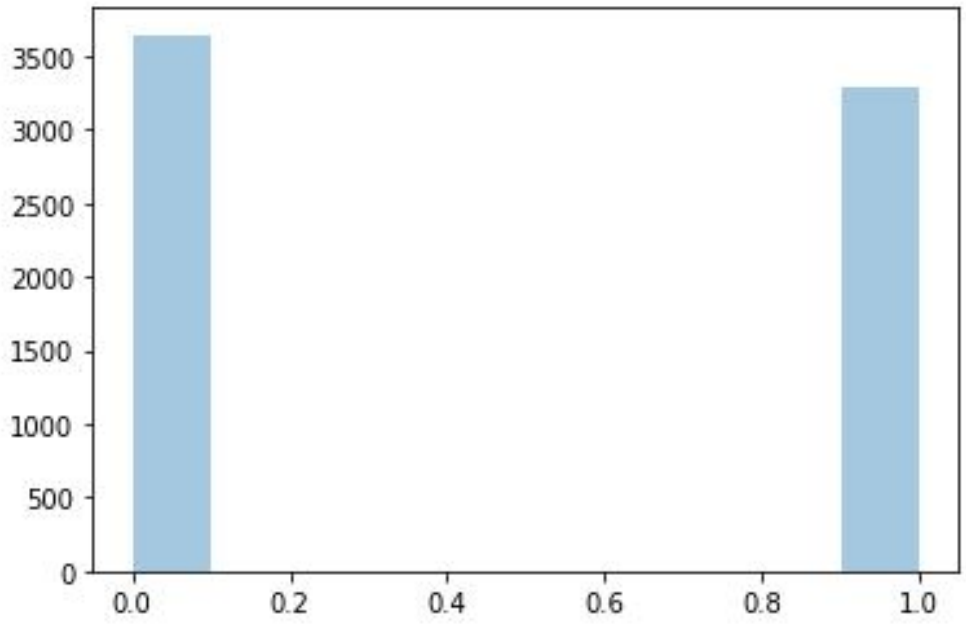

FIGURE 3.2: Total label samples, where 1(Hit songs) and Label 0 (non-hit songs).

We define two types of features- Low-level and High-level. Low-level features corresponds to all audio features which define underlying structure of the song, whereas we use lyrics of the song as a high-level feature representing context of the audio. We experimented with min-max and standard scaling for numerical variables. We chose standard scaling based on our experiments, as it gives better model learning results. 'Year' is also concatenated as a low-level feature for our model input. Furthermore, lyrics is a text based feature and we apply text normalisation, stop word removal and regex cleaning [**27**]. This is helpful in getting accurate text embedding in next stage of our process. For each song we define a label- 1 (hit-song) or 0 (non-hit). If a song ranking falls within Top-100 Billboard charts it is categorized as hit-song otherwise a non-hit. Additionally all missing lyrics are imputed with the phrase- '<Title> released by <Artist> in <Year>' For more details on preprocessing refer our code.

Numerical	zerocrossingrate_mean, tuning_diatonic_strength, spectral_kurtosis_mean, spectral_kurtosis_stdev, onset_rate, tuning_frequency, chords_strength_mean, chords_strength_stdev, zerocrossingrate_stdev, danceability, beats_loudness_mean, beats_loudness_stdev, chords_number_rate, chords_changes_rate, bpm
Categorical	chords_key, chords_scale, beats_count
Meta	name, year, 'ranking, artist, lyrics

TABLE 3.1: DataTypes of selected features in our dataset

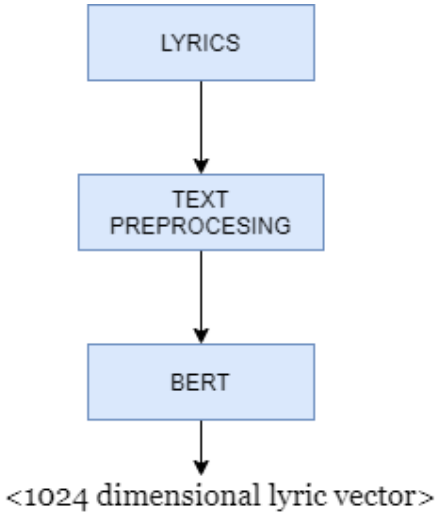

FIGURE 3.3: Process flow to get vector embeddings from lyrics

3.4 BERT based Lyric Embedding

Distributional word and sentence representations obtained in unsupervised manner on large-scale corpora are widely used in modern natural language processing systems. [**28**]. Researchers have shown that contextual embeddings pretrained on large-scale unlabelled corpora achieve state-of-the-art performance on a wide range of natural language processing tasks, such as text classification, text summarization and question answering [**29**]

