MGC: A Review on Music GENRE Classification Using Deep Learning Models

Authors: Rigved Alankar[1*], Gurmehar Singh Soni[2], Abhay Chaudhary[2]

Affiliations:

1* University School of Information, Communication and Technology, Guru Gobind Singh Indraprastha University, India.

2 School of Computer Science and Engineering, Vellore Institute of Technology, AP, India.

Addresses:

1* Shivalik Boys Hostel, Guru Gobind Singh Indraprastha University, Beside Odisha Bhawan, Dwarka Sector 16-C,Delhi,160003

2 VIT-AP University,G-30,Inavolu,Beside AP Secretariat Amravati, Andhra Pradesh 522237

Abstract: Music genre refers to a class of music based on the tones or sounds used in it and the classification of music based on its genre is called music genre classification. It is a crucial part of the music information retrieval process. With the advent of big data, machine learning technics are being widely used to extract logical inferences from large data sets. The paper describes the uses of recurrent neural networks, notably GRU and LSTM. Optimisation algorithms like Adam, SGD, RMSProp and Rectified Adam and then comparing their accuracies. We propose our model to revolutionise the way for a data set being handled in a data lake.

One Sentence Summary: A schematic procedure for music genre classification using deep learning models with metadata.

Introduction:

Music is a classical class which identifies certain pieces of music as part of a popular tradition or series of conventions, referring to the tones or sounds used, which occurs in single (melody) or multiple (harmony) lines and which can be sounded by or sounded by one or many voices or instruments. Music may be separated into multiple genres, such as popular music and art music or religious music and secular music, in various forms. Recovery of musical knowledge (MIR) is the interdisciplinary study of music retrieval. The classification of music is the method by which a piece of music is categorised into a specific category. In the field of information processing, it plays a vital function.

Background:

A neural network refers to a variety of algorithms used to derive logical results from a given dataset. Neural networks remain encouraged by the biological neurons in a human body responsible for communicating information to other areas of the body. Neural networks are a way to teach computers, in which a Cpu can evaluate the training examples to carry out such tasks. Neural networks are currently used to address several market challenges, including revenue analysis, consumer study, data validation and risk control. New networks are also used for estimation of time series, data analysis of anomalies and interpretation of natural languages. A neural network contains thousands or even millions of strongly interconnected superficial processing nodes. Today, most of the neural networks are structured into node layers, which ensures that data can travel in one direction. A single node may be related to

several nodes of the lower layer from which it collects data and several nodes of the above layer to which it transmits information.

A node will designate a number known as a 'weight' to each of its inbound links. Whenever the network is activated, each of the links will obtain a new data object, another number multiplied by the corresponding weight. The resultant items are then combined, and a single number is given. The node does not transfer data to the next layer if the number is below the threshold value, the node fires, that generally means sending the number several weighted inputs along with all its outgoing connexions in today's neural networks.

As a neural net is equipped, it is initially set to random values in both weights and thresholds. The training data is sent to the lower layer — the input layer — and the layers move thru, multiply and connect complexly together, before they eventually enter the output layer, dramatically transformed. The weights and levels are changed continuously during the training until training results with the same labels still yellow outputs.

The model is made up of parameters and architecture. The value of the parameters determines how precise the model executes the task for a given architecture. So what positive qualities do you find? In determining a loss function, which assesses the efficiency of the model *(1)*. The aim is to minimise the loss and thereby find parameter values that correlate with specific forecasts.

Establishing the optimisation dilemma

In different activities, the loss function can depend on the desired output. How you interpret, it affects the preparation and success of the model. Example: House price prognostication. We will use the durability of the land, the number of bedrooms and the height of the ceiling to estimate the price of the house. The quadrature failure attribute is:

$$L = \| y - \hat{y} \|^2_2 \quad (1)$$

where *y* is your predicted price and *y* is the actual price, also known as ground truth.

