FEATURE ENGINEERING AND CLASSIFICATION OF DIFFERENT SOUND WAVES

(CLASSIFY SONG GENRES FROM AUDIO DATA)

Project report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

Yasharth (191328)

Under the supervision of

Dr. Diksha Hooda

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Certificate

Candidate's Declaration

I hereby declare that the work presented in this report entitled **Feature engineering and classification of sound waves (Classify song genres from audio data)** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** submitted in the department of Computer Science and Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2022 to May 2023 under the supervision of **Dr. Diksha Hooda (**Senior Grade) professor in department of Computer Science and Engineering.

I also authenticate that I have carried out the above-mentioned project work under the proficiency stream Data Science.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Yasharth, 191328.

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Dr. Diksha Hooda

Assistant Professor (SG)

Computer Science and Engineering

Dated:

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ....

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: ___________________________ __**Department:** ________________ **Enrolment No** _________

Contact No. ______________________________**E-mail.** ______________________________________

Name of the Supervisor: ____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): ________________________

____

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	**Excluded**	**Similarity Index (%)**	**Generated Plagiarism Report Details** (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for his divine blessing makes us possible to complete the project work successfully. I am really grateful and wish my profound my indebtedness to Supervisor **Dr. Diksha Hooda, Assistant Professor (SG)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Machine Learning & keen interest of my supervisor in the field of “Classification of audio data” to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

I would like to express my heartiest gratitude to **Dr. Diksha Hooda,** Department of CSE, and lab assistants Mr Mohan Sharma and Mr. Ravi Raina for there kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straight forwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Yasharth, 191328

TABLE OF CONTENT

Title	**Page No.**
Certificate	i
Plagiarism Certificate	ii
Acknowledgement	iii
Table of Content	iv
List of Abbreviations	v
List of Figures	vi
List of Graphs	vii
List of Tables	viii
Abstract	ix
Chapter-1 (Introduction)	1-11
Chapter-2 (Literature Survey)	12-16
Chapter-3 (System Design and Development)	17-39
Chapter-4 (Experiment and Performance Analysis)	40-51
Chapter-5 (Conclusion)	52-53
References	54-55

List of Abbreviations

1. SVM : Support Vector Machine
2. CNN : Convolution Neural Networks
3. LMD : Latin American Music Database
4. MFCC : Mel-frequency Cepstral Coefficients
5. STFT : Short Term Fourier Transform
6. ZCR : Zero Crossing Rate
7. WVG : Weighted Visibility Graph
8. SDA : Stack Denoising Autoencoder
9. RA-TSM : Riemann Alliance Transfer Learning Modules
10. GCN : Graph Convolution Networks
11. BiLSTM : Bidirectional Long-term Short-term Memory
12. Hz : Hertz
13. PCA : Principal Component Analysis
14. ISMIR : International Society for Music Information Retrieval
15. GDR : Good Detection Rate
16. SLPP : Slow LPP
17. RBFK : Radial Based Function Kernal
18. ResNet : Residual Neural Network

List of figures:

1. Simple Convolution Neural Network Image
2. Machine Learning Pipeline
3. Label Encoding over class Labels.
4. Principal Component Analysis over dataset
5. a. Linearly Separable data b. Non-linearly separable data
6. Equation of SVM in different Dimensions
7. Gradient Descent
8. Decision Tree with root and internal nodes
9. Planer representation of SVM in different dimensions
10. LSTM internal Structure.
11. LSTM architecture.
12. Transfer learning diagram
13. Inception V3 developed by google.
14. Inception V3 Implementation
15. Inception V3 Model function
16. Structure of BiLSTM Model
17. Internal structure of BiLSTM
18. Attention Model
19. Dataflow diagram of the model.
20. SVM testing Accuracy.
21. Histogram Gradient Boosting testing accuracy
22. ResNet50 Model (pretrained model import)
23. A new model having first layer as ResNet50 Model
24. Transfer Learning model Summary Statistics
25. Transfer learning testing loss and accuracy measures

List of Graphs

1. Training Testing accuracy and loss of CNN Model.
2. Training Testing accuracy of LSTM Model.
3. Training Testing loss of LSTM Model.
4. Training testing accuracy and loss of Inception V3 Model.
5. Training testing accuracy of model BiLSTM Model.
6. Training testing loss of BiLSTM Model.
7. Training testing accuracy of BiLSTM with attention Model.
8. Training testing loss of BiLSTM with attention Model.
9. Comparison of losses of different models
10. Comparison of accuracy of different models

List of tables

1. Comparison of different models

ABSTRACT

With the rapid growth of multimedia technologies, a large number of music resources are now available online, leading to increased interest in classifying different music genres. The main objective of a music recommendation playlist is to identify a set of songs belonging to a similar genre. Machine learning, transfer learning, and deep learning concepts can be used to build a robust music classifier that can tag unlabelled music and improve the user experience of media players with music files. However, existing approaches in the past decade have several limitations, such as the manual extraction of features and traditional machine learning classification techniques, which impact the classification accuracy, particularly for multiclass classification problems and huge data sizes.

In this work GTZAN dataset has been used which has 1000 audio files of 10 different genres with two different parts 3 sec audio and 30 sec audio files. With this audio data STFT transformation have been used to generate images of audio signals. After pre-processing and feature extraction the data is visualized, and different machine learning mode have been applied upon the numeric data as well as image dataset. Models which are applied upon the numeric data are SVM, HGB, LSTM, BiLSTM, BiLSTM-attention model, where HGB performed well with accuracy score of 90.8% followed by SVM and then LSTM and BiLSTM had comparable results. Model applied on image dataset were CNN, Transfer learning (RESNET50 + CNN) where CNN performed well with accuracy score of 91.4%. After implementation of these models a comparative study has been done.

CHAPTER– 1

INTRODUCTION

1.1 Introduction

Music has its significant impact on enhancing focus at work and aiding in rapid stress relief, music plays a significant role in human life. Additionally, it aids in eradicating bad emotions and sentiments like loneliness, melancholy, and depression, which can significantly improve one's physical, mental, emotional, and spiritual wellbeing.

In recent years, powerful deep learning approaches have been utilised to explore the deep aspects of music because manual music feature extraction has limited applications and is less reliable. Users might be interested in listening to a particular song in a particular genre with a particular kind of feeling, for instance, and in that case, a music label is definitely necessary to obtain that individual music. In order to give users deeper insights material, many subscription and recommendation frameworks necessitate expertise about the necessary music categories. Due to its many different components, including rhythm, melody, and harmony, music is immensely diverse.

Because comprehending music in its actual form needs some depth of knowledge, all music media platforms utilise text labels to categorise or search for music. Consequently, one crucial area of music information retrieval is the genre classification[4].
A music genre is a wonderful term that distinguishes music and elaborates on the numerous categories that listeners are particularly interested in. Music genre categorization is always done by automated approaches, which gradually and carefully become an appealing study subject for scholars. Manual labelling procedures require professionals with a strong musical experience and take a lot of time.

The following is a discussion of some well-known and current work in the field of music genre classification[4]. Spectrograms were used to identify music using a convolutional neural network (CNN), and a high success rate of 92 percent on the LA Music Database was reported by the researchers (LMD). One of the frequent issues with this categorization of musical genres is that in certain instances, the validation findings need to be improved, and data augmentation should also be taken into account.

Genre is defined as a category of artistic composition characterized by similarity in form, style or theme. One way we can tell that two tracks are similar is by looking at the genre it belongs to. Two songs belonging to the same genre usually have more similarities than two songs belonging to different genres. Genre-based classification can be done by extracting the information that can be obtained from the raw data.

That's the million-dollar question through music streaming services such as Apple Music, Pandora, Spotify and YouTube to name a few. As engineers, we consider a track to be around 1.3 million data points depending on the length. To classify songs as similar, we need to look at these data points and it happens it's increasingly difficult to do this with raw data, especially when new songs are introduced daily. Also, similarity between songs is difficult to define as the parameters used describing how similar two songs are inherently subjective and not easily translated algorithm.

The music business has seen several changes as a result of the widespread usage of the Internet. The widespread usage of online music listening and purchasing platforms, music copyright regulation, genre classification, and music suggestion are a few examples of this evolution. Thanks to the evolution of music streaming services, listeners may now access millions of songs through different music listening sites like Spotify and last.fm at any time, anywhere.

It takes a lot of time to manually tag the genres of a big data set. Furthermore, due to aspects including cultural, historical, geographical, instrumental, and rhythmic patterns, musical genres lack specific definition and bounds. As a result, accurate labelling needs specialist knowledge and is very subjective. In recent years, powerful deep learning approaches have been utilised to investigate the deep aspects of music because manual music feature extraction has limited applications and is less reliable. Users could be interested in listening to a certain song in a particular genre with a particular sort of feeling, for instance, and in that case, a music label is definitely necessary to obtain that particular music. In order to give customers more in-depth material, many subscription and recommendation frameworks need knowledge about the appropriate music genres. Due to its many different components, including rhythm, melody, and harmony, music is immensely diverse.[5]–[7]

Zero-crossing speed, the Mel frequency, the spectral centroid, the spectral contrast, the spectral bandwidth, and the spectral roll off The characteristics that could be extracted from the music were discovered to be Cepstral Coefficients-MFCC. One of the greatest deep learning approaches, Convolutional Neural Network-CNN, has also been used to classify musical genres and provide music recommendations. CNN is used as follows to categorise music genres and identify song similarity: inserting a spectrogram into CNN, extracting the dense layer output, and applying STFT to a recorded audio. To compare performance results, all classification and recommendation algorithms were used to the GTZAN [2]dataset. The approaches for feature extraction, music recommendation, and musical genre categorization were established in the second phase.

Several significant feature extraction techniques employed in this work are discussed in this section. Several categories may be used to categorise feature extraction techniques in musical signal processing. Digital signal processing, namely in the time and frequency domain, is one of them. Another successful method for feature extraction is the use of statistical descriptors like mean, median, standard deviation, etc. Each of the methods described below is used to divide the raw music signal into N windows, and each of these operations is performed N times.

The useful features of music used in machine learning are: -

1. ZCR
2. Spectral Centroid
3. Spectral Roll-off
4. MFCC

ZCR : The pace at which the signal's sign changes over the course of an audio frame is known as the zero-crossing rate (ZCR).

Spectral Centroid : A function called spectral centroid is employed in the frequency domain to denote a position where the gravitational centre of frequencies vs. frequency stack is located.

Spectral Roll-off : The spectral roll off frequency is the normalised frequency at which the sum of the low-frequency sound power values travels throughout the whole power spectrum at a specific speed. It may be summed up as a frequency value that corresponds to a certain spectrum distribution ratio. In typically, this percentage is 85%.

MFCC: A limited group of characteristics known as The overall contours of a spectral envelope are described by the Mel Frequency Cepstral Coefficients (MFCCs). It might be viewed as a timbre characteristic. For the benefit of the human hearing system, the cepstral coefficients are modified using the MFCC. Cepstral coefficient scales are linear. However, for frequencies below 1KHz and above, one may hear a linear scale and a logarithmic scale, respectively. One of the most common capabilities of speech and speaker recognition systems is the regular use of MFCC. The MFCC phases include frame blocking, windows, quick Fourier Transform, Mel-Frequency Wrapping, and the spectrum. The total number of N-coefficients is an additional parameter.

1.2 Problem Statement

With terabytes of music files, related information, and internet streaming services in today's enormous datasets, music categorization is a difficult challenge for many applications.
It is really challenging to manage such a huge database and find the requested item. The appropriate storage and retrieval of music data will benefit from accurate classification. When classifying a dataset, attributes like genre, vocalist, and mood are very significant..

It might be time-consuming to manually tag the genres in a big data source. Additionally, due to aspects including cultural, historical, geographical, instrumental, and rhythmic patterns, musical genres lack specific definition and bounds. As a result, it is very subjective and requires specialist knowledge for proper labelling. In recent years, powerful deep learning approaches have been utilised to investigate the deep aspects of music because manual music feature extraction has limited applications and is less reliable. Users could be interested in listening to a certain song in a particular genre with a particular sort of feeling, for instance, and in that case, a music label is definitely necessary to obtain that particular music. In order to give customers more in-depth material, many subscription and recommendation frameworks need knowledge about the appropriate music genres. Due to its many different components, including rhythm, melody, and harmony, music is immensely diverse.
The validation findings in certain cases need to be improved, and data augmentation should also be taken into account. These are only a few of the usual issues this music genre categorization faces. Considerations for data quantity, thorough feature evaluation, improved model regularisation, and development of complex network structure should all be made while developing new methods for categorising music genres. In light of these factors, Traditional machine learning methods don't seem to be as effective or flexible as deep learning frameworks, sparse representation frameworks, and transfer learning frameworks. Here in study, 5 novel methodologies are given for best in music categorization genres.

Attempts to categorize music by extracting audio features from a sample have had mixed results. Some categories, such as classical music, are easy to identify, but attempts to distinguish between different types of popular music give poor results. Part of the problem is that people also disagree when classifying music.

1.3 Objective

Making song selection easier and quicker is the aim of automated music classification. If you must manually categorise songs or music, you must first listen to many tracks before selecting a genre. This takes a lot of time and is challenging. You may quickly identify essential information, such as trends, popular genres, and performers, by automating the categorising of music. The very first step in this route is determining the various musical genres.
Objectives for the project are –

- Transformation of numeric data to image dataset.
- Feature engineering of the dataset.
- Applying different machine learning model to the dataset
- Comparison of the accuracies of the models

1.4 Methodology

Each audio recording in the dataset we utilised for this study has pre-defined classifications. As a result, we classified the genre using a variety of supervised machine learning approaches. All the musical feature information came from the open GTZAN[2] dataset. In this data collection, 1000 audio recordings of 30 seconds each are categorised into one of ten different genres and provided as.au files. Our data set was increased to 8000 two-second clips by sampling a continuous 2-second window from each clip at four different random positions. Since this data was sampled at 22,050 Hz, the raw audio input had 44,100 characteristics. To reduce the amount of functions, we have limited our windows to two seconds. At 44100 features, the size of the feature or parameter space and the duration of the audio sample were found to be completely balanced.

So, following pre-processing, our input has the following shape: (8000,44100), where each feature represents the amplitude at a certain timestep out of the 44100. Additionally, we employed 100 samples of raw data for our test and cross-validation sets, respectively. By transforming unprocessed audio into Mel-spectrograms, we also experimented with pre-processing our data. This has resulted in a notable improvement in performance across all models. Because they properly depict how humans hear sound, Mel-spectrograms are a widely used technique for imaging sound (i.e., in log frequency). Raw audio must be transformed into a Mel-Spectrogram by applying short-time Fourier transforms to sliding audio windows, which are typically 20 ms wide. This is calculated as the product of signal x[n] with window w[n], amplitude-frequency axes, and shift results as m.

$$STFT\ \{X[n]\}\ (m,\omega) = \sum_{n} \omega\ *\ x\,[n]\ *\ [n-m]\ *\ e^{-j\omega n}$$

Sliding DFT techniques are used in practise to compute these more quickly. The frequencies f are then transformed by to be translated to the Mel scale.

$$m = 2595\ \log_{10}\left(1+\frac{f}{700}\right)$$

The discrete cosine transform of the result, which is frequently used in signal processing, is then used to obtain the final result is our Mel-spectrogram.

In our instance, we utilised the “Librosa” package from pip repository and decided to use 64 Mel-Ceptral-bins, a window length of 512 samples, and 50% overlap as between the song music windows. We next go on to log scaling using the log(X2) formula based on prior academic achievement with similar transformations.