We further convert lyrics to a feature vector with BERT document embeddings [**18**]. BERT is a multi-layer bidirectional Transformer encoder released by Google. It generates state of the art contextual embeddings and the model can be easily trained for any downstream natural language task [**30**]. Furthermore, BERT accurately to produces embedding vector for out-of-vocabulary words and variable length input, which is suitable for our method. For our task we will use 'bert-large-uncased' model which has 24-layer, 1024-hidden, 16-heads (340M parameters) and was trained on lower-cased English text. Additionally, we do not train the network further and use the pretrained network to generate 1024 dimensional output embedding for each lyric our our song. We develop a software built in Python programming language using 'bert-as-service'[**31**] library for this task. The process flow is represented in Figure 3.3 Our innovative approach of using contextual lyrics embedding for hit song classification task as not been used in any previous studies. Furthermore, we analyse and show the performance of this approach in comparison to using audio descriptors.

We also deduce that lyrics embedding can be used in joint training approach and successfully enhances model performance. For our task we are using lyrics from multiple genres- Rock, Pop, Hip-Hop, RnB, Folk which provides more generalisation ability to our modeling approach and classification results.

CHAPTER 4

Models

We design our experiments based on the three separate neural network models for hit song classification task. We further compare each model on same training and test data split.

4.1 Audio Model

Audio model or low level classification is a feed forward network we train using the standardised audio descriptors and year from our dataset. Audio descriptors represent low level information about the audio song and its key components. We can extract insights about the underlying structure of the song from time signature changes, MFCC, beats, chord duration. etc, which are salient in predicting popularity of it. Refer figure 4.2 for our audio model design. We use Adam optimizer with initial learning rate$= \ 0.001$. All first four layers have RELU activation function, whereas the last two dense layers have Sigmoid activations. We train our model with 'Binary Crossentropy Loss' or 'Log Loss' for 700 epochs and a bacth size $= \ 1024$. Figure 4.1 defines are logloss for two class labels.

4.2 Lyrics model

Our Lyrics model or high level classification model fine tunes a feedforward network with pretrained BERT embeddings as dense layer input. Lyrics represent a high level contextual

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

FIGURE 4.1: Log loss for classification task, N = 2 (two classes)

structure of song[**13**]. We all have experienced humming our favorite catchy lyrics, therefore lyrics contribute much to likeability of song [**32**] We use Adam optimizer with learning rate= 0.001 and employ learning rate step decay of 0.5 after every 5 epochs. Furthermore we apply learning rate if validation accuracy plateaus for 10 epochs. These techniques are effective regularization methods to prevent over fitting and control loss gradient momentum [**33**]. Refer Figure 4.3 for representation of our lyrics based model.