Cost function: Note that the loss L Take one example as feedback, such that the minimisation of model parameters for other instances does not guarantee better. The average failure measured over the whole training data collection should be minimised;

$$J = (1/m)\sum_{i=1}^{m} L^{(i)} \quad (2)$$

This attribute is what we call the rate. m is the training data set a scale and $\mathcal{L}^{(i)}$ loss of a single example of instruction $x^{(i)}$ labelled $y^{(i)}$

The model versus the cost function:

It is essential to distinguish between the function *f* that will perform the task (the model), and the function J you are optimising (the cost function). The input models feature (for example, floor and bedroom no.), and the price of the house is output. The structure and a set of parameters are defined, and a real function which performs the task is approximated. The model is allowed to execute the task with relative precision by optimising parameter values *(2)*. The cost function adds a collection of parameters and produces a cost and tests how well the collection of parameters (on the training set) executes the mission.

Augmenting the cost-utility

Healthy parameter values are unclear in the beginning. Yet you have a cost-function formula. Minimise the cost function, and you can find reasonable values for parameters in theory. This is achieved by feeding a collection of training data into the model and iteratively changing the parameter to reduce the costs to a minimum. In summary, the way you interpret the cost function defines the model's output for the mission.

RMSProp

RmsProp is an optimiser used for the normalisation of gradients in the magnitude of recent gradients. We still maintain a sweeping average above the mean root gradient, with Rms separating the current gradient *(3,4)*. Let $f'(\theta_t)$ be the derivative of the loss concerning the parameters at time step t. In some cases, adding a momentum term β is beneficial. Here, Nesterov momentum is used:

$$\theta_{t+1/2} = \theta t - \beta v_t \quad (3)$$

$$r_t = (1-\gamma)\, f'(\theta_{t+1/2})^2 + \gamma r_{t-1} \quad (4)$$

$$v_{t+1} = \beta v_t + \alpha/\sqrt{rt} * f'(\theta_{t+1/2}) \quad (5)$$

$$\theta_{t+1} = \theta_t - v_{t+1} \quad (6)$$

Furthermore, customisable phase speeds occur in this application. The step rate for that parameter is multiplied with 1 + step adapt when the step components and the momentum point to the same direction (thus with the same sign). It is compounded by 1-step adapt otherwise. In this way, the step rate min and step rate max minimum and maximum step thresholds are respected and truncated to satisfy values. For one, it is a very powerful optimiser with pseudo curvature knowledge. RmsProp has many advantages. It can also be very effective at managing stochastic targets, making it applicable to studying in miniatures.

Adam

Another approach that measures the adaptive learning rate of human parameters is Adaptive Moment Estimation (Adam). Adam also retains an exponentially decreasing average of prior quadrature gradients v_t like Adadelta and RMSprop, which is close to momentum. Whereas momentum can be assumed to be a pitched ball, Adam acted as a hardball with friction, choosing the error surface to have a low degree *(5)*. The first (middle) and second (un-centred variance) moment of the gradients are m_t and v_t, which is the method's name. Since m_t and v_t are initialised as vectors from the age of 0, the authors of Adam note that they are prejudicial towards zero, mainly when the decay rates are lower (i.e. β_1 and β_2 are near 1). When measuring first and second-moment biases, you combat these biases:

$$\hat{m}_t = m_t/(1-\beta_1^t) \quad (7)$$
$$\hat{v}_t = v_t/(1-\beta_2^t) \quad (8)$$

You then update the parameters as in Adadelta and RMSprop, which gives the updated law for Adam:

$$\theta_{t+1} = \theta_t - \eta/(\sqrt{\hat{v}t} + \varepsilon * \hat{m}_t \quad (9)$$

The authors propose default values of 0.9 for β1, 0.999 for β2, and 10^{-8} for ϵϵ. They empirically prove that Adam performs well in reality and correlates favourably with other algorithms for adaptive learning approaches.

Recurrent Neural Network

Recent neural networks are a category of neural networks known as RNNs that allow previous outputs to be used as inputs while hidden.

For each time step t, the activation $a^{<t>}$ and the output $y^{<t>}$ is expressed as follows:

$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$ (10)

$y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$ (11)

Where W_{ax}, W_{aa}, W_{ya}, b_a, b_y are constants that are distributed temporally and g_1, g_2 activation functions.

