Music Genre Classification using Machine Learning Techniques

Mudavath Vishnuvardhan
Dept. of Electrical Engineering
Indian Institute of Technology, Bombay
200070044@iitb.ac.in

Jay Charel
Dept. of Electrical Engineering
Indian Institute of Technology, Bombay
20d070022@iitb.ac.in

Arpita Gajanan Joshi
Dept. of Energy Science and Engineering
Indian Institute of Technology, Bombay
20d170004@iitb.ac.in

***Abstract*—This study explores the challenging task of categorizing music files into genres in the field of music information retrieval (MIR). Two approaches are compared: a deep learning approach using a CNN model trained solely on spectrograms to predict the genre, and a conventional machine learning approach utilizing manually extracted features from the time and frequency domains. Four classifiers are trained with the features, and the ones that contribute the most to the classification task are identified. The experiments are conducted on the Audio set data set, and the study concludes by suggesting the effectiveness of an ensemble classifier that combines both approaches.**

***Index Terms*—MEL-Spectrograms, VGG-16-Transfer learning, VGG-16-Fine tuning, CNN, Logistic Regression, Random Forest Classifier, Gradient boost classifier, MFCC.**

I. Introduction

As music streaming services like Spotify and similar apps become increasingly popular, the manual classification of music by genre has become a thing of the past. These platforms now use machine learning algorithms to automatically classify and organize songs based on their characteristics, such as rhythmic structure, harmonic content, and instrumentation. It is still important to understand the underlying techniques used for genre classification. This understanding can aid in the development and optimization of these platforms, leading to a better user experience for music enthusiasts.
In this research paper, we explore the application of machine learning algorithms for genre classification of audio files, utilizing both convolutional neural networks and traditional machine learning models with features extracted from the time and frequency domains of the audio signal. The models are evaluated on the Audio Set dataset, and their performance is compared, along with an analysis of the importance of different features. The results of this study can provide insights into the development of more effective genre classification algorithms for music streaming services, ultimately leading to a better user experience for music enthusiasts in today's era of music streaming.

II. Dataset

In this project, we leveraged the Audio Set, a vast collection of human-annotated database of sounds (Gemmeke et al., 2017), to explore music genre classification. The dataset, created by extracting 10-second sound clips from over 2.1 million YouTube videos, has been annotated based on an ontology that covers 527 classes of sounds, including musical instruments, speech, vehicle sounds, animal sounds, and more. Our focus was on audio files belonging to the music category, specifically with one of the seven genre tags listed in Table 1, and we recorded the number of audio clips in each category.

	Genre	**Count**
1	Pop Music	4751
2	Rock Music	4614
3	Hip Hop Music	3950
4	Techno	5537
5	Rhythm Blues	3055
6	Vocal	1851
7	Reggae Music	1469
	Total	25227

Table 1: Counts of each genre label

While the Audio Set data release did not provide the raw audio clips of these sounds, it did provide the corresponding YouTubeIDs, along with their start and end times. Hence, the initial step involved retrieving these audio files using yt-dlp, a command-line program that can download videos in mp4 format. A total of 25,227 files were retrieved. Then ffmpeg was used which is an audio converter, to convert the mp4 files into the desired .wav format. Each .wav files were approximately 880 KB in size, and the total data used in this project amounted to approximately 25.3 GB. Program we have used to extract this .wav files is placed in 1_audio_retrieval.ipynb file.

III. Methodology

This section provides the details of the data pre-processing steps followed by the description of the two proposed approaches to this classification problem.

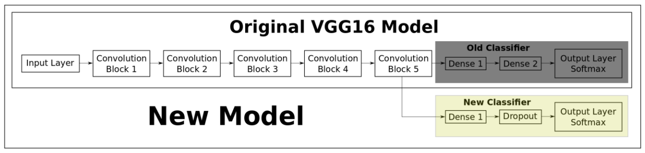

Figure 2: Convolutional neural network architecture (Image Source:[Hvass Tensorflow Tutorials]

A. Data Pre-processing

In order to improve the Signal-to-Noise Ratio (SNR) of the signal, a pre-emphasis filter,given by Equation is applied to the original audio signal.

$$y(t) = x(t) - \alpha * x(t-1)$$

B. Deep Neural Networks

Using deep learning, we can achieve the task of music genre classification without the need for hand-crafted features. Convolutional neural networks (CNNs) have been widely used for the task of image classification (Krizhevsky et al., 2012). The 3-channel (RGB) matrix representation of an image is fed into a CNN which is trained to predict the image class. In this study, the sound wave can be represented as a spectrogram, which in turn can be treated as an image (Nanni et al., 2016)(Lidy and Schindler, 2016). The task of the CNN is to use the spectrogram to predict the genre label (one of seven classes).