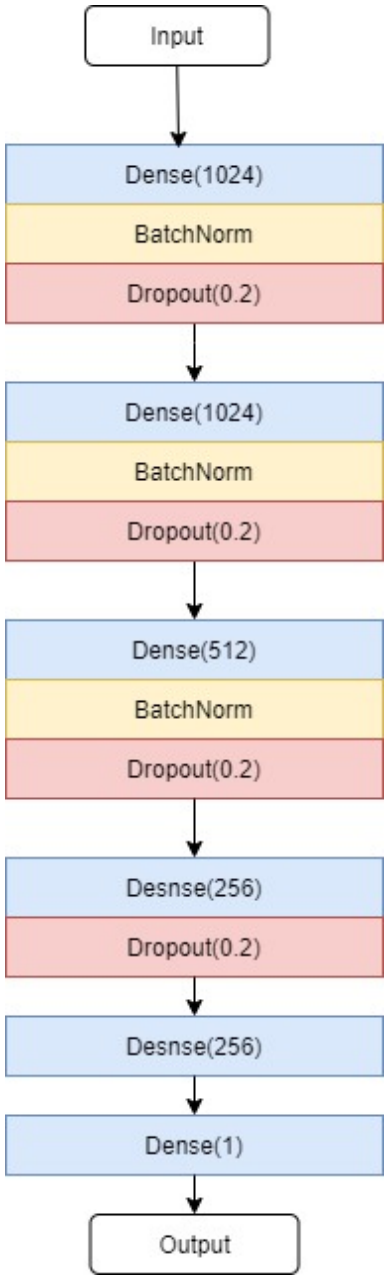

FIGURE 4.2: Hit Song classification model with audio descriptors input

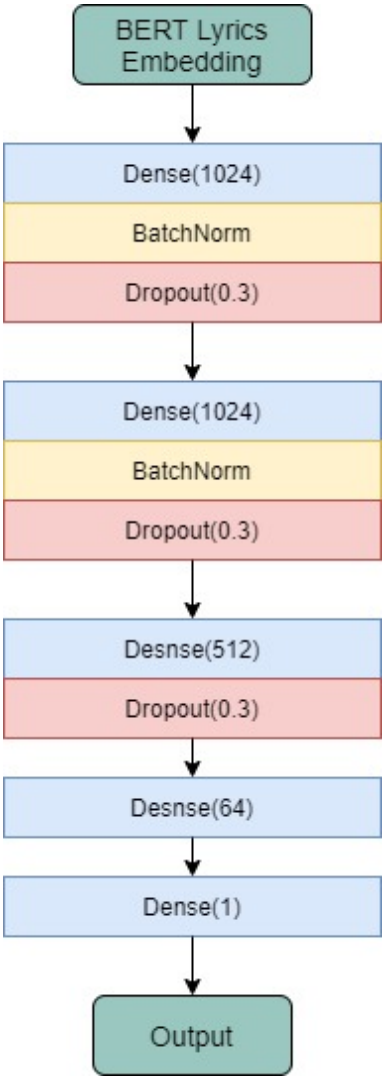

FIGURE 4.3: Hit Song classification model with BERT based lyrics embedding input

4.3 Audio+Lyrics Model

For our shared layer model we modify our existing audio descriptors and lyrics based model. We keep the same objective as our previous models however, we add a shared hidden layer for joint learning of audio descriptors and lyrical features from embeddings. We hypothesize that both audio and lyrics have insightful feature information which can be enhanced with joint representation learning. All layers except the last layer have RELU activation and the final dense layer weight output is followed by Sigmoid activation. We employ both step decay and learning rate decay. Additionally, we initialize weight distributions to be random normal. Figure 4.4

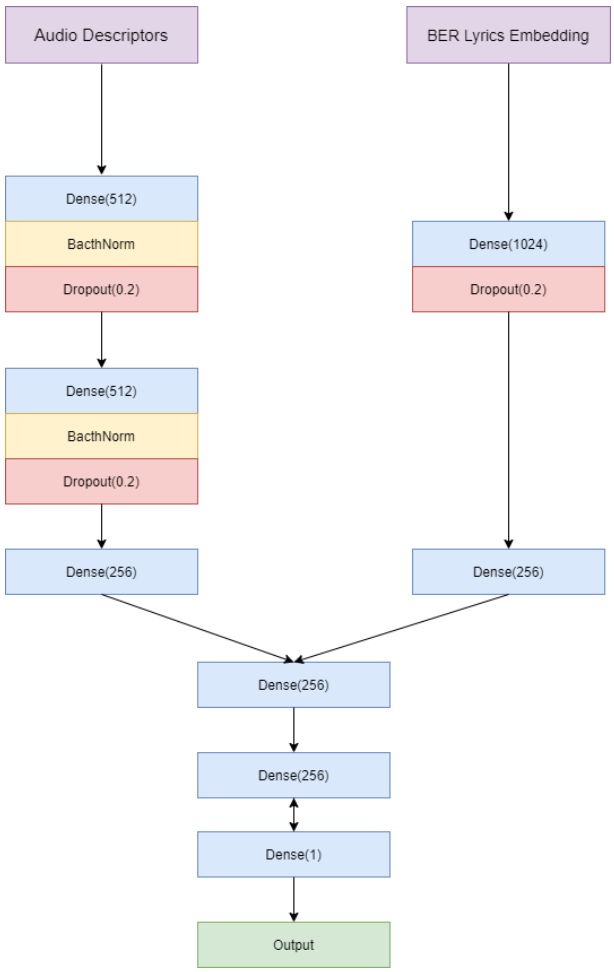

FIGURE 4.4: Hit Song classification with joint learning of lyrics and audio descriptors

CHAPTER 5

Results and Discussion

This section will illustrate the training and test cost, performance of previously designed models namely Audio model with input features as selected low level audio descriptors, Lyrics model with lyrics embeddings generated from BERT transfomer and a combination of these to models with a feature aggregation layer or shared layer for joint learning. Our analysis will focus on a comparison study between classification scores of the three defined models.