$\Gamma = \sigma(Wx^{<t>} + Ua^{<t-1>} + b)$ (12)

Where W, U, b are gate-specific coefficients, and μ is sigmoid. The most significant are summarised in Table 1.

GRU/LSTM The Gated Recurrent Unit (GRU) and LSTM are devoted to the issue of the disappearance gradient of typical RNNs, and LSTM is a general GRU. Table 2 is the summarises each architectural equation *(6)*.

Audio Preprocessing

Audio preprocessing refers to the process of converting the audio clip into a suitable form which can be easily understood by the model. Thus in audio preprocessing, we tend to extract features from the audio clip, which will be fed to the model. Various features can be extracted from the clips, namely Mel-Frequency Cepstral Coefficients, Spectral Centroid, Spectral roll-off, Chroma Frequencies, Zero Crossing Rate etc. You can use any one or more than one to generate your feature vectors. In our model, we will only use the Mel-Frequency Spectral Coefficients for our feature vector. Mel-Frequency Cepstral Coefficients(MFCC), What is the Mel scale? The Mel scale refers to the fundamental frequency, the perceived frequency or the pitch, of a single note. People detect minor pitch variations at low frequencies much better than those at high frequencies. With this size, our traits approximate what humans hear more closely. Conversion formula from frequency to Mel is:

$M(f) = 1125\ln(1 + f/700)$ (13)

To go from Mels back to frequency:

$M^{-1}(m) = 700(\exp(m/1125) - 1)$ (14)

The fundamental mechanics for these audio preprocessing methods rely on Fourier Transform, whose definition follows.

$f\ [\![\xi]\!] = \int_{-\infty}^{\infty} f(x)e^{-2\pi ix\xi}dx$ (15)

In the above equation, f is the original function of time where x represents time. ˆf is the transformed function of frequency where ⇠ represents frequency. A Fourier transformation (FT) is a mathematical transform in mathematics, which breaks down a function (often a function of time or signal) into its frequencies *(7)*. The analysis of the Fourier series is one inspiration for the Fourier Transformation. In the analysis of the Fourier Series, the mixture of simple waves, defined mathematically by sines and cosines is complicated but periodic functions.

Dataset

We have used the GTZAN Dataset from the website Kaggle. The dataset is also used.This dataset is widely used in the field of Music Genre Recognition(MGR) since its initial release in 2002. It consists of 10 genres, each containing 100 audio clips in .wav format. Each audio clip has a length of 30 seconds, are 22050Hz Mono 16-bit files. The dataset incorporates samples from a variety of sources like

CDs, radios, microphone recordings etc. We split the dataset in 0.8: 0.2 ratio. We have used the librosa package in Python to extract MFCC features. The ten genres are blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae and rock with each being 100.

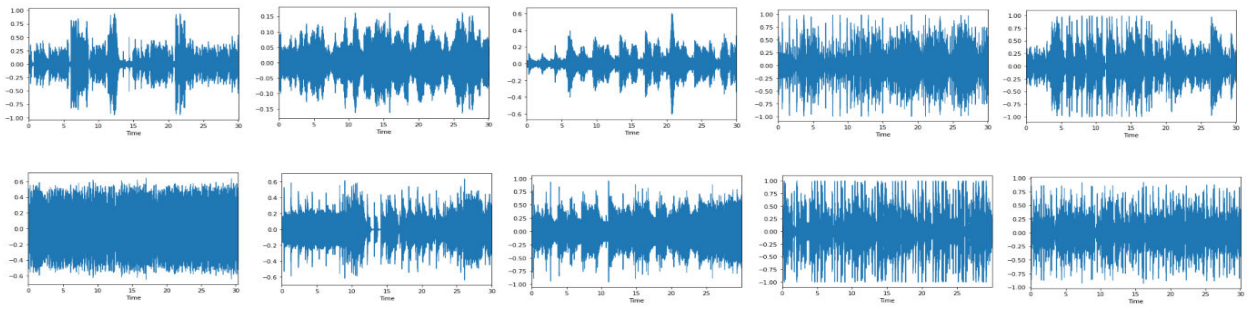

Fig. 1. Waveplots of various genres included in the dataset Classical, Blues, Rock and Pop