In addition to the aforementioned pre-processing of our data, two of our models' dimensionality was decreased using principal component analysis (K-NN and SVM). PCA[8] is a type of feature extraction and analysis that projects the data or variance in a low dimensional feature in an effort to low dimensional the data. This was accomplished by first standardising the mean and variance of the Mel-spectrograms. PCA[8] aims to maintain as much variation as feasible with m samples of x(i), the (u) as our unit length.

$$\frac{1}{m}\sum_{i=1}^{m}\left(x^{i^T}u\right)^2 = \frac{1}{m}\sum_{i=1} x^i x^{i^T}$$

Since the mean of our data is zero, obtaining the major eigenvector of the covariance matrix is equal. Empirically, we found that projecting our data into the region covered by the first 15 eigenvectors gave us the best results when the number of dimensions was reduced to 15. Scikit-learn was used in its implementation.

We utilised the k-nearest neighbour technique after applying PCA see to reduce the dimensionality to 15 features. For each case, predictions are formed by identifying the training examples that are most similar to the test or cross-validation example2 that has to be classified and predicting the label most commonly appears among those examples' real-world labels We found via trial and error that weighting each neighbor's label by distance with k = 10 yielded the best accuracy.

Our data piece is expressly identified as x, let x(1), . . . , where x(10) represents x's ten closest neighbours, those that yield the highest value at ||x - x||^2 where ||w||^2 denotes the Euclidean separation between the two locations. Next, we select the weights wi.

$$\omega_i \propto \left\|x - x^i\right\|_2 , \sum_{i=1} 0\, \omega_i = 1$$

Finally, we return

$$\arg\max\ y = \Sigma\omega_i \mid y = y^i \mid$$

We trained the SVM classifier after PCA was used to reduce the number of dimensions. SVMs are the best edge classifiers available, and they can employ kernels to uncover more intricate patterns in the input data. This equates to a discovery because there is no linear separability in our data.

$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \quad \text{s.t. } y^{(i)}(\sum_j \alpha_j K(x^{(j)}, x^{(i)}) + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

where x(i) are instances, j as weights and b as deviations. C as the penalty parameter, and 1 I the functional range for example K as our kernel function is K : R n × R n → R. In a conventional SVM. This function is equivalent to the product of the these vectors x(j) and other vector x(i), however in our instance, we utilise the RBF-kernelization:

$$K\left(x^{(j)}, x^{(i)}\right) = \exp\left(-\frac{\left\|x^{j} - x^{i}\right\|_2^2}{2\sigma^2}\right)$$

This kernel, which is sometimes referred to as the Gaussian kernel occasionally, corresponds to a feature space with an unlimited number of dimensions and a connection to the Euclidean distance. Sequential minimization optimization is frequently used to minimise this function as a whole. Scikit-learn was employed in its implementation.

This was our most complex model, with 3-fully connected layer of ReLU as our activation function and Softmax function in last output layer, and cross-entropy where loss feeding into five convolutional layers, each with their own max pool and regularisation. The majority of the equations are provided above, and the following diagram shows our architectural layout:

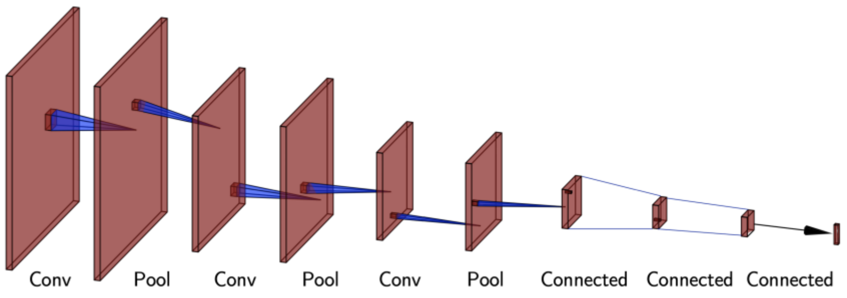


Fig. 1 : Simple Convolution Neural Network Image

Convolutional windows are used in this method to scan the input data and output the window's element sums. The max_pool layer receives this and chooses the maximum element from a different window. The model mentioned in Section 4.1 is then given the output. It was developed with Keras and TensorFlow.

Several significant feature extraction techniques employed in this work are discussed in this section. Several categories may be used to categorise feature extraction techniques in musical signal processing. One of these is digitalized signal pre-processing, namely in the time domain converts into frequency domain. Another successful method for feature extraction is the use of statistical characteristics like mean, median, standard deviation, etc. The raw music signal is divided into n number of windows using each of the ways listed below, and each of these processes is repeated n times.

The pace at which the signal's sign changes over the course of an audio frame is known as the zero-crossing rate (ZCR). In other words, it is the number of times the signal's value switches from positive to negative to the other way around, divided by the number of frames. Equation (1) defines ZCR as follows:

$$z(i) = \frac{1}{2w_L}\sum_{n=1}^{w_L}\left|\operatorname{sgn}\left[x_i(n)J_{-\operatorname{sgn}[x_i(n-1)]}\right]\right|$$

$$\text{Where: } \operatorname{sgn}[x_i(n)] = \begin{cases} 1, x_i(n) \geq 0 \\ -1, x_i(n) < 0 \end{cases}$$

Spectral centroid is a function is used in the frequency domain to indicate a point centre of gravity of frequencies v frequency stack. With their magnitudes acting as weights, it is determined as a weighted average of the Fourier-transformed frequencies present in the signal:

$$\text{Centroid} = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)}$$

where x(n) is the weighted values of frequencies, or magnitude, of bin number n, and f(n) stands for the bin's centre frequency. Since it accurately predicts the "brightness" of a sound, the spectral centroid is commonly used in digital audio and music processing as an automated gauge of musical-scales and timbre.

The normalised frequency where the whole of the low-frequency sound power values moves over the whole power spectrum at a particular speed is known as the spectral roll off frequency. It may be summed up as a frequency value that corresponds to a certain spectrum distribution ratio. In typically, this percentage is 85%.

A limited group of characteristics known as the **Mel Frequency Cepstral Coefficients (MFCCs)** characterise the general contours of a spectral envelope for sound waves. It might be regarded as a timbre features. The MFCC is used to modify the cepstral coefficient for the benefit of the human audible range. Scales for cepstral coefficients are linear. However, one may hear a linear scale for frequencies below 1KHz and a logarithmic scale for frequencies beyond. MFCC is frequently utilised in speech as well in voice recognition systems, making it one of their most prevalent functionalities. Frame blocking, windows, rapid Fourier Transforms and Mel Frequency Wrapping, and the whole band of spectrum are MFCC stages. Another parameter is the number of N-coefficients. N was supposed to be 13 in this investigation. MFCC may be explained in further depth by its coefficients.

Short-Time Fourier Transform (STFT): The audio data is subjected to a Short-Time Fourier Transform before being sent to the connection. A 1-D time series becomes a 2-D frequency domain as a result. This transform has the size of window parameter set to 2048 and the step length parameter set to 1024.

CHAPTER – 2
LITERATURE SURVEY

The purpose of this study is to categorise and advise against using the acoustic characteristics of songs as determined by electronic signal processing techniques along with convolutional neural network. The reports and researches has been attained in two stages: figuring out how to gather the data that will be utilised in the suggestion and developing a service that suggests music in response to user queries. In the beginning, feature extraction was carried out using digital signal processing techniques, and CNN was taught as a backup feature extraction approach. The best categorization method and the best referral results are then determined by using the acoustic characteristics of the music. According to the data given in the preceding tables, SVM outperformed other approaches in terms of classification accuracy. Additionally, there was a very slight performance difference when the window's type and size were changed. When it comes to feature effects classification performance, MFCC outperforms other approaches. Deep learning techniques shown that there is no meaningful performance chance to categorise musical genres. Next, SVM outperformed the CNN algorithm in terms of success rate. Music genre recommendations can be utilised as a solution request for music recommendations as there isn't a destination music recommendation metric. The outcomes of recommendations for some music genres, such as classical music, show that music referral is quite successful whereas performance in some species declines. Future research will focus on various deep learning architectures and high-level feature extraction techniques. We will concentrate on this topic, particularly because deep learning approaches demand high-performance computing infrastructures.

The testing and validation sets are carried out in an 8:1:1 ratio for this training random distribution data set. The other dataset is the ISMIR 2004, which comprises of musical compositions having 1458 datapoints. The training data and test data are specified as having a 50/50 split between the training file and the test file (Cano et al., 2006). It comprises of six distinct musical styles, including classical, blues, electronic, world, pop, and punk music. Since not all tracks have artist information accessible, no performer filter has been applied to this dataset. The total number of tracks for each class in this dataset is not equal because this dataset is not balanced. MagnaTagA-Tune is the third dataset that was used in this article. The audio files is presented here in the mp-3 extesion with a sampling frequency of 16 kHz and a transmission rate of 32 kbps. The sample data were divided into percent of 70 for training and validation and percent of 30 for testing. About 25,877 audio segments in this collection have multi-label genre, mood, and instrumentation annotations. Each segment in this data file is around 30 seconds long and there are many, many portions per song. Only the top 50 tags from this dataset are used in this study. Because a large correlation between performance metrics may suggest surplus, many information metrics were not examined, with the exception of F1-score, Good Detection Rate (GDR), accuracy, sensitivity, specificity, and mistake rate.

The categorization of musical genres is an intriguing and challenging subject with several applications in the enormous datasets and internet streaming services of today. In this study, we examined, using a publicly available dataset, the accuracy rate of several music genre categorization systems. We contrasted the outcomes using a two-layer neural network with the input signals' power spectrograms and raw audio data. investigated several learning strategies for dimensionality reduction, including PCA, SLPP, and SR. The classification outcomes were then contrasted using custom-made audio feature vectors, such MFCCs. According to experimental findings, 1-SVM classifiers and MFCC are equally effective at classifying music genres as neural networks based on power spectrograms. Future research will focus on stacked and convolutional neural networks, among other network topologies are sensitive of the network for classify task on it.

Here in this project, We looked at a novel multivariate integration of autoregressive functions and the integration of temporal aspects of short-term data in a music genre categorization challenge. Dependencies between feature dimension and correlation in the time domain were built into the schema. Two novel functions, DAR and MAR, were created by this technique and have been meticulously detailed and compared to the functions of the other temporary function integration approaches. Four distinct classifiers were employed to evaluate them on two different datasets, and the successful MFCC functions were used as short-term feature representation. The framework may be used to several short-term function types. Particularly, it was discovered that the features of MAR performed substantially better than existing functions, while DAR functions also outperformed FC and the fundamental Mean, Var. On a dataset of 11 musical genres, we conducted human genre classification studies and discovered that the average human test's accuracy was only around 18% higher than the best-performing MAR technique. Future study may look at other indices for a generic multivariate formulation of AR, which enables more flexible modelling of short-term functions on longer time scales, as well as non-stationary window handling techniques. Finally, it should be mentioned that the temporal feature integration framework is adaptable to various MIR-related fields.

This work presents the results of an extensive study of several LSTM[9] architectural variations. We come to the conclusion is we have most popular LSTM[9] architecture performs quite well for ML problems and on various datasets. None of the eight adjustments that were looked at significantly boost performance. However, in our trials, some adjustments, without significantly hurting performance, LSTM was simplified using techniques like adding input and forget gates (CIFG) or deleting peephole connections (NP). These two variations are also appealing since they lower the complexity and number of parameters required to run LSTM. The most important parts of the LSTM block are the output enable and forgotten gate functions. Performance is considerably reduced if any of them are removed. The output activation function, according to our theory, is required to stop an unconstrained cell state from spreading across the network and impairing learning. Given that the cell state of the LSTM variation of the GRU is constrained by the coupling of the input and forgetting gates, this would explain why it can function pretty well without it. The learning rate and network size are the two most crucial hyperparameters, as would be predicted. Interestingly, however, it was discovered that the usage of momentum was not significant in our online gradient descent system. For TIMIT, Gaussian noise on the inputs was shown to be mostly beneficial for some datasets, but detrimental for others.

Table 1: Comparison of different models

Title	Year	Dataset	Models	Performance
Holistic Approaches to Music Genre Classification using Efficient Transfer and Deep Learning Techniques	2022	GTZAN [1], ISMIR [2] 2004 and MagnaTagATune	Combining Graphical Convolution Network, Weighted Visibility Graph-based Elastic Net Sparse Classifier, and Bidirectional Long Short-Term Memory (BiLSTM) [BAG]	Since using such combination of advance techniques the computational accuracy is hindered.
Deep Neural Networks: A Case Study for Music Genre Classification	2015	Uni-Dortmund	SVM, [3]l1-SVM in combination with MFCCs, DNN with manifold learning.	Accuracy - 38.4%
Music Genre Classification Using Frequency Domain Features	2018	GTZAN, Ballroom, Uni-Dortmund	SVM, Random Forest	Accuracy - 80.1%

CHAPTER - 3
SYSTEM DEVELOPMENT

3.1 System Design

We have used GTZAN[2] dataset as our primary dataset for this machine learning project.

3.1.1 About the Dataset [GTZAN][2]

Music Genre Classification of Audio Signals, a well-known work on genre classification by P. Cook with collab withs G. Tzanetakis, was presented in 2002 and appeared in IEEE Transactions on Speech and audio processing. Here we are with 1000 audio recordings with a total runtime of 30 seconds that make up the GTZAN[2] Genre Dataset are represented by the dataset. Ten different genres, each containing 100 tracks, make up the dataset. In.wav format, all songs are 22050 Hz Mono 16-bit audio files. With only one line of Python code, you can instantly import the GTZAN[2] Genre dataset using the free and open-source Active loop Hub module.

The benchmarked music genre categorization dataset GTZAN[2] is utilised in machine learning and data science procedures.