5.1 Analysis of Audio Model

Our neural network model for hit song classification using audio descriptors took 700 epochs with a batch size of 1024 samples to complete training. Refer figure 5.1 for loss profile. From training profile, shown in figure 5.2, it can be observed that the model stops learning after 300 epochs.

From training profile, we can say that our developed classification model based on audio descriptors stops learning after 330 epochs. Refer Table 5.1 for Precision-Recall score analysis of audio based model.

5.2 Analysis of Lyrics Model

For classification based on high level feature of a song we designed a BERT based model and train it for 700 epochs with batch size of 128. We fine-tuned BERT generated embeddings with fully connected layers and Bacthnorm. Refer figure 5.3 for loss profile. From training profile, shown in figure 5.4, it can be observed that the model stops learning after 270 epochs.

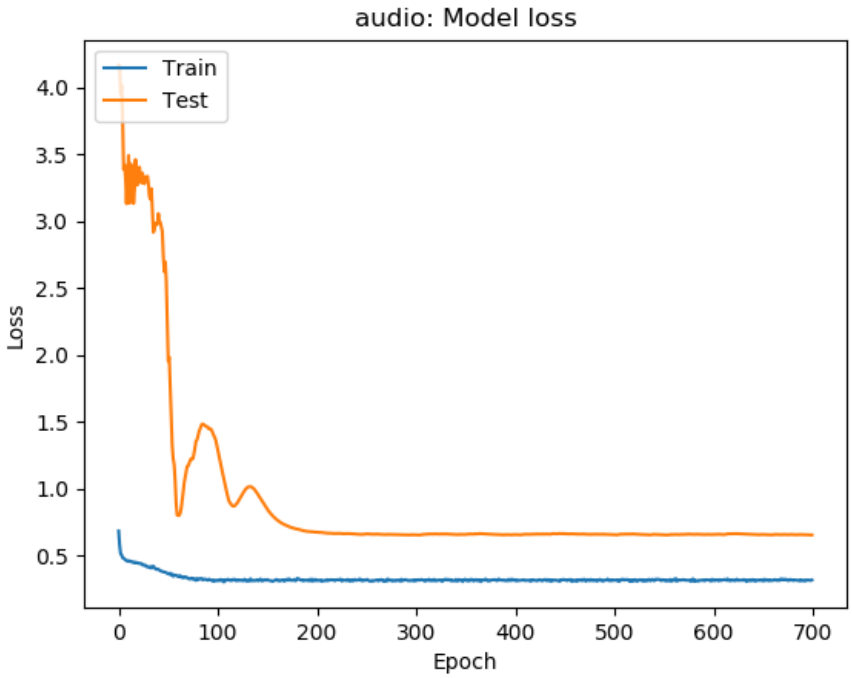

FIGURE 5.1: Loss profile of Audio model.

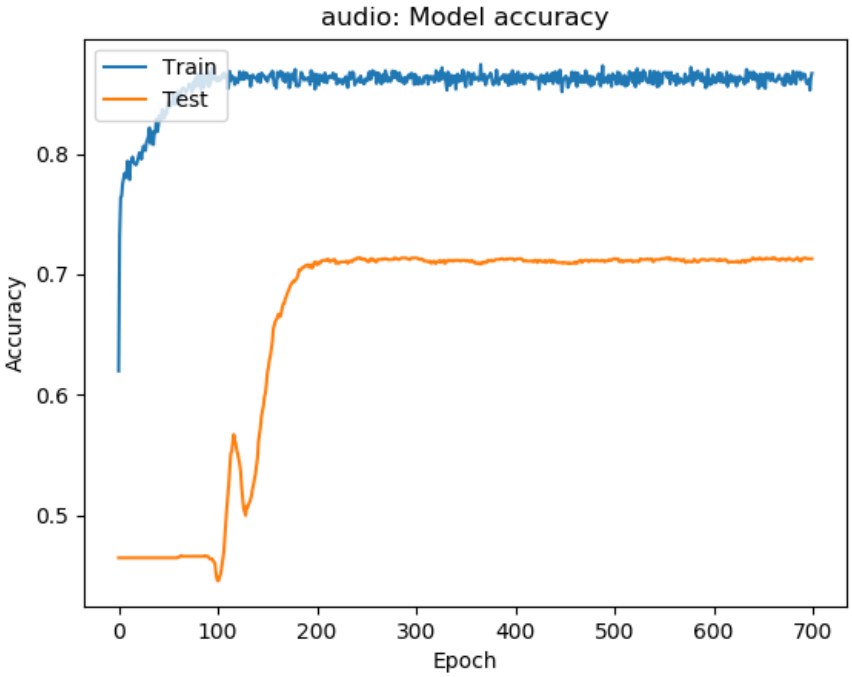

FIGURE 5.2: Accuracy profile of Audio model.