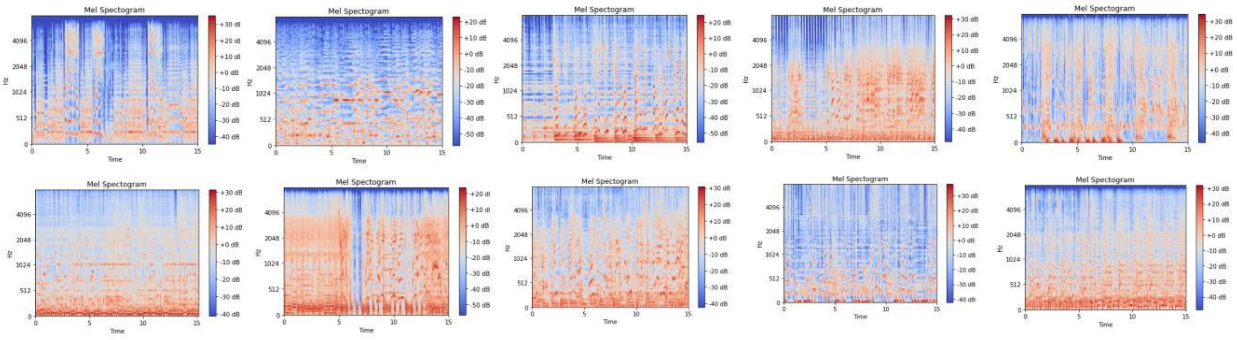

Fig. 2. Mel-Spectrograms of various genres included in the dataset Classical, Blues, Rock and Pop

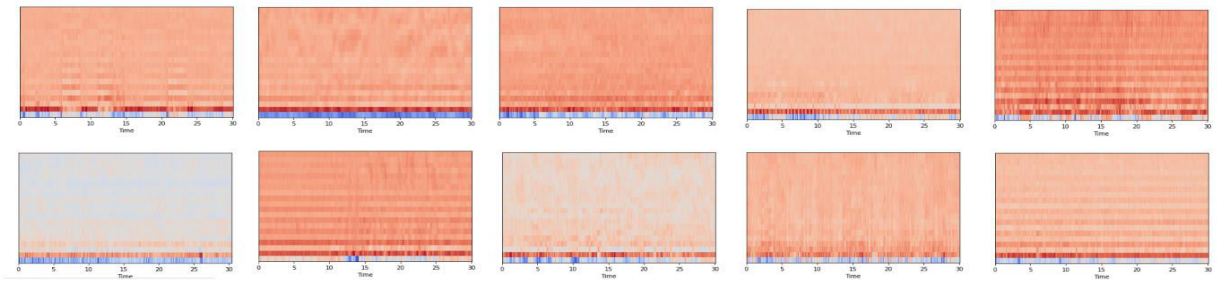

Fig. 3. MFCC's of various genres included in the dataset Classical, Blues, Rock and Pop

Model Description

Recurrent Neural Network (RNN's) are mostly used on tasks involving sequential data such as speech recognition, machine translation, sentiment analysis etc. . music shares a sequential nature with speech and text, as the flow from one note to the next determines the mood of melody and hence the genre. Given this knowledge, recurrent neural networks seemed like the logical next step. We are using Bidirectional RNN's to classify the genre of an audio clip. The paper entails comparing 3 optimisation algorithms, namely SGD, Adam, RMSprop and Rectified Adam on 2 different architectures *(8)*. The only difference between the first and second architecture is that the first layer of the first model involves an LSTM layer, while the first layer of the model involves a GRU layer. We chose Tensorflow as the preferred choice for our deep learning library because of its high flexibility and excellent functionality. The other libraries involved were Keras, NumPy, sci-kit learn, librosa etc.