1) Spectrogram Generation: A spectrogram is a 2D representation of a signal, having time on the x-axis and frequency on the y-axis. A colormap is used to quantify the magnitude of a given frequency within a given time window. In this study, each audio signal was converted into a MEL spectrogram (having MEL frequency bins on the y-axis). The parameters used to generate the power spectrogram using STFT are listed below:

- Sampling rate (sr) = 22050
- Frame/Window size (n fft) = 2048
- Time advance between frames (hop size) = 512 (resulting in 75% overlap)
- Window Function: Hann Window
- Frequency Scale: MEL
- Number of MEL bins: 96
- Highest Frequency (f max) = sr/2

2) Convolutional Neural Networks: From the Figure 1, one can understand that there exists some characteristic patterns in the spectrograms of the audio signals belonging to different classes. Hence, spectrograms can be considered as 'images' and provided as input to a CNN, which has shown good performance on image classification tasks. Each block in a CNN consists of the following operations:

- **Convolution:** This step involves sliding a matrix filter (say 3x3 size) over the input image which is of dimension image width x image_height. The filter is first placed on the image matrix and then we compute an element-wise multiplication between the filter and the overlapping portion of the image, followed by a summation to give a feature value. We use many such filters , the values of which are 'learned' during the training of the neural network via backpropagation.
- **Pooling:** This is a way to reduce the dimension of the feature map obtained from the convolution step, formally know as the process of down sampling. For example, by max pooling with 2x2 window size, we only retain the element with the maximum value among the 4 elements of the feature map that are covered in this window. We keep moving this window across the feature map with a pre-defined stride.
- **Non-linear Activation:** The convolution operation is linear and in order to make the neural network more powerful, we need to introduce some non-linearity. For this purpose, we can apply an activation function such as ReLU4 on each element of the feature map

In this study, a CNN architecture known as VGG-16, which was the top performing model in the ImageNet Challenge 2014 (classification + localization task) was used (Simonyan and Zisserman, 2014). The model consists of 5 convolutional blocks (conv base), followed by a set of densely connected layers, which outputs the probability that a given image belongs to each of the possible classes.

For the task of music genre classification using spectrograms, we download the model architecture with pre-trained weights, and extract the conv base. The output of the conv base is then send to a new feed-forward neural network which in turn predicts the genre of the music.

There are two possible settings while implementing the pre-trained model:

- **Transfer learning:** The weights in the conv-base are kept fixed but the weights in the feed-forward network (represented by the yellow box in Figure 2) are allowed to be tuned to predict the correct genre label.
- **Fine tuning:** In this setting, we start with the pre-trained weights of VGG-16, but allow all the model weights to be tuned during training process.

The final layer of the neural network outputs the class probabilities (using the softmax activation function) for each of the seven possible class labels. Next, the cross-entropy loss is computed as follows:

$$l = -\sum_{c=1}^{M} y_{o,c} * log(p_{o,c})$$

where, M is the number of classes; $y_{o,c}$ is a binary indicator whose value is 1 if observation o belongs to class c and 0 otherwise; $p_{o,c}$ is the model's predicted probability that observation o belongs to class c. This loss is used to back propagate the error, compute the gradients and thereby update the weights of the network. This iterative process continues until the loss converges to a minimum value.

3) Implementation Details: The spectrogram images have a dimension of 216 x 216. For the feed-forward network connected to the conv base, a 512-unit hidden layer is implemented. Over-fitting is a common issue in neural networks. In order to prevent this, two strategies are adopted:

- **L2-Regularization:** The loss function of the neural network is added with the term $\lambda \sum_i w_i^2$, where w refers to the weights in the neural networks. This method is used to penalize excessively high weights. We would like the weights to be diffused across all model parameters, and not just among a few parameters. Also, intuitively, smaller weights would correspond to a less complex model, thereby avoiding overfitting. λ is set to a value of 0.001 in this project.