This model performs moderately better than our low level feature based classification network. Refer Table 5.1 for Precision-Recall score analysis of lyrics based model.

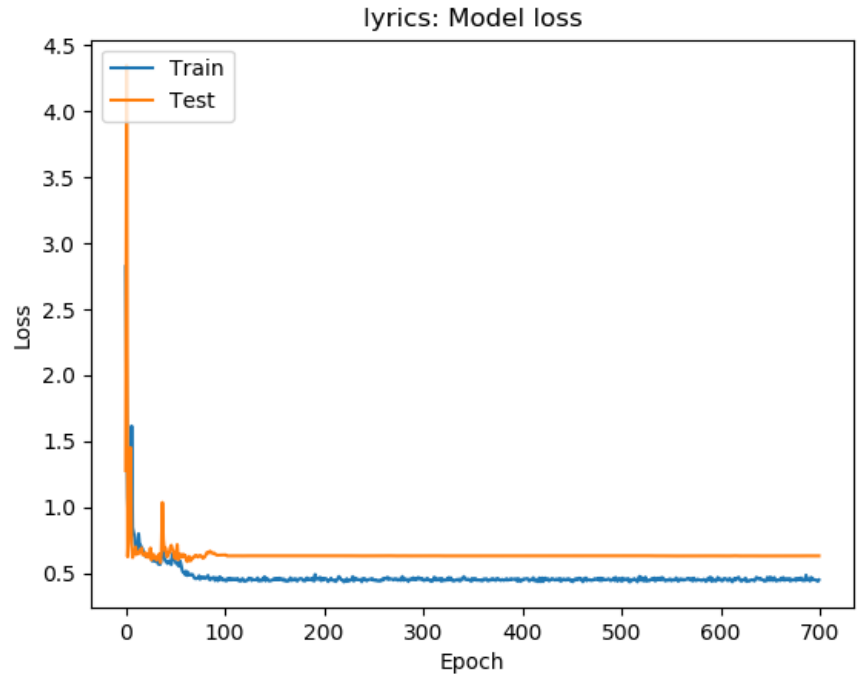

FIGURE 5.3: Loss profile of Lyrics model.

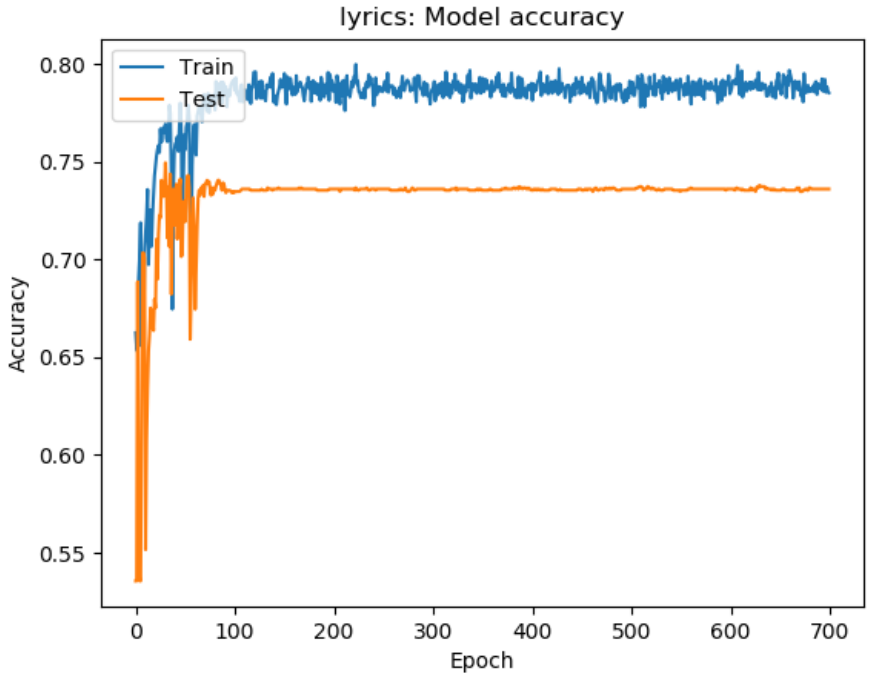

FIGURE 5.4: Accuracy profile of Lyrics model.