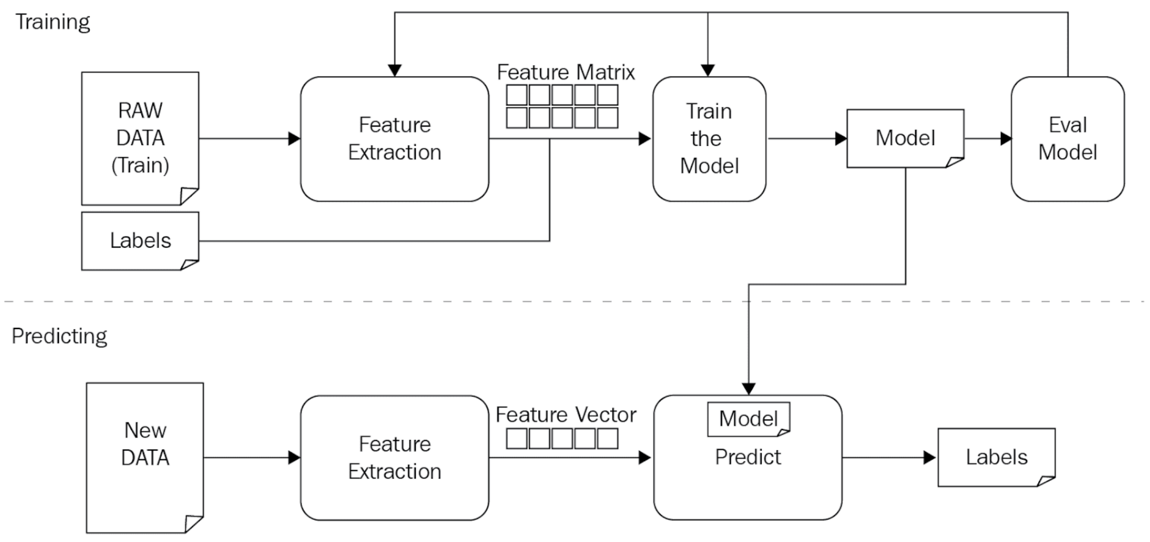
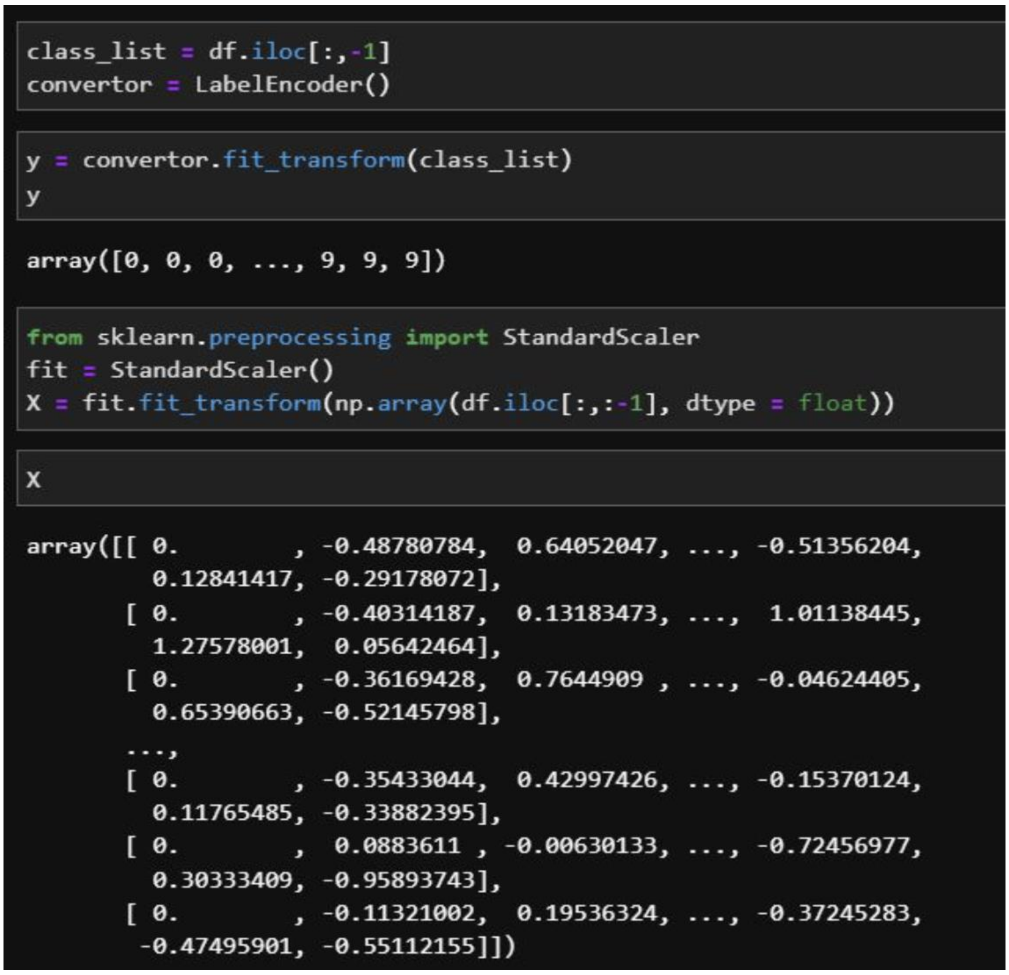


Fig 2: Machine Learning pipeline

3.1.2 Feature extraction

Reducing the size of resources required to explain a big data collection is the goal of feature extraction. One of the biggest issues with sophisticated data analysis is the sheer volume of variables. A substantial amount of memory and processing power are often needed for an analysis with many variables, and the classification algorithm may also overflow on training examples and perform badly on fresh samples. A way of creating combinations of variables to get around these issues while still accurately characterising the data is known as feature extraction. The key to effective model creation, in the opinion of many machine learning specialists, is appropriately optimised feature extraction.



```
class_list = df.iloc[:,-1]
convertor = LabelEncoder()
```

```
y = convertor.fit_transform(class_list)
y
```

```
array([0, 0, 0, ..., 9, 9, 9])
```

```
from sklearn.preprocessing import StandardScaler
fit = StandardScaler()
X = fit.fit_transform(np.array(df.iloc[:,:-1], dtype = float))
```

```
X
```

```
array([[ 0.        , -0.48780784,  0.64052047, ..., -0.51356204,
         0.12841417, -0.29178072],
       [ 0.        , -0.40314187,  0.13183473, ...,  1.01138445,
         1.27578001,  0.05642464],
       [ 0.        , -0.36169428,  0.7644909 , ..., -0.04624405,
         0.65390663, -0.52145798],
       ...,
       [ 0.        , -0.35433044,  0.42997426, ..., -0.15370124,
         0.11765485, -0.33882395],
       [ 0.        ,  0.0883611 , -0.00630133, ..., -0.72456977,
         0.30333409, -0.95893743],
       [ 0.        , -0.11321002,  0.19536324, ..., -0.37245283,
        -0.47495901, -0.55112155]])
```

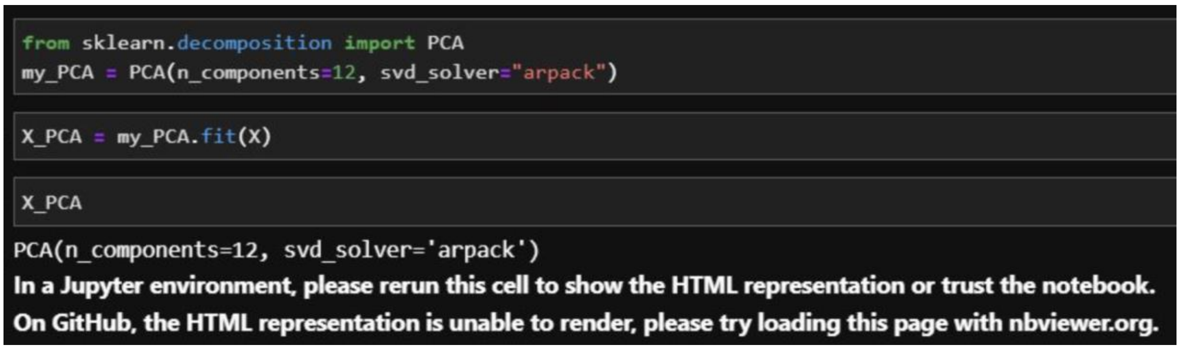
Fig. 3: Label Encoding Snippet over Class labels

Using application-specific feature sets, which are often developed by a professional, can improve results. The technique of feature engineering is one of these. In contrast, broad dimensionality reduction methods like:

1. Principal component analysis
2. Multifactor dimension reduction
3. Nonlinear dimension reduction
4. Semi-definitive embedding
5. Autoencoder

Here we have used Principal component analysis for dimensionality reduction
A common method for analysing huge data sets with many dimensions or characteristics per observation is principal component analysis (PCA), which improves the data's interpretability while keeping the greatest details possible and enables the display of data in several dimensions. According to its formal definition, A statistical technique for reducing the dimensionality of a data collection is PCA. The data are linearly converted into a new coordinate system in order to do this, allowing for the expression of (the majority of) data changes in terms of fewer dimensions than the original data. Studies commonly display data in two dimensions and visually pinpoint collections of closely linked data points using the first two major components. Principal component analysis is used in several fields, including population genetics, microbiome study, and atmospheric science.

PCA is used to create prediction models and for exploratory data analysis. In order to acquire lower dimensional information while keeping as much variety in the whenever possible, It is widely applied to reduce projections of each data point's dimensionality to just the first few major components. The first primary component may be defined as the route that maximises the variance of the forecasted data. The ith principal component may be seen as the path that is orthogonal to the first i-1 principal component analysis and maximises the variance in inputted projected data.



```
from sklearn.decomposition import PCA
my_PCA = PCA(n_components=12, svd_solver="arpack")

X_PCA = my_PCA.fit(X)

X_PCA

PCA(n_components=12, svd_solver='arpack')
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.
```

Fig. 4: Principal Component Analysis over dataset

3.1.3 Background

We have used various machine learning models such SVM, logistic regression, SGD-classifier, tree-based models, ensemble-based model like-random forest, HGB classifier.

Simple neural network, LSTM, and concept of transfer learning.

More elaboration about the models is done below-

SVM - Simple supervised machine learning methods like the Support Vector Machine (SVM) can be applied to classification or regression problems. Although it is frequently It is most frequently used in classification, which is particularly beneficial for regression. SVM essentially locates a hyper-plane that creates a separation between various types of data. In two-dimensional space, this hyper-plane is nothing more than a line. The total number of features and characteristics in the dataset, N, is used to map each dataset item into an N-dimensional space for SVM. Then, the data should be divided using the ideal hyperplane. You must be aware by now that because to the nature of SVM, it can only do binary classification (i.e. choose between two classes). For multi-class issues, there are several strategies that may be applied. Multi-class issues with a support vector machine We may develop a binary classifier for each data class so that SVM can be applied to multi-class issues.

Each classifier's two outputs will be.

1.The data point is a member of any label

2. The data point does not fall within that category.

To execute multi-class bracketing, for instance, For each fruit in a class of fruits, we have created a dual classifier. To determine if anything IS a mango or IT IS

NOT a mango, there will be a double classifier for the "mango" class, to use an example. The classifier with the greatest score is the one selected for the SVM. SVM for difficult non-linearly divisible data When dealing with data that is linearly divided, SVM performs brilliantly. Data is considered to be linearly separable if it can be grouped together in a graph and separated into classes along a straight line.

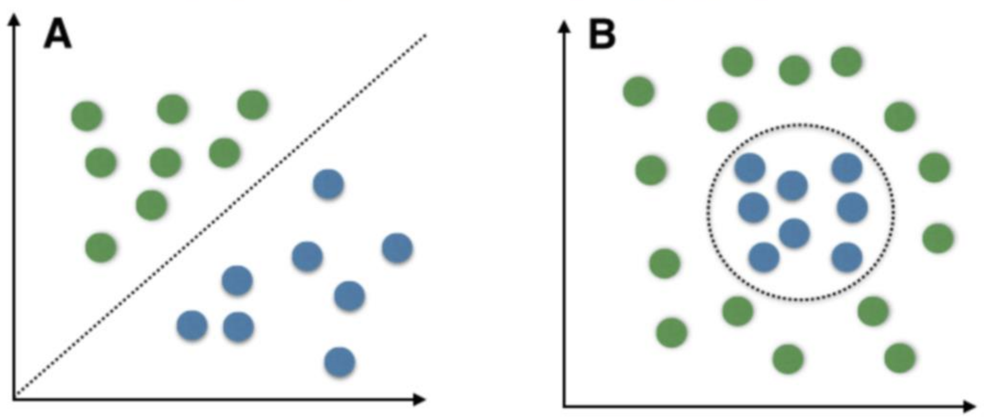


Fig 5: A Linearly Separable Data B: Non-Linearly Separable Data

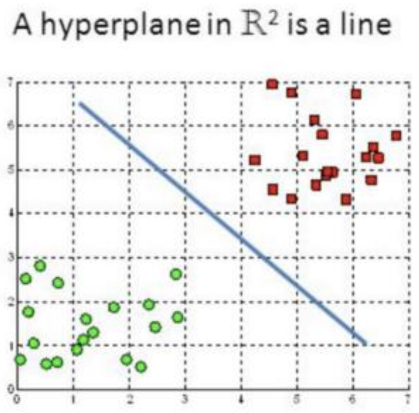
We use Kernelized SVM for data that is not linearly divisible. Consider a set of data that isn't linearly partitioned into dimensions. If we can translate the data into two confines, it will be linearly divided into those two confines. To do this, each 1-D data point is translated to a corresponding 2-D ordered brace. Therefore, for any non-linearly divisible data in any dimension, we can simply collision the data to an advanced dimension and make it linearly commutable. Unquestionably large and pervasive, this shift. A kernel is just a comparison of how similar two sets of data. The kernel function in a kernelized SVM displays the level of similarity between the points in the newly converted point space given two data points in the original point space. There are several other kernel functions, but only two are widely used. Kernelized SVM is used by Radial Base Function Kernels (RBFKs) for data that is not linearly divisible. Let's say we have some data that is not linearly divided in one dimension. The data will be linearly divided in two confines if we can transform it into two confines. To do this, each 1-D data point is translated to a corresponding 2-D ordered brace. Therefore, for any non-linearly divisible data in any dimension, we can simply collision the data to an advanced dimension and make it linearly commutable. Unquestionably large and pervasive, this shift. A kernel is just a comparison of how similar two sets of data are. The kernel function in a kernelized SVM displays the level of similarity between the points in the newly converted point space given two data points in the original point space. Although the kernel has many active functions, only two are really often used.

Radial Base Functions Kernel (RBF) As seen in the graph below, the similarity between two points in the converted point space is proportional to the distance between the vectors and the original input space. The dereliction kernel in SVM is RBF. As seen in the graph below, the similarity between two points in the converted point space is proportional to the distance between the vectors and the original input space. RBF serves as SVM's dereliction kernel.

$$K(x, x') = \exp(-\gamma \|x - x'\|)$$

Polynomial Kernel: The higher degree, a second parameter needed by the polynomial kernel, controls the complexity of the model and how computationally demanding the transformation in polynomial kernalization.

The two sets of data points might be split using a variety of different hyperplanes. Finding an aircraft with the greatest periphery—the most space between data points from both classes—would be ideal. The classification of future data points may be done with more assurance by maximising the periphery distance.



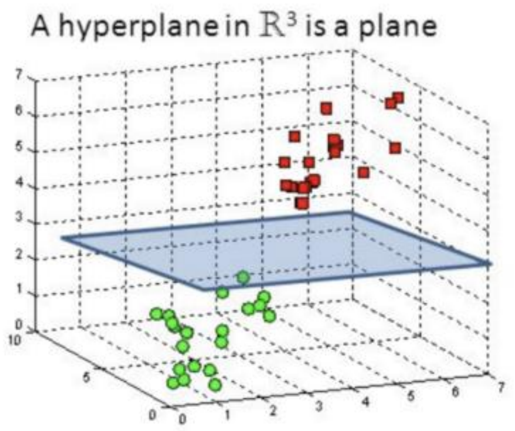


Fig 6: Equation of SVM in different Dimensions

The size of the hyperplane depends on the quantity of features. However, the hyperplane is also just a line if there are just two input features. Still, the hyperplane collapses into a two-dimensional plane if there are three input features. When there are more than three features, visualisation becomes more challenging.

Logistic Regression - A supervised machine literacy technique called logistic retrogression completes double bracket tasks by estimating the likelihood of an outcome, an occurrence, or an observation. The model outputs a double or dichotomous result with the options yea/no, 0/1, or true/false as the only two potential outcomes.

Data is divided into classes and the link between one or more independent variables is examined using logistic regression. It is heavily utilised in prophetic modelling, where the model calculates the precise likelihood of whether a case falls under a given order or not.

As an example, 0 represents the negative label or class while 1 denotes the positive label or class. In double bracket issues where the outgrowth variable discloses any of the two orders, logistic retrogression is often applied (0 and 1).

Some exemplifications of similar groups and cases where the double response is anticipated or inferred are

1. Determine the probability of heart attacks with the help of a logistic model, medical interpreters can determine the relationship between variables similar as the weight, exercise, etc., of an individual and use it to prognosticate whether the person will suffer from a heart attack or any other medical complication.

2. Possibility of enrolling into a university operation aggregator can determine the probability of a pupil getting accepted to a particular university or a degree course in a council by studying the relationship between the estimator variables, similar as GRE, GMAT, or TOEFL scores.