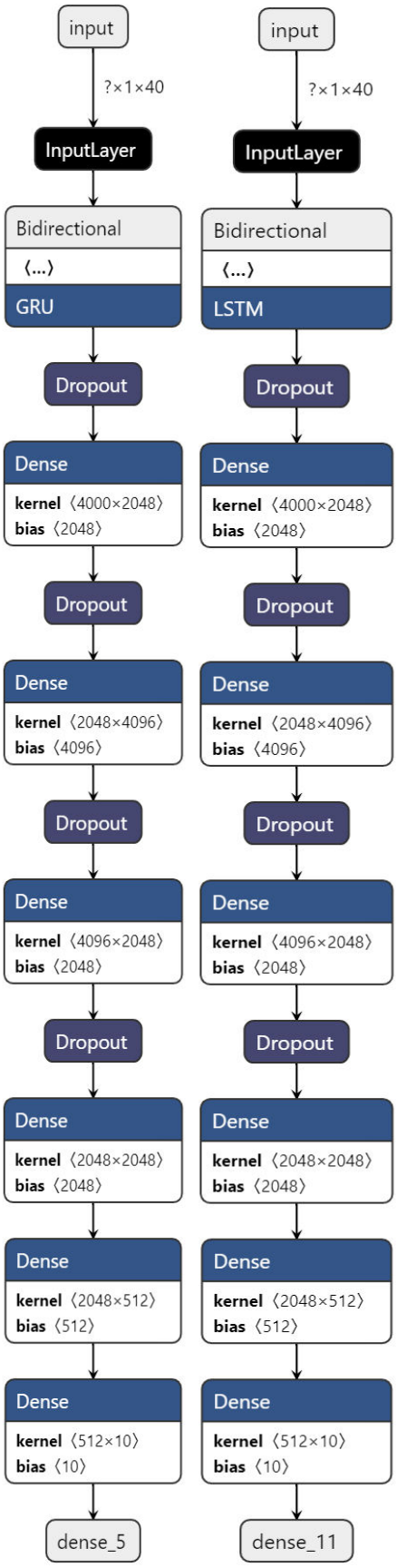

Fig. 4. The architecture of the First and Second Model

Results

After compiling the model, we ran it for 200 epochs with a batch size of 150. To train the model, we employed the 80%-20% splitting strategy was used for training and testing set, respectively. The accuracy was computed using the formula given below:-

$$Accuracy = (Number\ of\ Songs\ correctly\ classified)/(\text{Total number of Songs}) * 100 \quad (16)$$

Results for various optimisation algorithms for different architectures are given in the following subsections, Stochastic Gradient Descent. After training the model with a batch size of 150 over 200 epochs with the SGD optimiser with a knowledge rate of 0.001 along with a decay rate of 0.01, we obtained the results of the following models. For the GRU Model, we obtained an accuracy of 20.53% on the training set and 32.50% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 8.

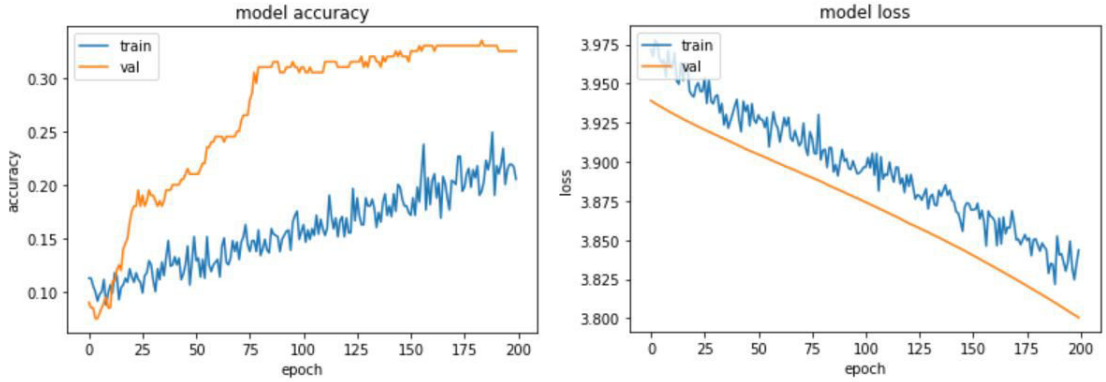

Fig. 5. Variation of the model's accuracy and loss during training

For the LSTM Model, we obtained an accuracy of 14.14% on the training set and 30.50% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 9.