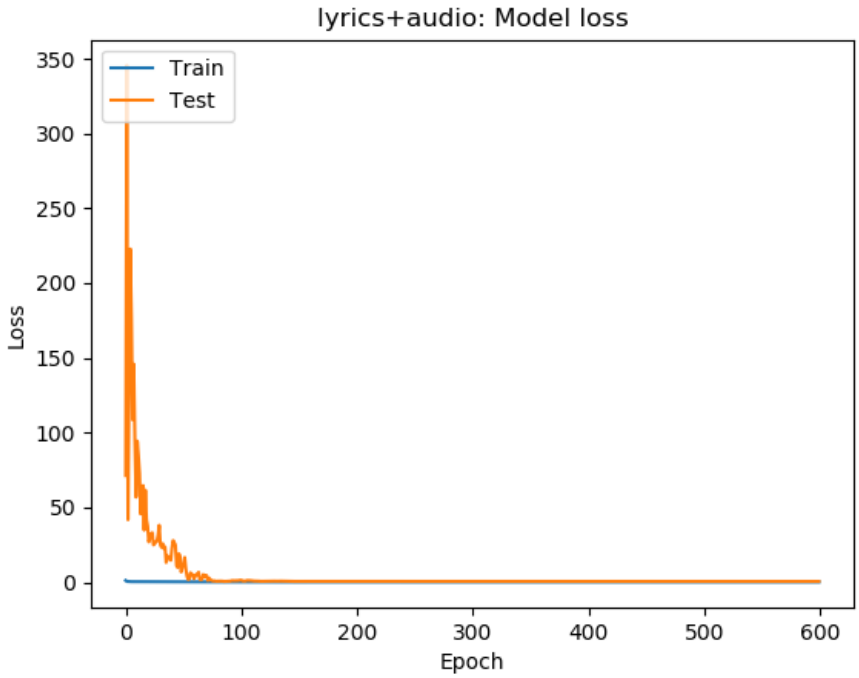

FIGURE 5.5: Loss profile of Lyrics+Audio model.

	Accuracy	F1	Precision	Recall
audio	0.7043	0.6454	0.7318	0.5943
lyrics	0.7360	0.7176	0.7070	0.7375
audio+lyrics	**0.7689**	**0.7490**	**0.7629**	**0.7530**

TABLE 5.1: Models accuracy metrics on Test set

5.3 Analysis of Audio+Lyrics model

In third comparison, we conduct analysis of audio descriptors and lyrics based classification model. We preform joint training of high level (lyrics) and low level (audio descriptors) to predict whether a song will be in Top-100 annual billboards or not. We train the model for 700 epochs with batch size of 512. From accuracy profile of the model (Figure 5.6 it is clear that with joint training and without any considerable changes we can improve the classification score for Top-100 Billboard hit song prediction. The result support our hypothesis that both low level and high level features contribute to structural information about popularity of a song. From figure 5.6 it can be seen that learning saturates after 300 epochs for this model. refer Table 5.1 for Precision-Recall score comparison of all models.