- **Dropout:** This is a regularization mechanism in which we shutoff some of the neurons (set their weights to zero) randomly during training. In each iteration, we thereby use a different combination of neurons to predict the final output. This makes the model generalize without any heavy dependence on a subset of the neurons.

The dataset is randomly split into train (90%), validation (5%) and test (5%) sets. The same split is used for all experiments to ensure a fair comparison of the proposed models. The neural networks are implemented in Python using Tensorflow 5 ;VGG transfer learning model was trained for 6 epochs wereas VGG fine tuning model and feed forward learning were trained for 3 epochs beacuse low processing power of GPU took approx. 9 hrs to train fine tuning model wereas took nearly 7hrs to train transfer learning model with a batch size of 32 with the ADAM optimizer (Kingma and Ba, 2014). One epoch refers to one iteration over the entire training dataset.

Figure 3 shows the learning curves - the loss (which is being optimized) keeps decreasing as the training progresses. Although the training accuracy keeps increasing, the validation accuracy first increases and after a certain number of epochs, it starts to decrease. This shows the model's tendency to overfit on the training data. The model that is selected for evaluation purposes is the one that has the highest accuracy and lowest loss on the validation set (epoch 2 in Figure 3).

4) Baseline Feed-forward Neural Network: To assess the performance improvement that can be achieved by the CNNs, we also train a baseline feed-forward neural network that takes as input the same spectrogram image. The image which is a 2-dimensional vector of pixel values is unwrapped or flattened into a 1-dimensional vector. Using this vector, a simple 2-layer neural network is trained to predict the genre of the audio signal. The first hidden layer consists of 512 units and the second layer has 32 units, followed by the output layer. The activation function used is ReLU.

C. Manually Extracted Features

In this section, we describe the second category of proposed models, namely the ones that require hand-crafted features to be fed into a machine learning classifier. Features can be broadly classified as time domain and frequency domain features. The feature extraction was done using librosa, a Python library.

1) Time Domain Features: These are features which were extracted from the raw audio signal.

- **Central moments:** This consists of the mean, standard deviation, skewness and kurtosis of the amplitude of the signal.
- **Zero Crossing Rate (ZCR):** The entire 10 second signal is divided into smaller frames, and the number of zero-crossings present in each frame are determined. The frame length is chosen to be 2048 points with a hop size of 512 points. Note that these frame parameters have been used consistently across all features discussed in this section. Finally, the average and standard deviation of the ZCR across all frames are chosen as representative features.
- **Root Mean Square Energy (RMSE):** The energy in a signal is calculated as:

$$\sum_{n=1}^{N} |x(n)|^2$$

Further, the root mean square value can be computed as:

$$\sqrt{1/N * \sum_{n=1}^{N} |x(n)|^2}$$

RMSE is calculated frame by frame and then we take the average and standard deviation across all frames.

- **Tempo:** In general terms, tempo refers to the how fast or slow a piece of music is; it is expressed in terms of Beats Per Minute (BPM). Intuitively, different kinds of music would have different tempos. Since the tempo of the audio piece can vary with time, we aggregate it by computing the mean across several frames. The functionality in librosa first computes a tempogram following and then estimates a single value for tempo.

2) Frequency Domain Features: The audio signal can be transformed into the frequency domain by using the Fourier Transform.We then extract the following features.

- **Mel-Frequency Cepstral Coefficients (MFCC):** the Short-Time Fourier Transform (STFT) of the signal is taken with n fft=2048 and hop size=512 and a Hann window. Next, we compute the power spectrum and then apply the triangular MEL filter bank, which mimics the human perception of sound. This is followed by taking the discrete cosine transform of the logarithm of all filterbank energies, thereby obtaining the MFCCs.
- **Chroma Features:** This is a vector which corresponds to the total energy of the signal in each of the 12 pitch classes. (C, C, D, D, E ,F, F, G, G, A, A, B) (Ellis, 2007). The chroma vectors are then aggregated across the frames to obtain a representative mean and standard deviation.
- **Spectral Centroid:** For each frame, this corresponds to the frequency around which most of the energy is centered (Tjoa, 2017). It is a magnitude weighted frequency calculated as:

$$f_c = \frac{\sum_{n=k} S(k)f(k)}{\sum_{n=k} fk}$$

where S(k) is the spectral magnitude of frequency bin k and f(k) is the frequency corresponding to bin k.