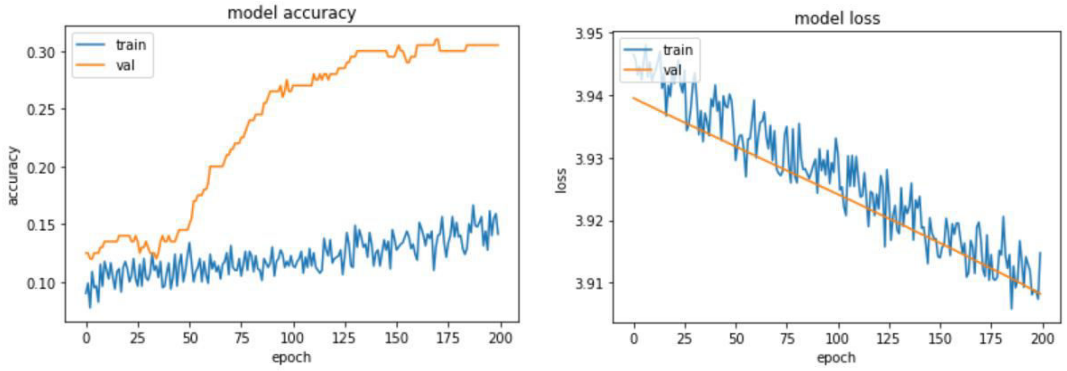

Fig. 5. Variation of the model's accuracy and loss during training

RMSprop: After training the model with a batch size of 150 over 200 epochs with the SGD optimiser with a discovering rate of 0.001 along with a decay rate of 0.01, we obtained the results of the following models. GRU Model: For the GRU Model, we obtained an accuracy of 99.12% on the training set and

47.50% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 10.

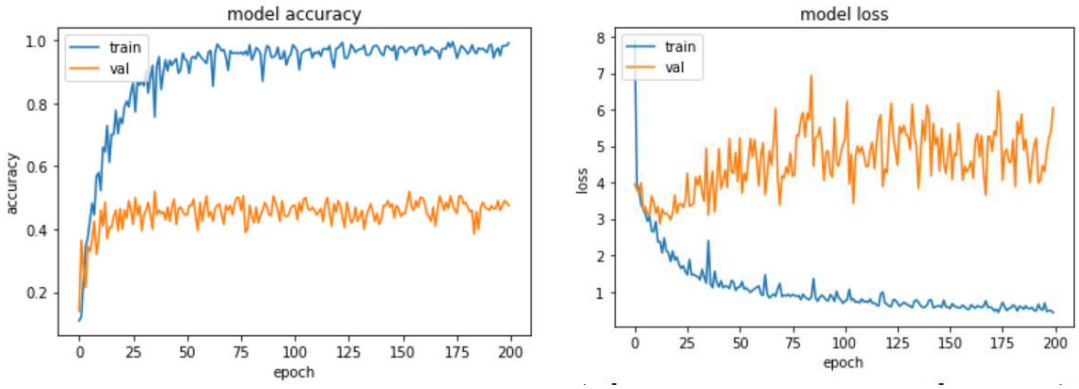

Fig. 6. Variation of the model's accuracy and loss during training

For the LSTM Model, we obtained an accuracy of 96.62% on the training set and 45.00% on the validation set.

Adam: After training the model with a batch size of 150 over 200 epochs with the SGD optimiser with a realising rate of 0.001 along with a decay rate of 0.01, we obtained the results of the following models. GRU Model: For the GRU Model, we obtained an accuracy of 98.75% on the training set and 45.50% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 11 (a,b). For the LSTM Model, we obtained an accuracy of 97.00% on the training set and 45.00% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 11(c,d).