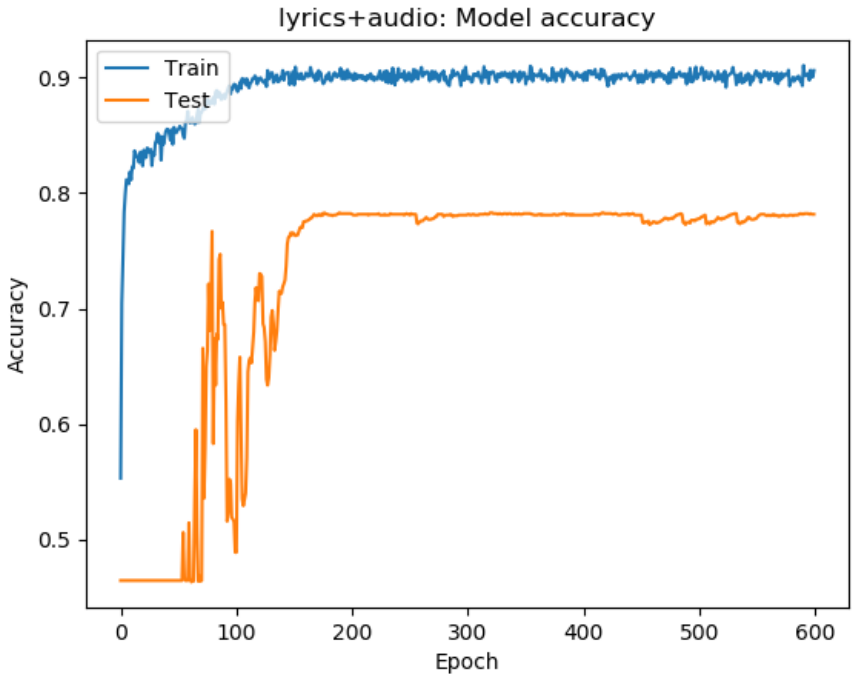

FIGURE 5.6: Accuracy profile of Lyrics+Audio model.

Finally, we test all three models on our test set consisting of 1735 samples. It can be seen from Table 5.1 that sensitivity and accuracy is highest for audio+lyrics among all three models. Although audio model successfully disambiguate hit or non-hit songs , our lyrics based sentence embedding network outperform audio descriptor model with better sensitivity. Moreover, we can say that our finetuned lyrics embedding generated from BERT is a better at depicting a song likability in comparison to our selected audio descriptors. This can be improved by leveraging better features selection and audio data preprocessing [**10, 34**]. The joint learning approach with a shared layer and common objective for song classification, proves to be effective technique[**35, 36**] in achieving high accuracy with moderate training cost and modifications to existing architectures.

CHAPTER 6

Conclusion

We successfully employed three neural network models for our Top-100 Billboard Song classification task. It can be concluded from our results that both low level features such as audio descriptors and high level features like song lyrics are essential structures for identifying song popularity. We achieved higher accuracy than our two independent models namely lyrics model and audio model, with relatively low training cost. Comparison studies , show that joint learning based on aggregated features or shared layers improves model performance as well as generalisation capability. Moreover, a BERT can be further fine-tuned with a task specific network to give satisfactory results for Lyrics based hit song classification. We can further improve this performance by training BERT on targeted corpora. Penalizing loss function and applying BatchNorm significantly improved the training procedure and are essential regularization techniques for this classification task. As mentioned in [**37**], one of the major challenges for machine learning researchers is unavailability of "high quality data" . If we can have a significantly larger training sample size, with more audio and lyrical features we will achieve sound improvements to our models without heavily relying on network tuning and class imbalance issues. These observations made also suggest that additional hyper parameter tuning can enhance performance of all three models. Additionally using more audio descriptors from Essentia framework can add to model generalisation capabilities. Nevertheless, the our designed model proves efficient in identifying Billboard Hit songs from a large sample set.

CHAPTER 7

Future Work

On a closing note, our intention and effort to build an open source dataset for modern Western commercial music was successfully concluded. To encourage research and discussion on this task we will make our dataset publicly available for use in innovative and creative manners. We hope that with this collaborative learning effort can improve existing techniques as well as promote research in the field of Hit Song Science. Furthermore, we successfully prove joint learning method with lyrics and audio descriptors is efficient technique for hit song classification task. This approach can be enhance by using better lyrics based embedding or inducing for information in network learning process through social media tags, news, sales trends etc. We aim to further analyse our lyrics based approach to gain insights on musical trends and associated natural language to provide better interpretability of our model. Additionally, we can study different genres of music over specific time periods to interpret and identify key audio or lyrical features for musical score generation or automatic lyrics generation.

References

[1] M. A. Nielsen, "Neural networks and deep learning," 2018.

[2] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[4] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5329–5336, 2019.

[5] F. Pachet and P. Roy, "Hit song science is not yet a science.," in *ISMIR*, pp. 355–360, 2008.

[6] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1. MIT press Cambridge, 2016.

[7] C. Donahue, H. H. Mao, Y. E. Li, G. W. Cottrell, and J. McAuley, "Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training," *arXiv preprint arXiv:1907.04868*, 2019.

[8] B. Brost, R. Mehrotra, and T. Jehan, "The music streaming sessions dataset," in *The World Wide Web Conference*, pp. 2594–2600, 2019.

[9] K. Jakubowski, S. Finkel, L. Stewart, and D. Müllensiefen, "Dissecting an earworm: Melodic features and song popularity predict involuntary musical imagery.," *Psychology of Aesthetics, Creativity, and the Arts*, vol. 11, no. 2, p. 122, 2017.