3. Relating spam emails Dispatch inboxes are filtered to determine if the dispatch communication is promotional/ spam by understanding the predictor variables and applying a logistic retrogression algorithm to check its authenticity.

$$f(x) = \frac{1}{1 + e^{-x}}$$

SGD Classifier - The term Stochastic Gradient Descent- Classifier (SGD-Classifier) might mislead some stoners into believing that SGD is a classifier. But that's not the case! An SGD-optimized direct classifier is SGD Classifier (SVM, logistic regression). There are two distinct generalisations here. While Logistic Retrogression or SGD is an optimization method, while Direct Support Vector Machine is a machine learning algorithm/model. You can imagine that an optimization system minimises and maximises a loss function defined by a machine learning model.

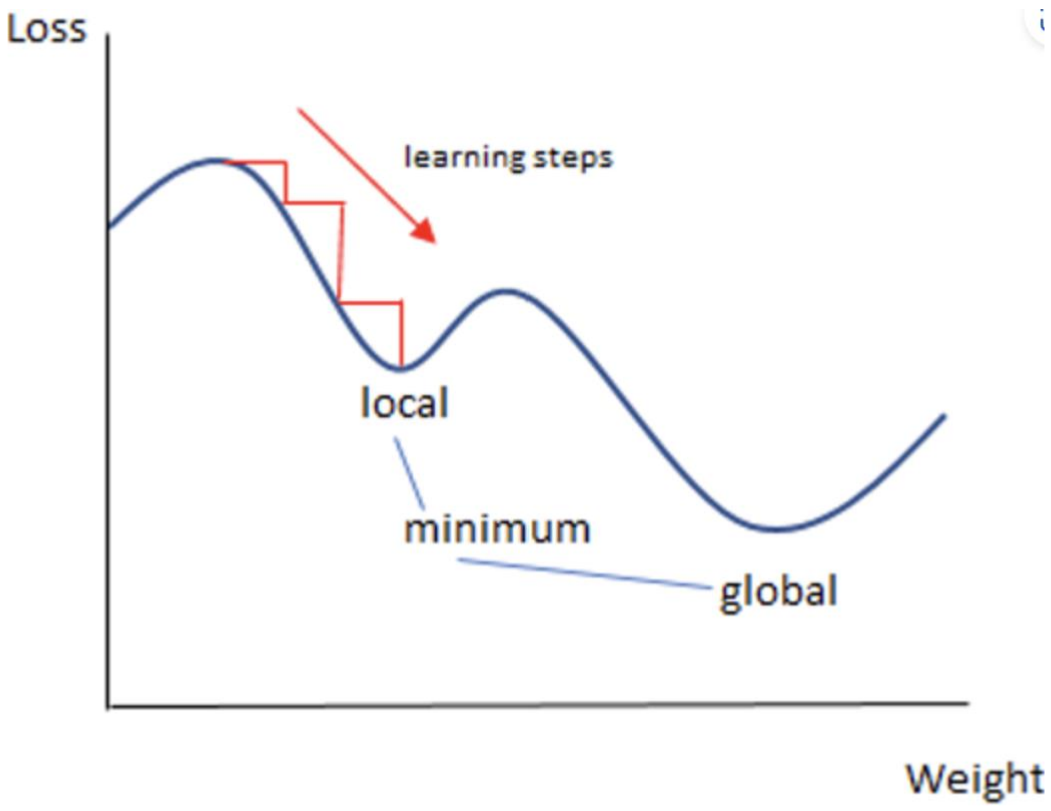


Fig 7. Gradient Descent

In essence, grade descent is a method for reducing a cost function. One of the most popular and by far the most effective algorithms for optimization typical method for optimising neural networks is grade descent. However, we can also utilise these methods to improve our direct classifier in a similar way to that of direct Support Vector Machines and logistic regression.

There are three well given types of grade-decent

1. Batch Gradient Descent
2. Mini-Batch Gradient Descent
3. Stochastic Gradient Descent

Using the entire information, batch grade descent calculates the grade in order to determine the minimum in the magnet's receptacle.

The grade is computed using stochastic grade descent (SGD) on a single sample.

In the end, mini-batch grade descent executes an update for each mini-batch of n training exemplifications, combining the best of both worlds.

Decision Tree - A non-parametric supervised literacy technique called a decision tree is used for bracket and retrogression tasks. It features a tree-like, hierarchical structure made up of internal bumps, splint bumps, branches, and a root knot.

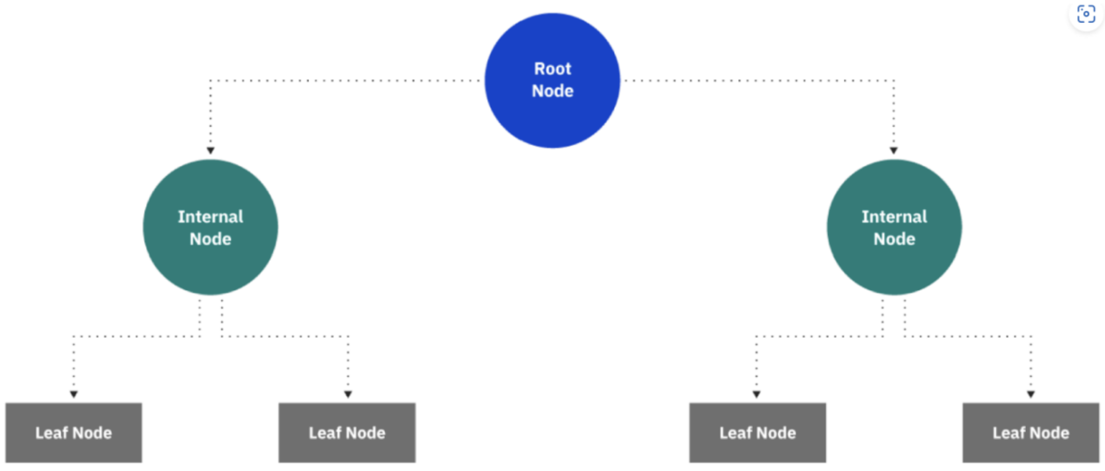


Fig. 8: Decision Tree with root and internal nodes

Types of decision tree are –

ID3: ID3, or "Iterative Dichotomise 3," is a method that uses entropy and information gain as measures to assess potential splits. Ross Quinlan is credited with its creation.

C4.5: This algorithm is regarded as a more recent version of Quinlan's ID3, which was also created. To assess split points inside decision trees, it might employ information gain or gain ratios.

CART: Leo Bierman coined the acronym CART, which stands for "classification and regression trees." Gini impurity is often used by this method to choose the best characteristic to split on. Gini impurity quantifies the frequency of misclassification for a randomly selected attribute. A lower number is preferable when using Gini impurity to evaluate.

Random Forest Classifier - Popular supervised literacy algorithm Random Forest is part of the machine literacy movement. It may be applied to ML bracket and retrogression issues. It is based on the idea of ensemble literacy, which is the process of merging several classifiers to simplify complicated problems and improve model performance.

HGB classifier – Scikit-Learn v0.21.0 or a later interpretation will include HGB-Classifier. In plain English, we all understand that binning is a concept employed in data pre-processing, which entails taking into account VIT university and splitting the students based on the state in our nation, such as Tamil Nadu, Kerala, Karnataka, and so forth. The Decision Tree (DT) technique is also binned using the same principle once segmentation converted to numerical data. It will be used to speed up the algorithm by decreasing the amount of characteristics. The HGB classifier, which groups data using histograms, is used in DT to implement the same idea.

We generally have a number of parameters for fine-tuning our unique algorithms to get the best outcomes for all groups. The same is true for the HBG classifier; while there are many variables, some are crucial. These parameters include max iter, max depth, and l2 regularization, each of which serves a different function for perfecting the model. Learning rate addresses loss, Max iter addresses the quantity of repetitions necessary to obtain a satisfactory result, Max depth addresses multiple trees (Decision tree generalities), and L2 regularization addresses the regularisation idea to address overfitting issues.

Long Short-Term Memory or in short we call it as LSTM networks are expansion of intermittent neural networks (RNNs). which were primarily developed to address failure scenarios for RNNs. RNN is a network that analyses the input currently being received while simultaneously taking into account the past and momentarily keeping it in memory (short- term memory). The most well-known uses of its many operations may be found in the fields of non-Markovian control, speech processing, and musical composition. RNNs do have certain flaws, though. It can't initially store data for a lengthy period of time. It's often required to make future predictions using information that was preserved in the past. However, RNNs are completely incapable of managing the same "long-term dependences". Second, The balance between how much of the past should be "lost" and how much of the environment should be maintained cannot be more precisely managed. Another issue with RNNs is the exploding and vanishing slants (discussed below), which happen while a network is being trained by moving backwards. Long Short- Term Memory (LSTM) was introduced as a result. The training model has not been altered, and the vanishing grade problem has been almost entirely eliminated. LSTMs are used to bridge long time delays in some cases, and they can also deal with noise, dispersed representations, and nonstop values. Unlike the retired Markov model, LSTMs do not need keeping a limited number of nations from the past (HMM). LSTM parameters include learning rates, input and output impulses, and learning rates, among many more. Therefore, fine changes are not necessary. With LSTMs, the difficulty to modernise each weight is lowered to O(1), which is comparable to BPTT's (Back Propagation Through Time) complexity.

When in a prominent location, LSTM functions very much like an RNN cell. The internal functions of the LSTM network come next. Each of the three corridors that make up the LSTM, as seen in the figure below, has a distinct purpose.

An intermittent neural network that can handle long-term dependencies is known as an LSTM.

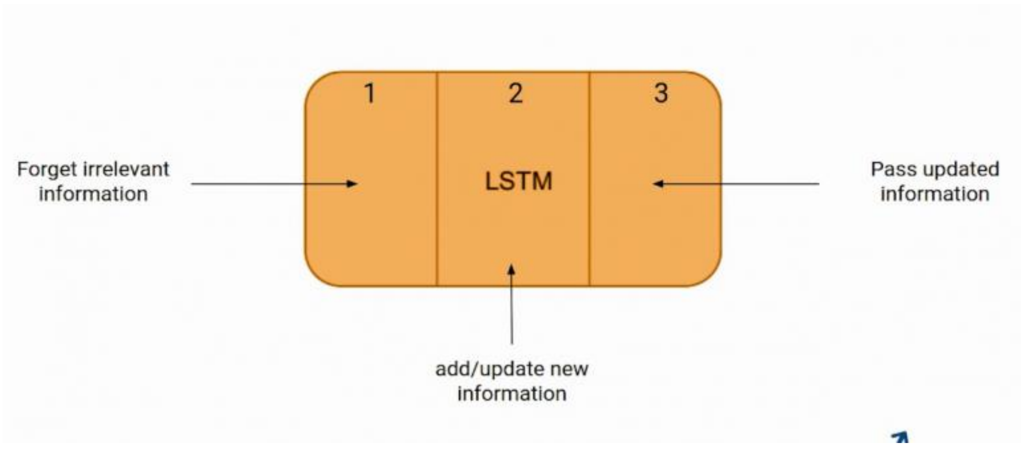


Fig 9: Simple LSTM Network

Recognize the structure and operation of an LSTM network.

Information can persist thanks to an improved RNN, often known as a long short term memory network. It can deal with the issue of RNN's grade disappearing. Patient memory is employed by an intermittent neural network, or RNN.

Imagine that you flash back to a previous scene while viewing a film or that you are aware of a prior chapter's events while reading a book. RNNs also function by flashing back the previous data and using it to recycle the present input. RNN's flaw is that they can't flash back long-term dependencies since grades are disappearing. LSTMs are purposefully made to prevent issues with long-term dependency.

LSTM Architecture

When at a high position, LSTM behaves very similarly to an RNN cell. The internal workings of the LSTM network are as follows. Each of the three corridors that make up the LSTM, as seen in the figure below, has a distinct purpose.

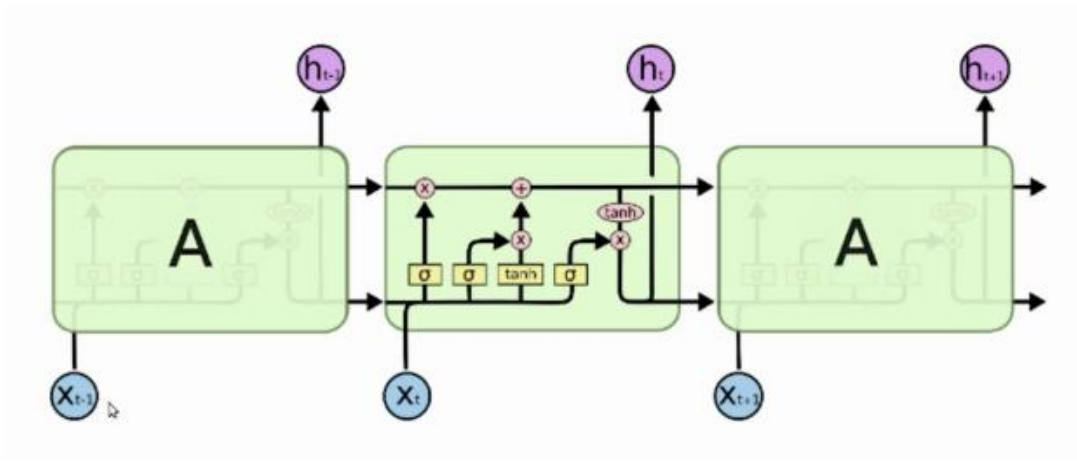


Fig 10: LSTM internal Structure

The first portion decides whether or not the information from the prior timestamp should be flashed back or ignored since it is unimportant. The input to this cell in the alternative section is used by the cell to try to learn new knowledge. In the third segment, the cell transfers the condensed data from the present timestamp to the future timestamp.

The three lanes that make up an LSTM cell are known as gates. The first, second, and third bones are referred as as Forget gate, Input gate, and Affair gate, respectively.

Similar to a standard RNN, an LSTM has a retired state where H(.t- 1) represents the retired state of the previous timestamp and H(t) represents the retired state of the present timestamp. A cell state that is separately represented by C(t- 1) and C(t) for past and present timestamps is another feature of LSTMs.

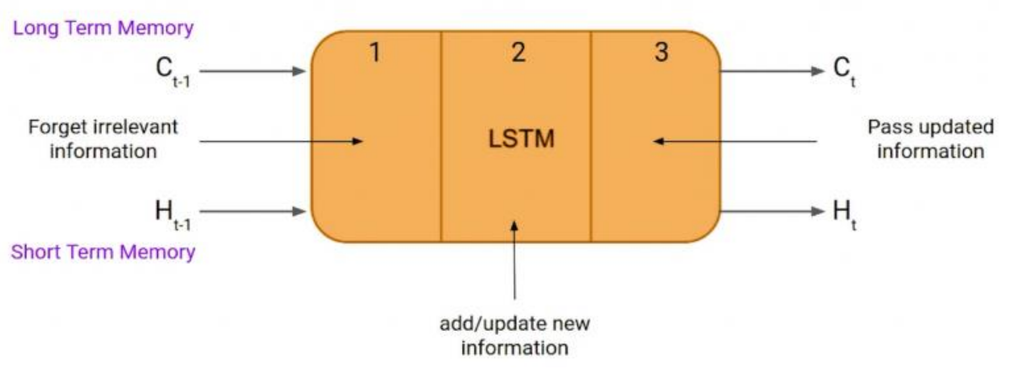


Fig 11: LSTM architecture

As a result, long term memory is referred to as the cell state and short term memory as the retired state.

Transfer Learning

There are two regions in the total procedure. Deep RNN training on a big musical is one aspect.Dataset (Magnatagatune dataset can be considered for example). The second element is genre.

Classification process after targeting the previously trained deep RNN and dataset (GTZAN [2]dataset is used in this paper). In particular, scattering transform is used the morning of each segment to slash the raw music data and to honour prior characteristics for the next neural network procedure training. labelled data is used to train a 5-subcaste RNN with GRU and Softmax classifier.