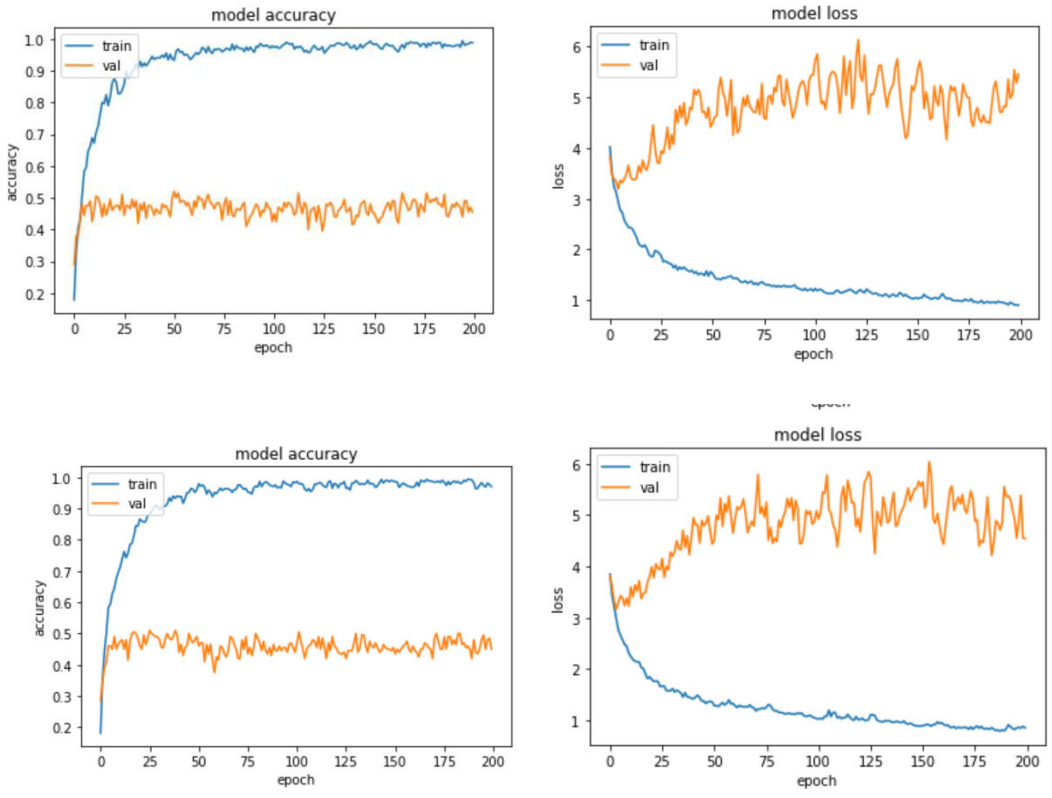

Fig. 7. Variation of the model's accuracy and loss during training

Rectified Adam: After training the model with a batch size of 150 over 200 epochs with the SGD optimiser with a studying rate of 0.001 along with a decay rate of 0.01, we obtained the results of the subsequent models. GRU Model: For the GRU Model, we obtained an accuracy of 98.25% on the training set and 46.00% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 12(a,b). For the LSTM Model, we obtained an accuracy of 98.25% on the training set and 51.50% on the validation set. Variation of the model's accuracy and loss during training is depicted in Figures 12(c,d).