- Spectral Band-width: The p-th order spectral band-width corresponds to the p-th order moment about the spectral centroid (Tjoa, 2017) and is calculated as

$$[\sum_{n=k}(S(k)f(k) - f_c)^p]^{1/p}$$

For example, p = 2 is analogous to a weighted standard deviation.
- Spectral Contrast: Each frame is divided into a pre-specified number of frequency bands. And, within each frequency band, the spectral contrast is calculated as the difference between the maximum and minimum magnitudes (Jiang et al., 2002).
- Spectral Roll-off: This feature corresponds to the value of frequency below which 85% (this threshold can be defined by the user) of the total energy in the spectrum lies (Tjoa, 2017).

For each of the spectral features described above, the mean and standard deviation of the values taken across frames is considered as the representative final feature that is fed to the model. The features described in this section would be would be used to train machine learning algorithms (refer Section 4.4). The features that contribute the most in achieving a good classification performance will be identified and reported.

D. *Classifiers*

This section provides a brief overview of the four machine learning classifiers used in this project.

- **Logistic Regression (LR):** For this multi-class classification task, the LR is implemented as a one-vs-rest method. That is, 7 separate binary classifiers are trained. During test time, the class with the highest probability from among the 7 classifiers is chosen as the predicted class.
- **Random Forest (RF):** Random Forest is a ensemble learner that combines the prediction from a pre-specified number of decision trees.
- **Gradient Boosting (XGB):** Boosting is another ensemble classifier that is obtained by combining a number of weak learners (such as decision trees). However, unlike RFs, boosting algorithms are trained in a sequential manner using forward stagewise additive modelling. XGB refers to extreme Gradient Boosting, which is an implementation of boosting that supports training the model in a fast and parallelized manner.
- **Support Vector Machines (SVM):** SVMs transform the original input data into a high dimensional space using a kernel trick . The transformed data can be linearly separated using a hyperplane. The optimal hyperplane maximizes the margin. In this study, a radial basis function (RBF) kernel is used to train the SVM because such a kernel would be required to address this non-linear problem. Similar to the logistic regression setting discussed above, the SVM is also implemented as a one-vs-rest classification task.

IV. Result

Metrics chosen for evaluating models are:

- **Accuracy :** Percentage of correctly classified test samples.
- **F-score :** The harmonic mean between precision and recall.
- **AUC :** The ROC is a graph between the True positive rate and the false positive rate.

The best performance in terms of all metrics is observed for the convolutional neural network model based on VGG-16 that uses only the spectrogram to predict the music genre. It was expected that the fine tuning setting, which additionally allows the convolutional base to be trainable, would enhance the CNN model when compared to the transfer learning setting. However, as shown in Table 2, the experimental results show that there is no significant difference between transfer learning and fine-tuning. The baseline feed-forward neural network that uses the unrolled pixel values from the spectrogram performs poorly on the test set. This shows that CNNs can significantly improve the scores on such an image classification task. Among the models that use manually crafted features, the one with the least performance is the Logistic regression model. This is expected since logistic regression is a linear classifier. SVMs outperform random forests in terms of accuracy. However, the XGB version of the gradient boosting algorithm performs the best among the feature engineered methods.

	Accuracy	**F-score**	**ROC AUC**
Spectrogram-based models			
VGG-16 CNN Transfer Learning	0.52	0.48	0.832
VGG-16 CNN Fine Tuning	0.59	0.55	0.866
Feed-forward NN baseline	0.36	0.27	0.720
Feature Engineering based models			
Logistic Regression (LR)	0.49	0.44	0.809
Random Forest (RF)	0.51	0.44	0.829
Support Vector Machines (SVM)	0.50	0.43	0.808
Extreme Gradient Boosting (XGB)	0.53	0.48	0.850
Ensemble Classifiers			
VGG-16 CNN + XGB	0.61	0.56	0.874

Table 2: Comparison of performance of the models on the test set

As can be observed from Figure 4, Mel-Frequency Cepstral Coefficients (MFCC) appear the most among the important features. Previous studies have reported MFCCs to improve the performance of speech recognition systems . Our project show that MFCCs contribute significantly to this task of music genre classification. The mean and standard deviation of the spectral contrasts at different frequency bands are also important features. The music tempo, calculated in terms of beats per minute also appear in the top 20 useful features.