As the deep RNN we indicated, music clips. Finally, we apply the intended genre. GTZAN [2]classification dataset to fine-tune the RNN's training parameters. Below is a diagram of the proposed system's framework.

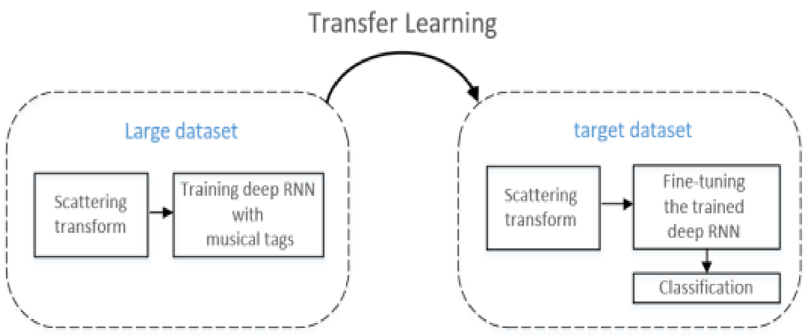


Fig 12: Transfer learning diagram

Here we used ResNet50[12] (Residual Neural Network 50) as our pretrained model. This is a custom classifier using GTZAN[2] dataset that has 10 classes of music genres[1], which are not included in original ImageNet dataset which is used to pretrain ResNet50 Model.

So, we are going to do 2 things:

1. Selecting the ResNet50 model from keras.application and
2. Retain the top (last) layer to recognize the classes from our dataset (GTZAN[2] dataset).

This method works well because the images were trained on a huge corpus of images and required the model to generate estimates on a variety of classes, which required the model to effectively learn how to remove features from images in order to solve the problem.

RESNET50[12]
For use cases of transfer learning, be sure to read the companion to transfer literacy & fine-tuning.
Keep in mind that every Keras application requires a certain form of input pre processing. Before giving your inputs to the model in ResNet50 , use “tf.keras.applications.resnet.preprocess” the follow-up input on them. The “resnet. preprocess” input function will null or zero-centre to each and convert the input photographs from RGB to BGR, and then compare the colour channel to the ImageNet collection without scaling.
Arguments
Include_top : whether to include the network top-most fully connected subcaste.
Input_tensor : Keras tensor, or the matter of layers. To provide the model with picture input, use Input ().
only given if include top is False; otherwise, the input shape must be one of the following: (224, 224, 3) (with "channels last" data format), or (3, 224, 224). (with "channels first" data format). It must have precisely 3 input channels, at least 32-bit breadth, and 32-bit height. One potential value is (200, 200, 3).
When include top is False, an optional point birth pooling mode is available.
None implies that the last convolutional block's 4D tensor product will be the model's output.
Avg indicates that the last convolutional block's yield will be subjected to global norm pooling, producing a 2D tensor as the model's final output.
Maximum denotes the application of global maximum pooling.

optional classes, number of classes that can be used to categorise photos, to be *determinedinclude_top* is True and there is Weights are not discussed.
classifier activation a callable or str. The "top" level activation function to employ
unless_include_top = 1 (true).
Setclassifier_activation = The "top" level logits cannot be returned. There are only two options for classifier activation when loading pretrained weights: None or "SoftMax".
Returns
An example instance of a Keras model.

InceptionV3[10] - Commencement v3(1)(2) is a convolutional neural network that was made available as a Google-net module to aid in the identification of objects and the analysis of images. This is the third version of the Google Convolutional Neural Network, which was first introduced during the ImageNet Recognition Challenge. Inceptionv3[11] features "under 25 million parameters," as compared to 60 million for AlexNet, and was created with the intention of allowing deeper networks while simultaneously keeping the number of parameters from rapidly rising. In the realm of computer vision, starting helps with item grouping(3), much as ImageNet may be used as a database of categorised visual objects. Many other processes have utilised the Inceptionv3 armature, which is usually "pre-trained" from ImageNet. In life lore, it helps in the study of leukaemia, which is one use that is comparable. The original term (commencement) was given this pseudonym when We must delve farther. is a popular internet joke that alludes to a line from Christopher Nolan's film Inception.

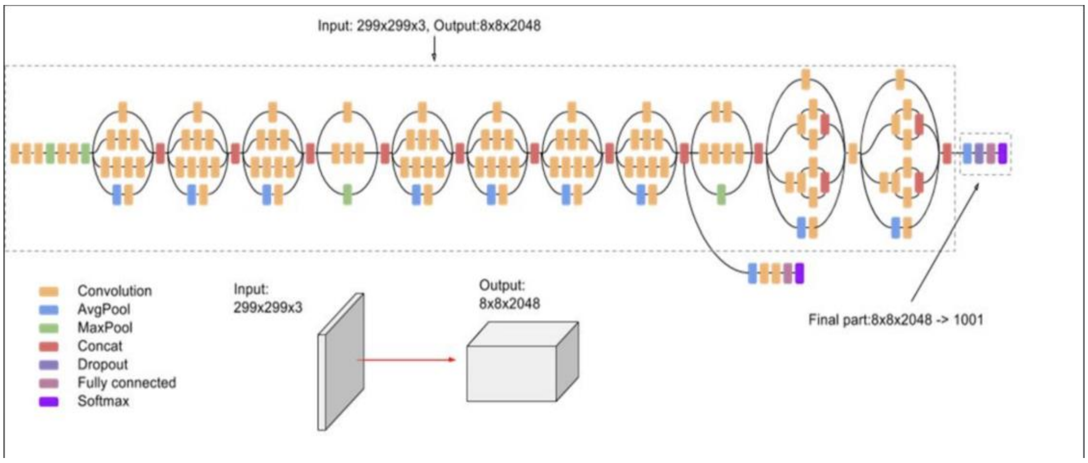


Fig. 13: Inception V3 developed by google.

Input Pipeline of Inception V3

The hosts get data from the memory or train system, do any necessary data preparation, and then send this data to TPU cores with preprocessed data. There are fixed 3-stages of the host's data processing and handling as a whole and put them as

1) Storage
2) Preprocessing,
3) Transfer

The following figure depicts the visual in high-definition.

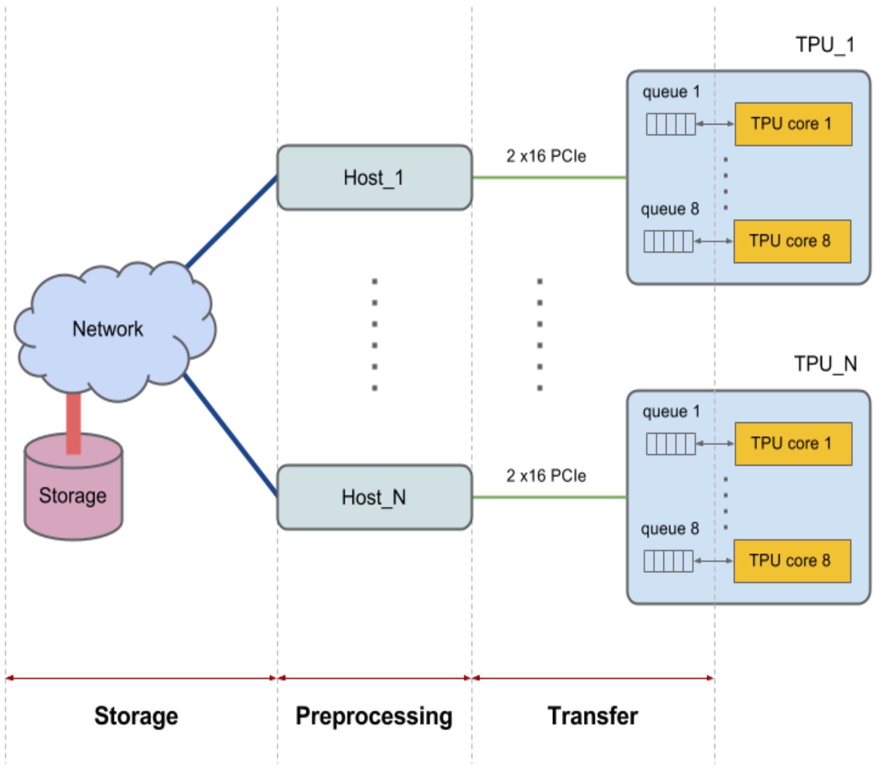
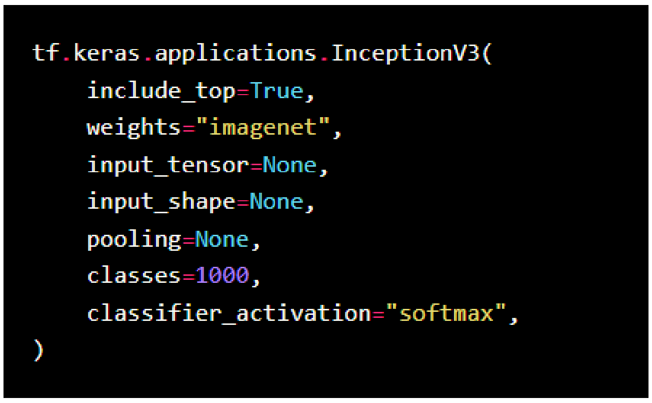


Fig 14: Inception V3 Implementation

Every Keras application anticipates a unique type of input preprocessing. Before passing your inputs to the model for InceptionV3, calltf.keras.applications.inception v3.preprocess input on them. This function measures input pixels between -1 and 1.



```
tf.keras.applications.InceptionV3(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    classifier_activation="softmax",
)
```

Fig 15: Inception V3 model Function

Arguments for Incpetion V3

Include top: Boolean, indicating whether to include the topmost level of the network that is fully linked as the final subcaste. neglect for True.

weights : A path to the weights training to be loaded, image-net (pre-training on ImageNet), or None (arbitrary initialization). disregard for image-net

input_tensor : Optional Keras tensor, also known as a layer affair. To provide the model with picture input, use Input(). For participating inputs amongst several distinct networks, input tensor is helpful, a failure to None.

Input_shape : Input shapes must be either (3, 299, 299) or (299, 299, 3) (with channels last data format) unless include top is False (with channels first data format). It should have exactly three input channels and a maximum of 75 degrees in both height and range. A possible value might be (150, 150, 3). If the input tensor is given, the input shape will be disregarded.

pooling : Point descent with an optional pooling mode when include top is False.

None (dereliction) suggests that the problem with the model is a 4D tensor problem in the last convolutional block.

The model's focus will be a 2D tensor since global normal pooling will be applied to the last convolutional block, according to the average.

Maximum denotes the application of global maximum pooling.

classes : An optional number of classes to group photographs into may be specified if include top is True and no weights argument is given. 1000 days of disregard as callable or str. The "top" level activation function to employ Unless include top = True, ignored. Returns the logical of the "top" level, set classifier activation to None. There are only two options for classifier activation when loading pretrained weights: None or "softmax". Returns a keras.model case.

BiLSTM

Two hidden layers with opposing directions are connected to the same output via bidirectional recurrent neural networks (BRNN). This kind of generative deep learning allows the output layer to simultaneously receive data from previous (backwards) and future (ahead) states. The purpose of BRNNs, which were developed in 1997 by Schuster and Paliwal, was to expand the network's access to input data. The flexibility of the input data is constrained by the need for fixed input data in multilayer perceptron (MLP) and temporal delay neural network (TDNN) models, for instance. Because the future input information cannot be accessed from the current state, standard recurrent neural networks (RNNs) are likewise constrained. BRNNs, however, do not need their input data to be fixed. They can also access their upcoming input data.

When the input context is required, BRNN are very helpful. Knowing the letters that come before and after the current letter, for instance, can improve performance when recognising handwriting.

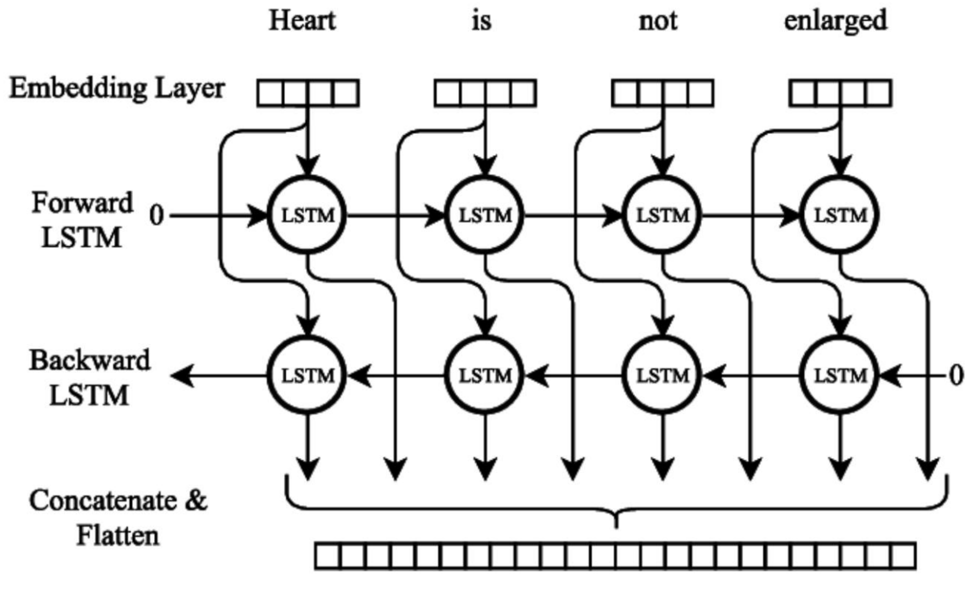


Fig 16: Structure of BiLSTM Model

Architecture of BiLSTM

According to the BRNN concept, a normal RNN's neurons are divided into two directions: one for positive time direction (ahead states) and another for negative time direction (backward states). The outputs of such two states are not coupled with their corresponding counterpart states' inputs. The right diagram may be used to show the overall structure of RNN and BRNN. Unlike normal RNN, which needs delays for future information inclusion, two-time directions allow input data from the past and future of the present time frame to be utilised.

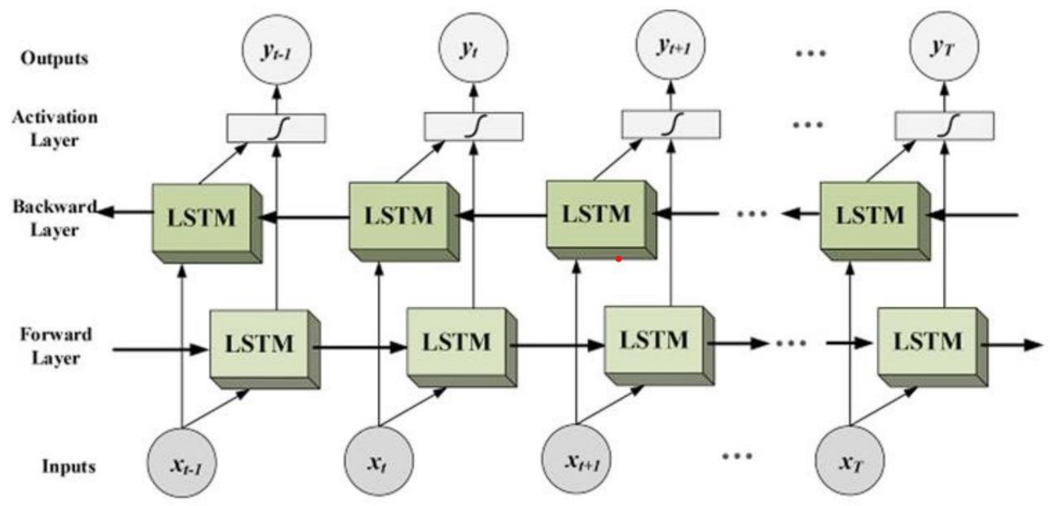


Fig 17: Internal structure of BiLSTM

BiLSTM + Attention Model

Attention models, often referred to as attention mechanisms, are input processing methods for neural networks that enable the network to concentrate on components of a complicated input, one at a time, until the entire dataset is categorized. Complex activities must be divided into smaller attentional regions that are handled sequentially. Like how the human mind approaches a new problem by breaking it down into smaller jobs and tackling each one at a time. For attention models to work, there must be ongoing reinforcement or back population training.