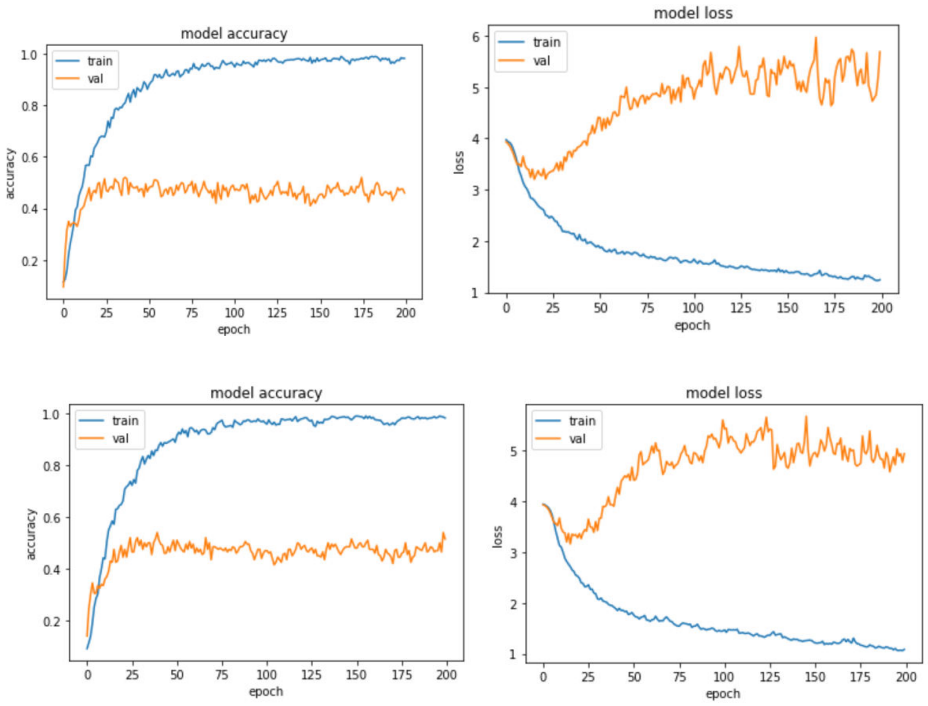

Fig. 8. Variation of the model's accuracy and loss during training

The results obtained above indicate overfitting in our model. This can be attributed due to the following reasons –

i) Training the model for a more massive no. of epochs
ii) Less training data

Conclusion and Future Work

The role of classifying music genres is explored using audio set data in this work. To solve this dilemma, we suggest two different methods. Firstly, a spectrogram should be produced and processed for the audio signal. In these photos, the A CNN dependent image classifier, VGG-16, predicts the music based on this spectrogram. The second approach consists of the extraction of time and frequency domain features from audio signals and the preparation of standard machine learning classifiers. The CNN-based deep learning models have proved to be supervisory to functionality-based models; the most significant characteristics are also mentioned. We also demonstrate that it has been helpful to assemble the CNN and XGBoostmodels. It is worth noting that the dataset used in this analysis were audio clips of YouTube videos that are very distorted in general. Scientific investigations can recognise the ways that chaotic data can indeed be preprocessed before they are fed into a prototype of artificial intelligence to boost perforation.

Conflict of Interest: The authors declare that they have no conflict of interest.

References

[1] Abdel-Hamid, O., Mohamed, A. R., Jiang, H., Deng, L., Penn, G., & Yu, D. (2014). Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, *22*(10), 1533-1545.

[2] Amit, Y., & Geman, D. (1997). Shape quantisation and recognition with randomised trees. *Neural computation*, *9*(7), 1545-1588.

[3] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

[4] STEVEN B DAVIS MEMBER, I. E. E. E., & PAUL MERMELSTEIN SENIOR MEMBER, I. E. E. E. (1990). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *Readings in speech recognition* (pp. 65-74). Morgan Kaufmann.

[5] Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., ... & Ritter, M. (2017, March). Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 776-780). IEEE.

[6] Gouyon, F., Pachet, F., & Delerue, O. (2000, December). On the use of zero-crossing rate for an application of classification of percussive sounds. In *Proceedings of the COST G-6 conference on Digital Audio Effects (DAFX-00), Verona, Italy* (Vol. 5).

[7] Grosche, P., Müller, M., & Kurth, F. (2010, March). Cyclic tempogram—A mid-level tempo representation for music signals. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5522-5525). IEEE.

[8] Bahuleyan, H. (2018). Music genre classification using machine learning techniques. *arXiv preprint arXiv:1804.01149*.

Table 1: Significant summary of W, U, b are gate-specific coefficients, and μ is sigmoid

Type of gate	Role	Used in
Update gate Γ_u	How much past should matter now?	GRU, LSTM
Relevance gate Γ_r	Drop previous information?	GRU, LSTM
Forget gate Γ_f	Erase a cell or not?	LSTM
Output gate Γ_o	How much to reveal of a cell?	LSTM

Table 2: is the summarises each architectural equation.

Characterization	**Gated Recurrent Unit** (GRU)	**Long Short-Term Memory** (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u \star \tilde{c}^{<t>} + (1 - \Gamma_u) \star c^{<t-1>}$	$\Gamma_u \star \tilde{c}^{<t>} + \Gamma_f \star c^{<t-1>}$
$a^{<t>}$	$c^{<t>}$	$\Gamma_o \star c^{<t>}$
Dependencies