[10] L.-C. Yang, S.-Y. Chou, J.-Y. Liu, Y.-H. Yang, and Y.-A. Chen, "Revisiting the problem of audio-based hit song prediction using convolutional neural networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 621–625, IEEE, 2017.

[11] K. Frieler, K. Jakubowski, and D. Müllensiefen, "Is it the song and not the singer? hit song prediction using structural features of melodies," *W., Auhagen, C., Bullerjahn, R. von Georgi,(Eds.), Yearbook of music psychology*, pp. 41–54, 2015.

[12] L.-C. Yu, Y.-H. Yang, Y.-N. Hung, and Y.-A. Chen, "Hit song prediction for pop music by siamese cnn with ranking loss," *arXiv preprint arXiv:1710.10814*, 2017.

[13] A. Singhi and D. G. Brown, "Hit song detection using lyric features alone," *Proceedings of International Society for Music Information Retrieval*, 2014.

[14] K. Middlebrook and K. Sheik, "Song hit prediction: Predicting billboard hits using spotify data," *arXiv preprint arXiv:1908.08609*, 2019.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[16] X. Li, S. Chen, X. Hu, and J. Yang, "Understanding the disharmony between dropout and batch normalization by variance shift," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2682–2690, 2019.

[17] Q. Liu, M. J. Kusner, and P. Blunsom, "A survey on contextual embeddings," *arXiv preprint arXiv:2003.07278*, 2020.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[19] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual bert?," *arXiv preprint arXiv:1906.01502*, 2019.

[20] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. McDermott, "Publicly available clinical bert embeddings," *arXiv preprint arXiv:1904.03323*, 2019.

[21] B. van Aken, B. Winter, A. Löser, and F. A. Gers, "How does bert answer questions? a layer-wise analysis of transformer representations," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1823–1832, 2019.

[22] G. Zhang, C. Wang, B. Xu, and R. Grosse, "Three mechanisms of weight decay regularization," *arXiv preprint arXiv:1810.12281*, 2018.

[23] D. Bogdanov, N. Wack, E. Gómez Gutiérrez, S. Gulati, H. Boyer, O. Mayor, G. Roma Trepat, J. Salamon, J. R. Zapata González, X. Serra, *et al.*, "Essentia: An audio analysis library for music information retrieval," in *Britto A, Gouyon F, Dixon S, editors. 14th Conference of the International Society for Music Information Retrieval (ISMIR); 2013 Nov 4-8; Curitiba, Brazil.[place unknown]: ISMIR; 2013. p. 493-8.*, International Society for Music Information Retrieval (ISMIR), 2013.

[24] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," 2011.

[25] Y. Liu, H. T. Loh, and A. Sun, "Imbalanced text classification: A term weighting approach," *Expert systems with Applications*, vol. 36, no. 1, pp. 690–701, 2009.

[26] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. of the Int'l Conf. on Artificial Intelligence*, Citeseer, 2000.

[27] S. Vijayarani, M. J. Ilamathi, and M. Nithya, "Preprocessing techniques for text mining-an overview," *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7–16, 2015.

[28] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural*

language processing (EMNLP), pp. 1532–1543, 2014.

[29] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[30] I. Beltagy, A. Cohan, and K. Lo, "Scibert: Pretrained contextualized embeddings for scientific text," *arXiv preprint arXiv:1903.10676*, 2019.

[31] H. Xiao, "bert-as-service." https://github.com/hanxiao/bert-as-service, 2018.

[32] A. Singhi and D. G. Brown, "Can song lyrics predict hits," in *Proceedings of the 11th International Symposium on Computer Music Multidisciplinary Research*, pp. 457–471, 2015.

[33] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[34] E. Zangerle, R. Huber, M. Vötter, and Y.-H. Yang, "Hit song prediction: Leveraging low-and high-level audio features," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*, 2019.

[35] R. Mayer and A. Rauber, "Multimodal aspects of music retrieval: Audio, song lyrics–and beyond?," in *Advances in music information retrieval*, pp. 333–363, Springer, 2010.

[36] Y. Yu, S. Tang, F. Raposo, and L. Chen, "Deep cross-modal correlation learning for audio and lyrics in music retrieval," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 15, no. 1, pp. 1–16, 2019.

[37] Z. Obermeyer and E. J. Emanuel, "Predicting the future—big data, machine learning, and clinical medicine," *The New England journal of medicine*, vol. 375, no. 13, p. 1216, 2016.