A function that transfers a query and "s set" of key value pairs to an output is how attention is generally described. a situation where the query, keys, values, and output are all vectors. Following that, the result is produced as a weighted sum of the values, with the weights allocated to each value being indicated by how well the query matches the associated key value.

In actual application, attention enables neural networks to simulate the human visual attention process. Like humans processing a novel picture, the model concentrates on a specific part of an image with intense "high resolution" concentration while seeing the surrounding portions with "low resolution," and it changes the focusing point as the network learns more about the scene.

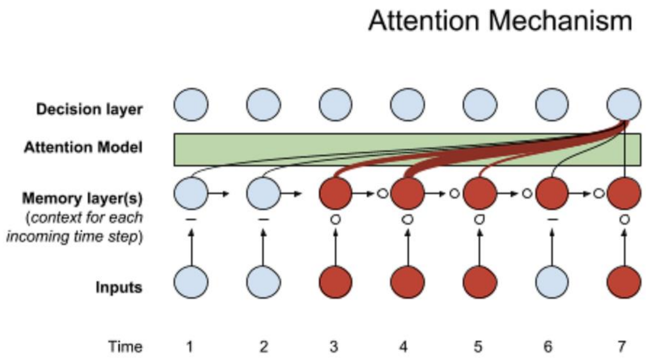


Fig 18: Attention Model

3.1.4 System Design

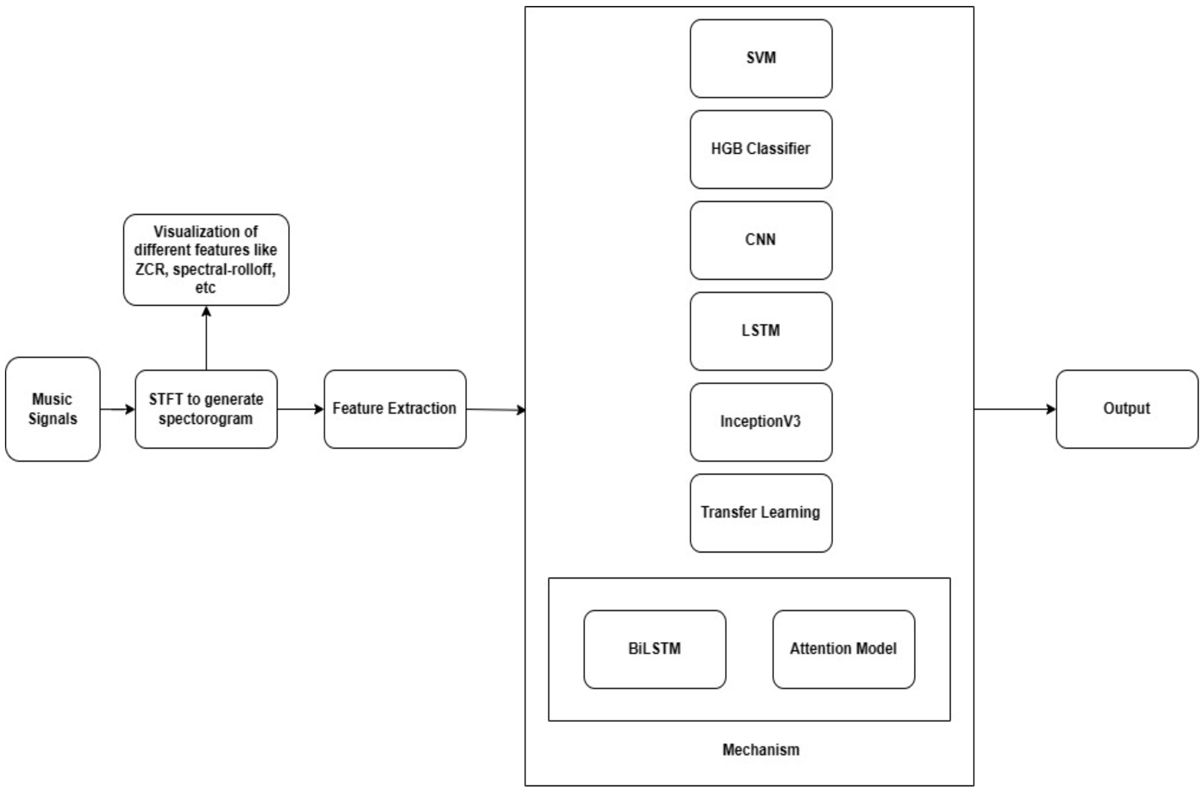


Fig 19: Dataflow diagram

The proposed data flow diagram works in the following way.

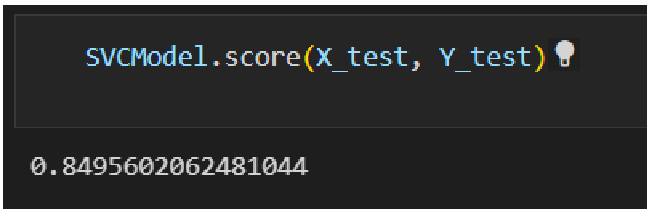
1. GTZAN dataset us used as the primary dataset.

2. In pre-processing dataset was increased to 8000, two-second clips by sampling a continuous 2-second window. Raw audio files are converted to Mel-spectrograms by applying STFT using librosa package.
3. Features of music like ZCR, Spectral-centroid, MFCC are visualized using librosa library.
4. Dataset is divided in training and testing with test size of 0.2.
5. Logistic regression and SGD classifier are used to evaluate the basic performance of the model.
6. SVM classifier is used as they are best edge classifier, and the employ kernels to uncover intricate patterns in the input data as the there is no linear separability.
7. CNN is used as 3 fully connected Relu activation function and SoftMax function in output layer.
8. LSTM model is used as it is better than RNN model.
9. Inception V3 developed by google-net which uses image-net dataset which has 3 parts input, pre-processing, and transfer is used.
10. Transfer Learning with ResNet50 as pretrained model and CNN is used to retrain the last layer to recognize the classes.
11. BiLSTM with Attention model is the last model for the project.
12. Comparison of performances of the models are shown through a table as well as graphs.

CHAPTER - 4

PERFORMANCE ANALYSIS

SVM:



```
SVCModel.score(X_test, Y_test)

0.8495602062481044
```

Fig 20: SVM testing R^2 score.

Histogram Gradient Boosting Algorithm:

It is a gradient Boosting Classification Tree that is based on histograms.

For large datasets with n samples more than 10,000 data points, this estimator is more effective and quicker than Gradient Boosting Classifier.

Native support for missing values is provided by this estimator (NaNs). At each split point during training, the tree farmer learns, Which child should get samples with missing values depends on the implicit benefit. When constructing predictions, samples with missing values are appropriately assigned to the left or right child. Samples with missing values are also counterplotted to the child with the most samples if no missing values for a particular attribute were discovered during training.

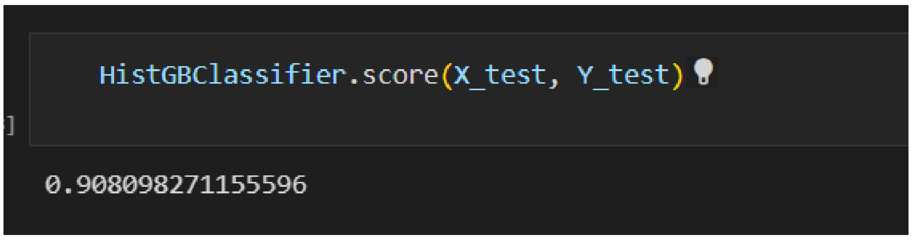
This fulfillment is inspired by LightGBM.

Parameter for Histogram Gradient Boosting

In general, we have a number of factors for fine-tuning our unique algorithms to get the best outcomes for all groups. The same is true for the HBG classifier; even though there are many variables, certain are essential. These variables for the HBG classifier include:

1. *max_iter,*
2. *max_depth,*
3. *l2_regularization,*

Each has a specific goal for fine-tuning the model. For example, max iter deals with the number of repetitions necessary to get a good result, max depth deals with multiple trees (Decision tree generalities), and l2 regularization deals with the regularisation concept to help with overfitting issues. They all deal with some aspect of loss or repetition requirements.



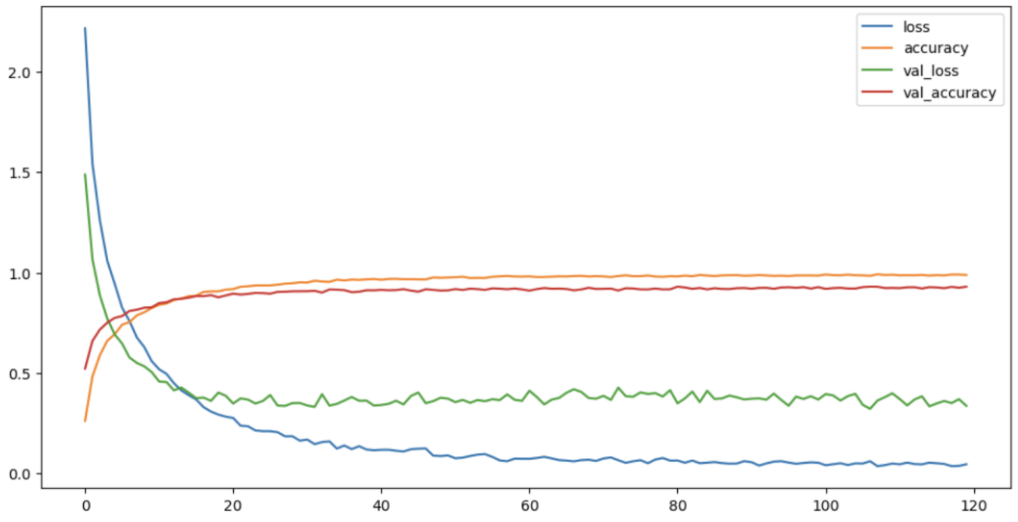
```
HistGBClassifier.score(X_test, Y_test)
```

```
0.908098271155596
```

Fig 21: Histogram Gradient Boosting R^2 score

CNN

CNN accuracy and validation accuracy and loss plot over 100 Epochs



Graph 1: Train and Test accuracy of CNN Model

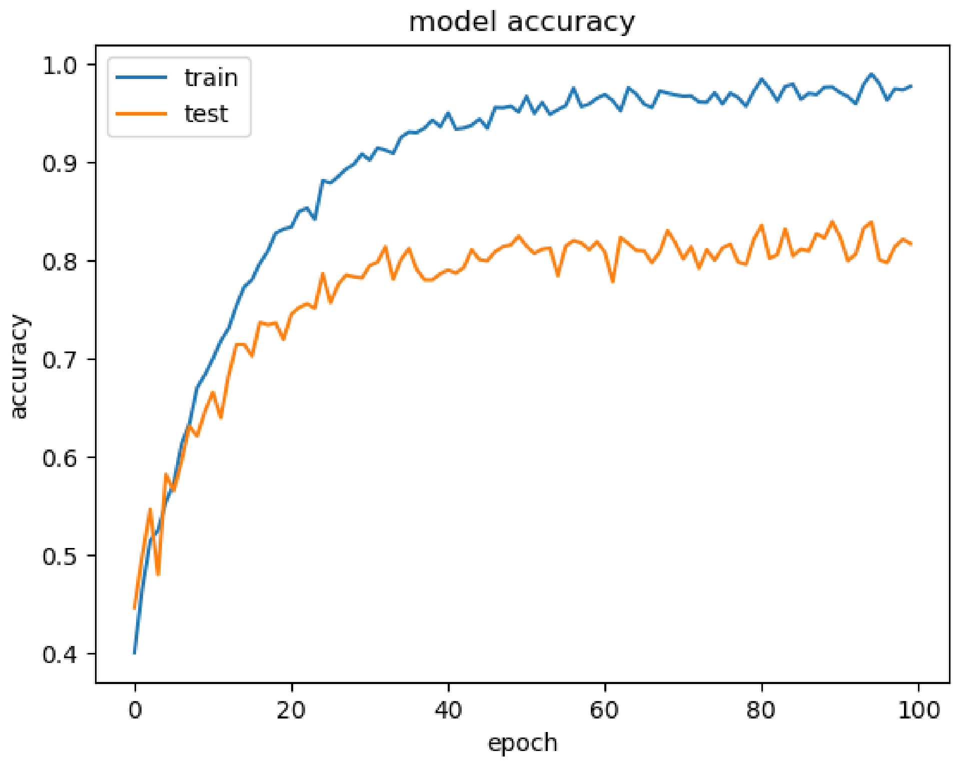
As we can infer from the graph and the model there are 5 layers, 4 of them have ReLu activation function and last layer has the SoftMax function. Metric for evaluation of the result is accuracy.

LSTM

Information can persist thanks to an improved RNN, often known as a long short term memory network. It can deal with the issue of RNN's grade disappearing. Patient memory is employed by an intermittent neural network, or RNN.

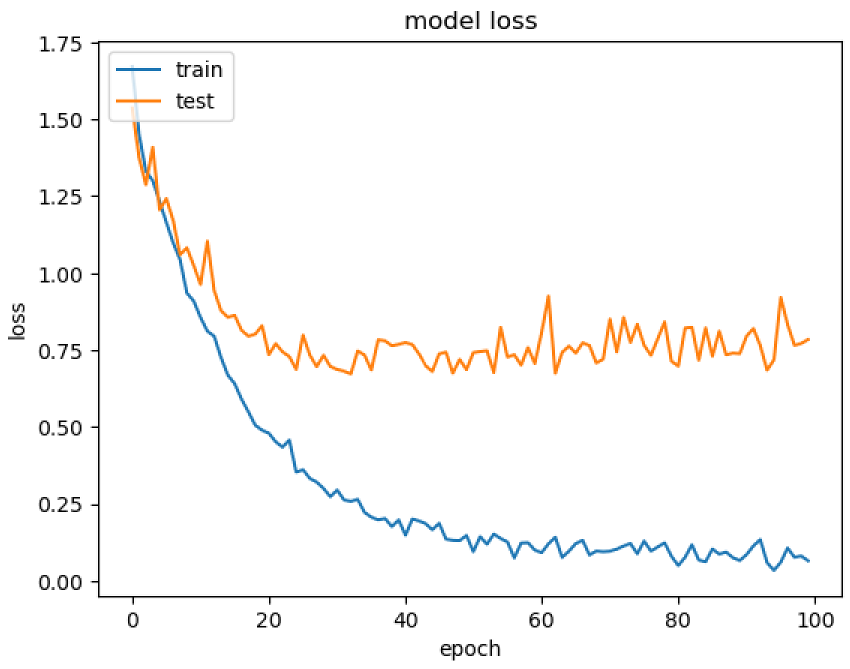
Additionally, RNNs function by flashing back the earlier data and using it to recycle the present input. RNN's flaw is that they can't flash back long-term dependencies since grades are disappearing. LSTMs are purposefully made to prevent issues with long-term dependency.

LSTM accuracy and validation accuracy plot over 100 Epochs



Graph 2: Train and Test accuracy of LSTM Model

As we can infer from the graph and the model there are 6 layers, 3 of them have LSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is accuracy.



Graph 3: Train and Test model loss for LSTM

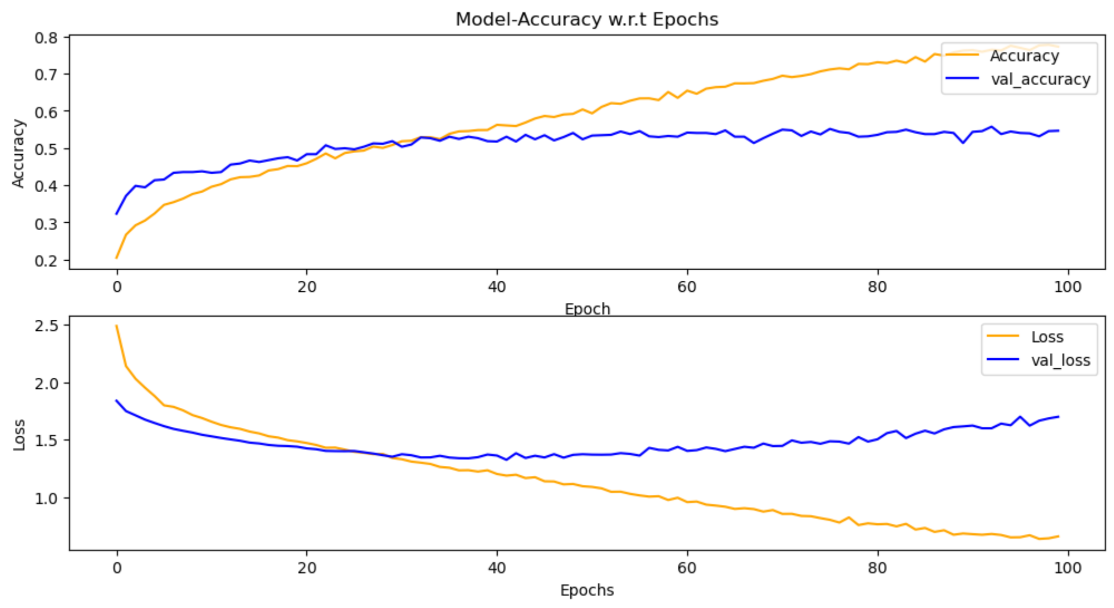
As we can infer from the graph and the model there are 6 layers, 3 of them have LSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is loss.

Inception V3:

It has been established that the Inception v3 image recognition model performs with less accuracy than 78.1 percent on the ImageNet dataset. The model is the result of several ideas that have been developed through time by numerous experimenters. It is based on Szegedy, et al original.'s article, "Redefining the Inception Architecture for Computer Vision."

$$g_{k+1}^{-2} = \alpha g_k^{-2} + (1-\alpha) g_k^2$$

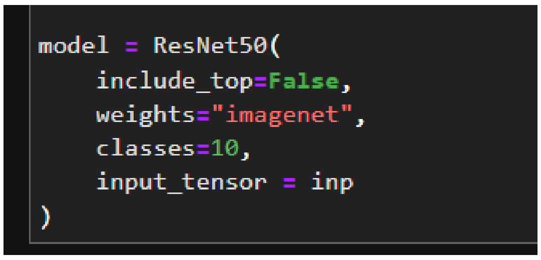
$$\omega_{k+1} = \beta \omega_k + \frac{\eta}{\sqrt{g_{k+1}^{-2}}} \nabla f(\omega_k)$$



Graph 4: Inception V3 Model accuracy and loss w.r.t epochs

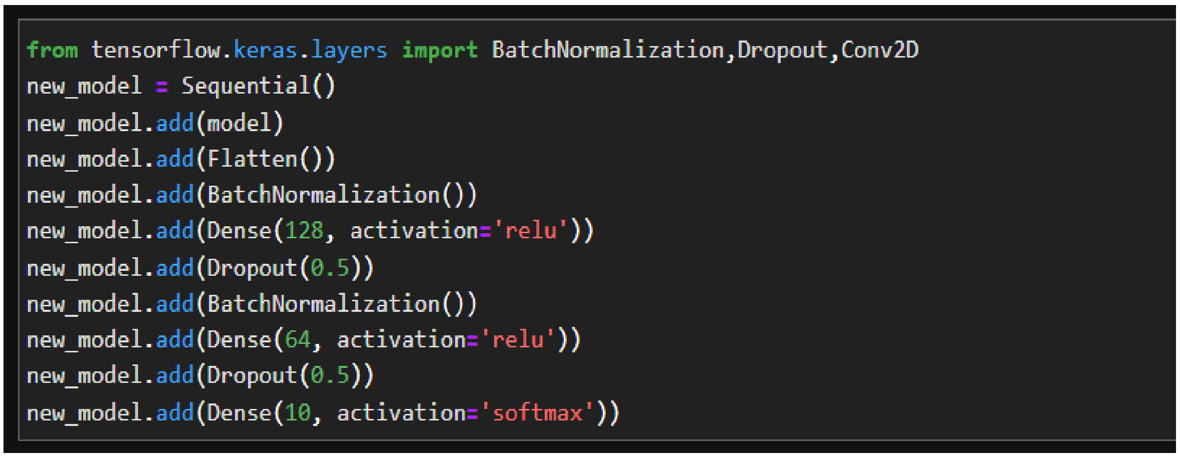
As we can infer from the graph and the model there are 4 layers, 3 of them have ReLu activation function and last layer has the SoftMax function. Metric for evaluation of the result is accuracy and optimizer used is SGD.

Transfer Learning:



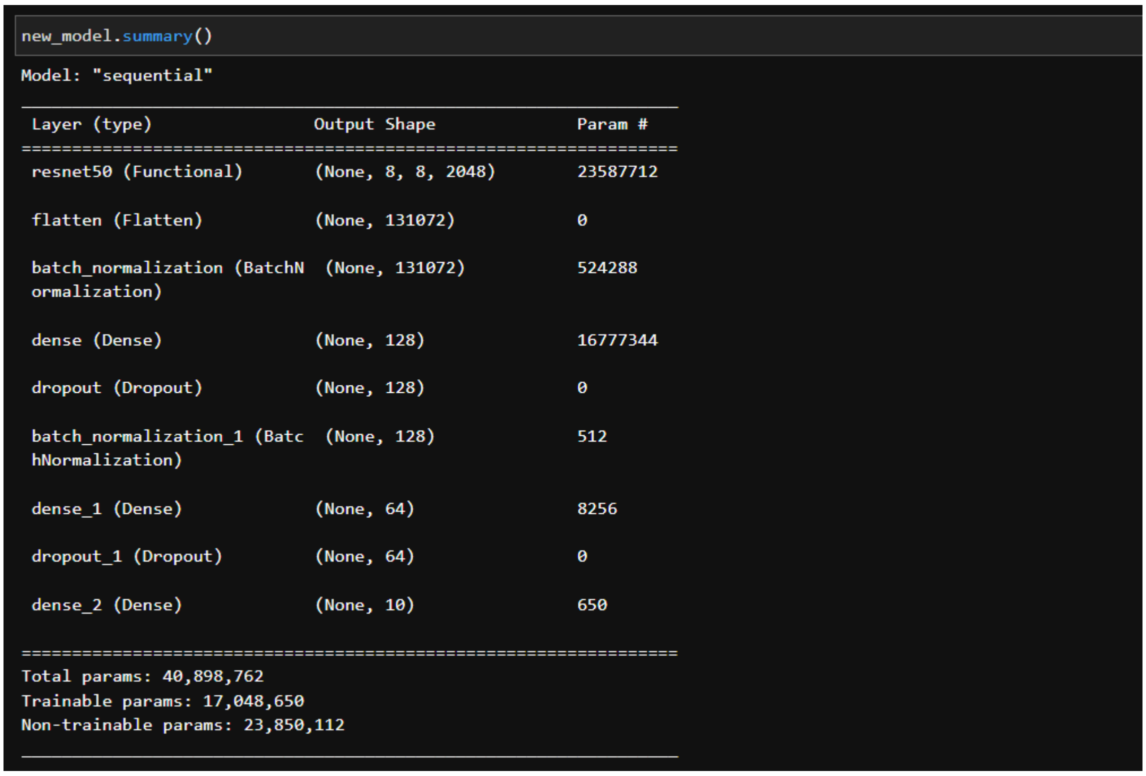
```
model = ResNet50(
    include_top=False,
    weights="imagenet",
    classes=10,
    input_tensor = inp
)
```

Fig 22: ResNet50 Model (pretrained model import)



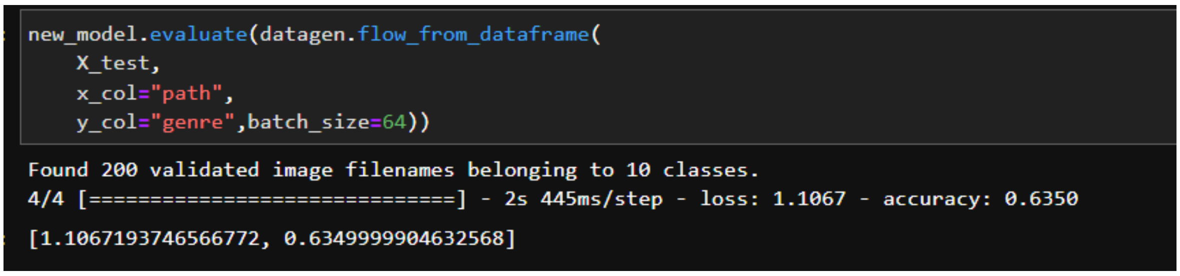
```
from tensorflow.keras.layers import BatchNormalization,Dropout,Conv2D
new_model = Sequential()
new_model.add(model)
new_model.add(Flatten())
new_model.add(BatchNormalization())
new_model.add(Dense(128, activation='relu'))
new_model.add(Dropout(0.5))
new_model.add(BatchNormalization())
new_model.add(Dense(64, activation='relu'))
new_model.add(Dropout(0.5))
new_model.add(Dense(10, activation='softmax'))
```

Fig 23: A new model having first layer as ResNet50 Model



```
new_model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 8, 8, 2048)        23587712

 flatten (Flatten)           (None, 131072)            0

 batch_normalization (BatchN  (None, 131072)           524288
 ormalization)

 dense (Dense)               (None, 128)               16777344

 dropout (Dropout)           (None, 128)               0

 batch_normalization_1 (Batc  (None, 128)              512
 hNormalization)

 dense_1 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 10)                650

=================================================================
Total params: 40,898,762
Trainable params: 17,048,650
Non-trainable params: 23,850,112
_________________________________________________________________
```

Fig 24: Transfer Learning model Summary Statistics



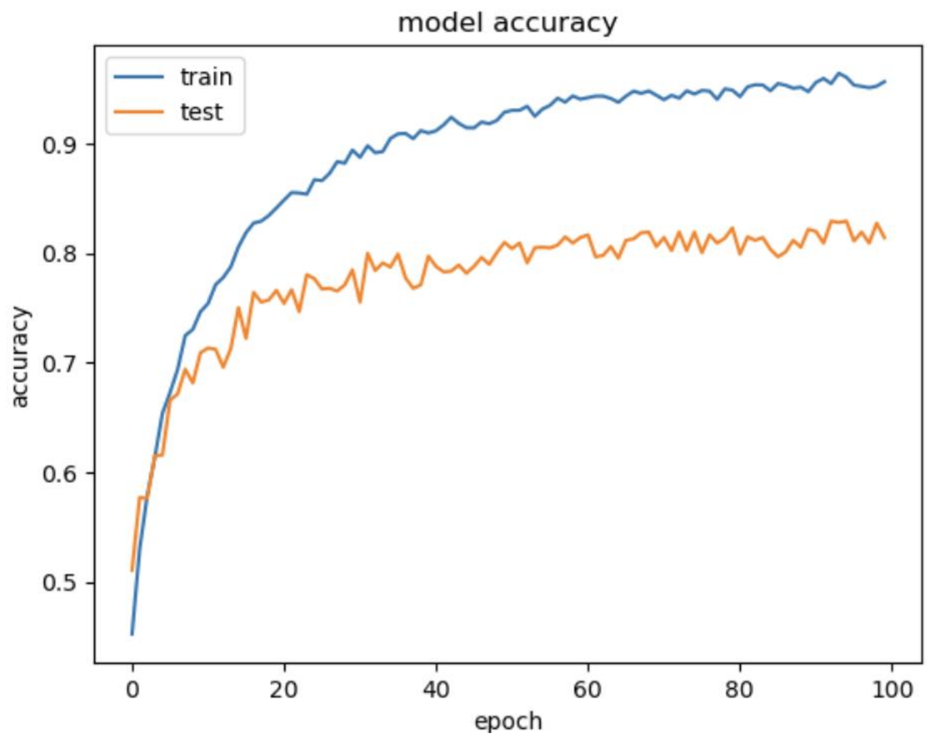
```
new_model.evaluate(datagen.flow_from_dataframe(
    X_test,
    x_col="path",
    y_col="genre",batch_size=64))

Found 200 validated image filenames belonging to 10 classes.
4/4 [==============================] - 2s 445ms/step - loss: 1.1067 - accuracy: 0.6350
[1.1067193746566772, 0.6349999904632568]
```

Fig 25: Transfer learning testing loss and accuracy measures

Transfer Learning with ResNet50 as pretrained model and CNN is used to retrain the last layer to recognize the classes. Performance metric used is accuracy.

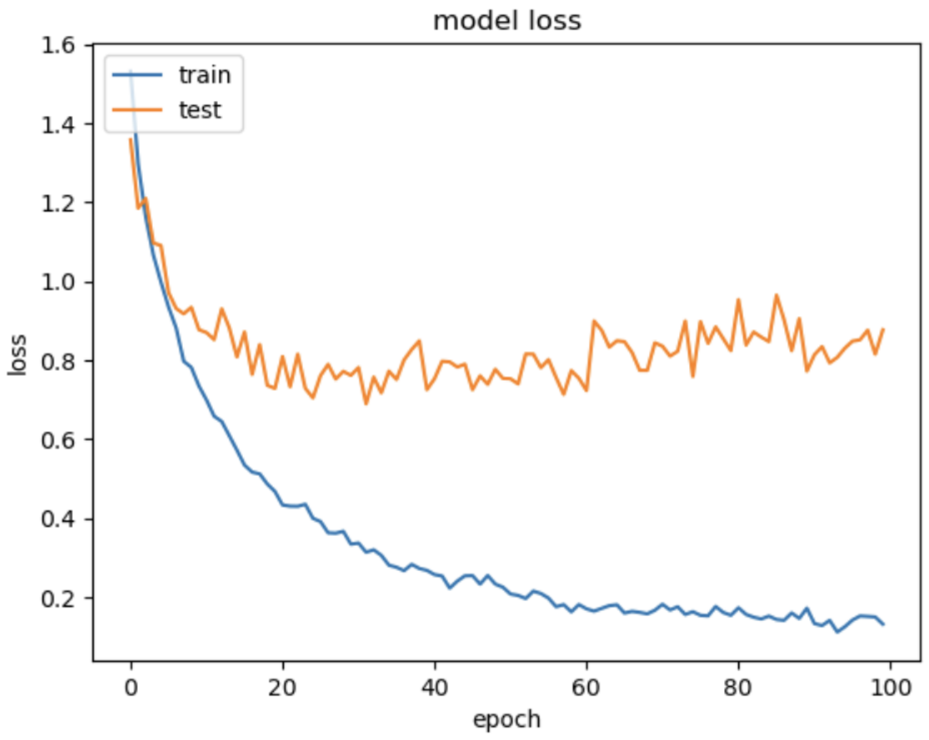
BiLSTM [13]

BiLSTM training accuracy and validation accuracy plot over 100 Epochs



Graph 5: Training and testing accuracy of BiLSTM model

As we can infer from the graph and the model there are 6 layers, 3 of them have BiLSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is accuracy.

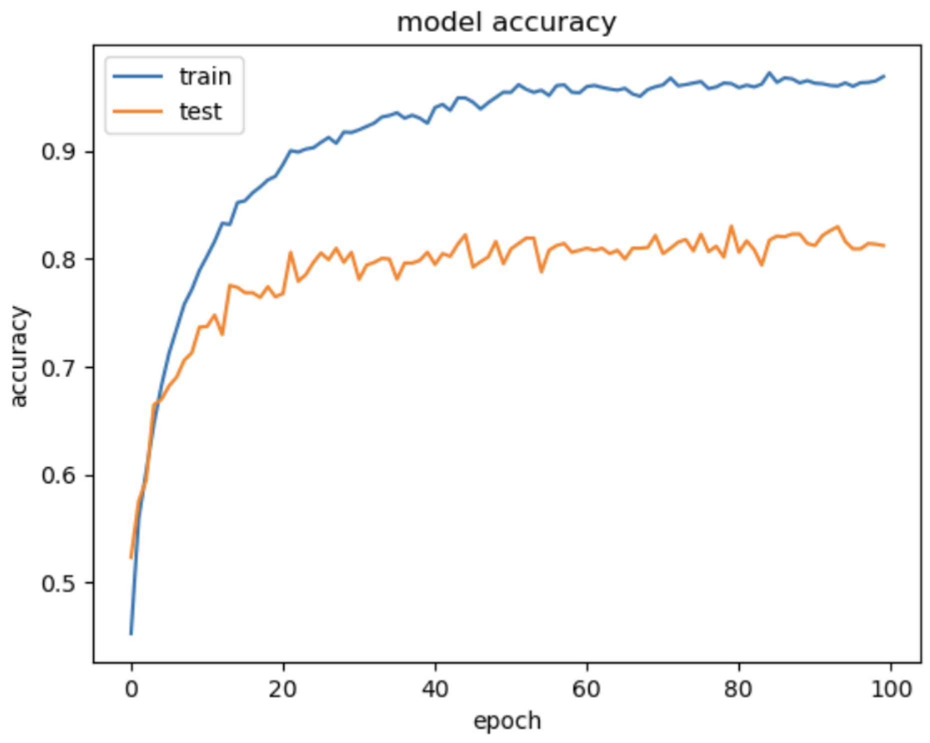


Graph 6: Training and Testing loss of BiLSTM model

As we can infer from the graph and the model there are 6 layers, 3 of them have LSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is loss.

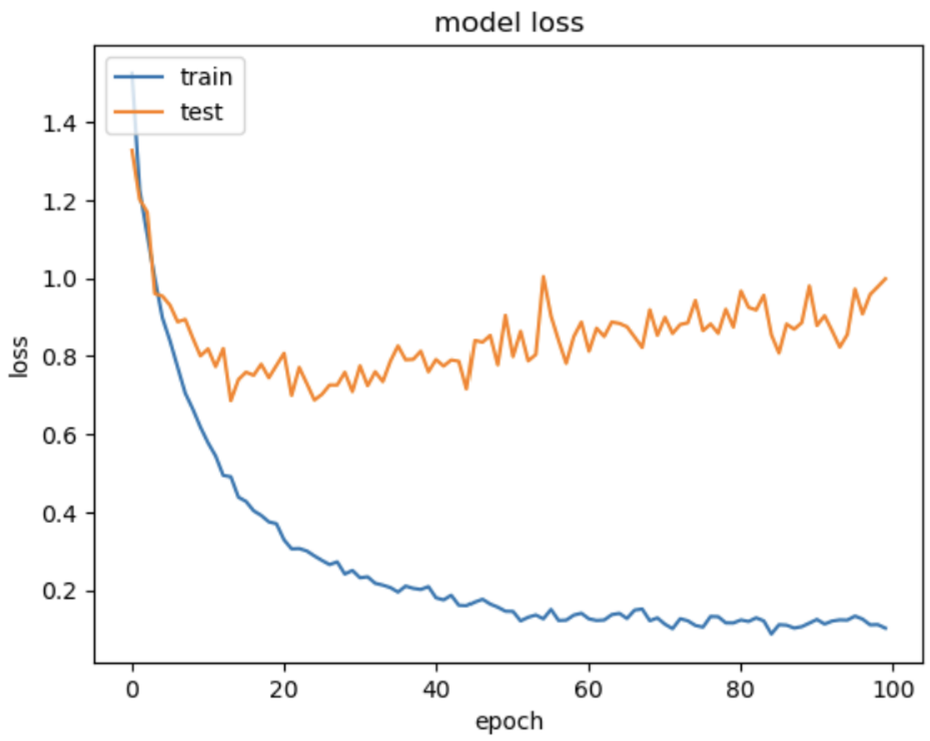
Attention Model + BiLSTM [13]

BiLSTM + Attention model training accuracy and validation accuracy plot over 100 Epochs



Graph 7: Training and testing accuracy of BiLSTM + Attention model

As we can infer from the graph and the model there are 8 layers, 1 layer is attention layer, 3 of them have BiLSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is accuracy.



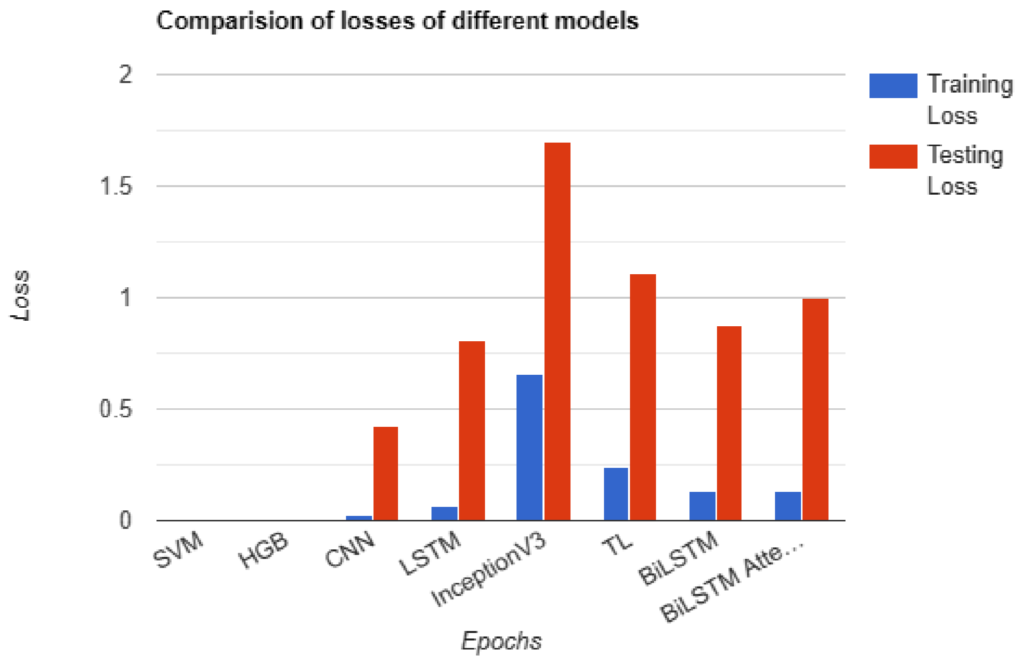
Graph 8: Training and testing loss of BiLSTM + Attention model

As we can infer from the graph and the model there are 8 layers, 1 layer is attention layer, 3 of them have BiLSTM layer and 2 intermediate layers are normalization layers and last layer has the SoftMax function. Metric for evaluation of the result is loss.

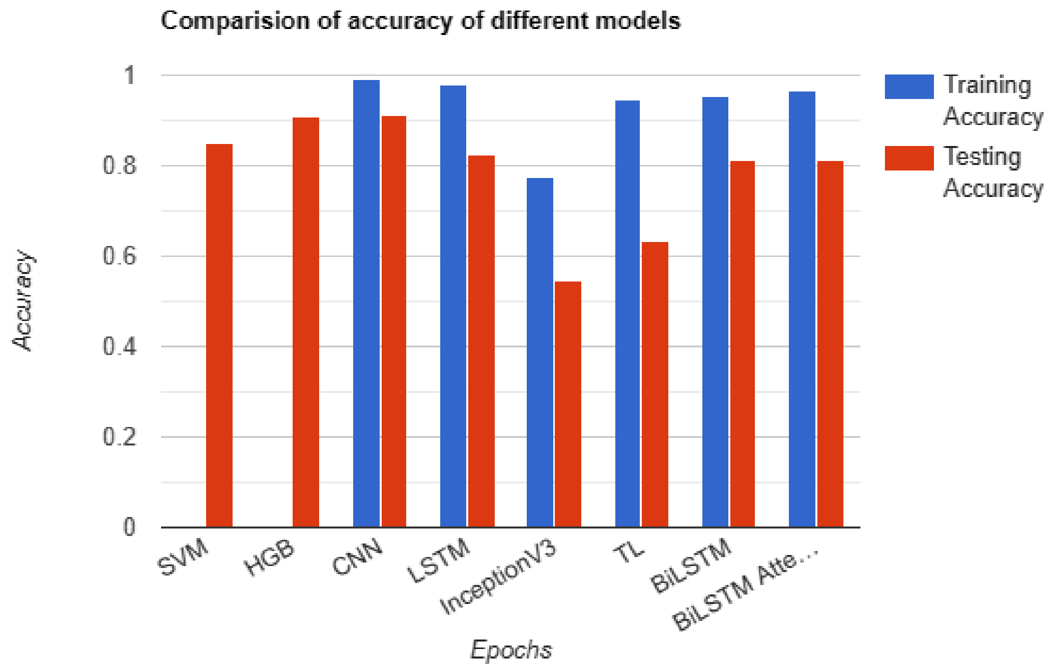
Comparison table for different algorithms and models used in this project standard comparison of 100 epochs.

Table 2: Comparison Table for Different Algorithms and models

Sr. No	Model / Algorithm Name	Training Info		Testing Info	
		Loss	Accuracy	Loss	Accuracy
1	SVM	-	-	-	0.849
2	HGB Classifier	-	-	-	0.908
3	CNN (200 epochs)	0.0254	0.993	0.429	0.914
4	LSTM[9]	0.0658	0.9775	0.807	0.826
5	Inception V3[11]	0.658	0.777	1.698	0.546
6	Transfer Learning (ResNet50[12]+CNN[13]) (23 epoch only)	0.243	0.944	1.106	0.634
7.	BiLSTM	0.1318	0.954	0.876	0.814
8.	BiLSTM + Attention Model	0.102	0.965	0.998	0.812



Graph 9: Comparison of losses of different models



Graph 10: Comparison of accuracy of different models

CHAPTER - 5
CONCLUSIONS

5.1 Conclusions

As we have tried 7 different algorithms in our project:

1. Support Vector Machines (SVM Multiclass Classifier)
2. Histogram based Gradient Boosting (HGB Classifier)
3. Convolution Neural Networks (CNN)
4. Long Short-Term Memory Neural Networks (LSTM)[9]
5. Inception V3 (Google Inception Project)[10]
6. Transfer Learning (ResNet50 and CNN) [12], [13]
7. BiLSTM with Attention model [14]

We got the highest accuracy figures with CNN after 200 epochs of training the model we get 91.4 % testing accuracy.

While other models performs better than neural networks for standard of 100 epochs training, e.g., SVM is 84.9%, HGB classifier 90.8% accuracy score, LSTM with 82.6% accuracy BiLSTM with 81.4% accuracy, BiLSTM with 81.2% accuracy.

1. The frequency and breadth variations that occur quickly can be used to distinguish and categorise the audio into separate characteristics.
2. We can easily create a wave plot using librosa to visualise the audio frequency rise in terms of frequency and width about time.
3. A total of 39 frequentness and amplitude-related characteristics are provided by MFCCs. In that the amplitude of frequencies is connected to 12 parameters. Because MFCCs give us sufficient frequency channels to dissect audio, they are frequently employed as the point of origin in audio files.
4. The primary functions of MFCC are to eliminate vocal excitation (pitch information) by framing audio, make uprooted characteristics independent, support human-recommended loudness and frequency ranges for sound, and record the environment.

GTZAN[2] and other datasets which is used for Music genre classification they have numerous features used to differentiate music genres, but a standard features MFCC is used a lot and it also drives out most of the feature using MFCC. We can conclude that Simple CNN and SVM and HGB classifier are more accurate in classifying MFCC better than other learnings algorithms.

5.2 Future Scope

The extension of this would work on real songs and music in various formats (mp3, au etc). The styles that each genre reflects will also continue to change throughout time. Therefore, the future intention is to keep up with changes in genre-specific styles and to expand our programme to support these updated styles. This work may be developed to serve as a music suggestion system depending on the user's mood.

5.3 Applications Contributions

1. Classifying music by genre is a fundamental step in creating a powerful recommendation system.
2. Automating music classification aims to speed up and reduce the time required for song selection.
3. Beyond this line, variety promotes higher learning, which enhances the manifestation of culture.

REFERENCES

[1] J. Hayes, “Music Genres | Defining Different Types Of Music Genres,” *www.openmicuk.co.uk*. https://www.openmicuk.co.uk/advice/types-genres-of-music/ (accessed Nov. 17, 2022).

[2] A. OLTEANU, “GTZAN Dataset - Music Genre Classification | Kaggle,” *Kaggle.com*. https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification (accessed Nov. 17, 2022).

[3] J. Prashad, “prasad213/music-genre-classification - Jovian,” *jovian.ai*. https://jovian.ai/prasad213/music-genre-classification (accessed Nov. 17, 2022).

[4] Tao Li and M. Ogihara, “Music Genre Classification with Taxonomy,” in *Proceedings. (ICASSP ’05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, pp. 197–200. doi: 10.1109/ICASSP.2005.1416274.

[5] T. Li, M. Ogihara, and Q. Li, “A comparative study on content-based music genre classification,” in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’03*, 2003, p. 282. doi: 10.1145/860435.860487.

[6] Andresmanzalini, “andresmanzalini/ClasificacionGeneroMusical_DL: Clasificacion de género musical con Deep Learning.” https://github.com/andresmanzalini/ClasificacionGeneroMusical_DL (accessed Nov. 17, 2022).

[7] L. Feng, S. Liu, and J. Yao, “Music Genre Classification with Paralleling Recurrent Convolutional Neural Network,” Dec. 2017, doi: 10.48550/arxiv.1712.08370.

[8] S. Doshi, “Music Feature Extraction in Python | by Sanket Doshi | Towards Data Science,” *towardsdatascience.com, medium.com*. https://towardsdatascience.com/extract-features-of-music-75a3f9bc265d (accessed Nov. 17, 2022).

[9] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, “LSTM: A Search Space Odyssey,” *IEEE Trans Neural Netw Learn Syst*, vol. 28, no. 10, pp. 2222–2232, Oct. 2015, doi: 10.1109/TNNLS.2016.2582924.

[10] “InceptionV3.” https://keras.io/api/applications/inceptionv3/ (accessed Nov. 17, 2022).

[11] “Advanced Guide to Inception v3 | Cloud TPU | Google Cloud.” https://cloud.google.com/tpu/docs/inception-v3-advanced (accessed Nov. 17, 2022).

[12] “ResNet and ResNetV2.” https://keras.io/api/applications/resnet/#resnet50-function (accessed Nov. 17, 2022).

[13] A. Chowdhry, “Music Genre Classification Using CNN,” *clairvoyant.ai*. https://www.clairvoyant.ai/blog/music-genre-classification-using-cnn (accessed Nov. 17, 2022).