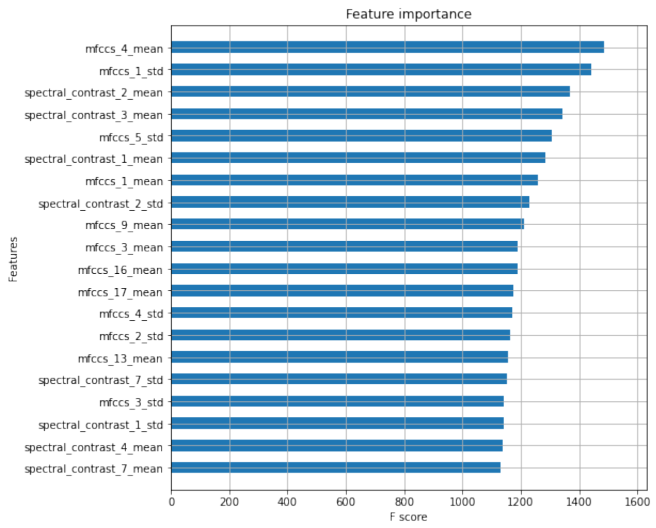

Figure 3: Relative importance of features in the XGBoost model; the top 20 most contributing features are displayed

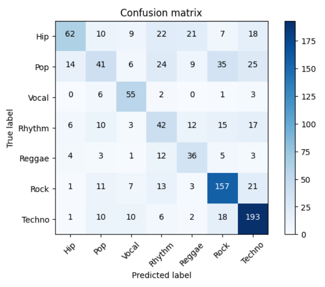

Figure 4: VGG-16 CNN Fine tuning Learning Confusion matrix

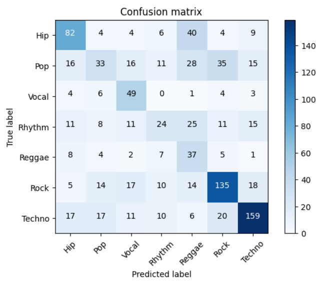

Figure 5: VGG-16 CNN Transfer Learning Confusion matrix

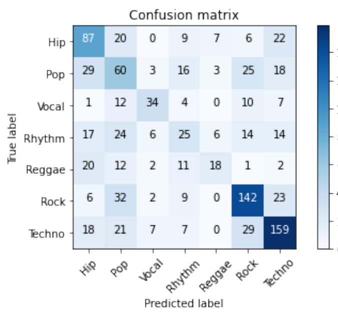

Figure 6: Extreme Gradient Boosting Confusion matrix

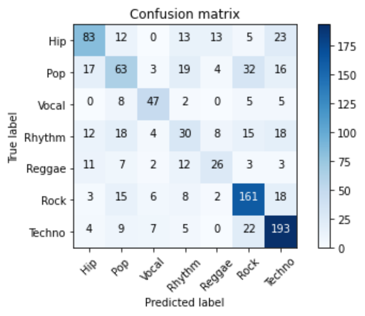

Figure 7: Ensemble Model Confusion matrix

In this project, the best CNN model namely, VGG-16 Transfer Learning is ensembled with XGBoost the best feature engineered model by averaging the predicted probabilities. As shown in Table 2, this ensembling is beneficial and is observed to outperform the all individual classifiers. The ROC curve for the ensemble model is above that of VGG-16 Fine Tuning and XGBoost as illustrated in Figure 8.

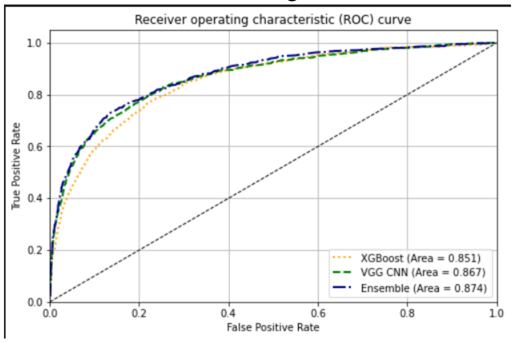

Figure 8: ROC Curves for the best performing models and their ensemble

V. Conclusion

The present report explores the problem of music genre classification using the Audioset dataset. Two different approaches were proposed to solve this task. The first approach involved generating a spectrogram of the audio signal and treating it as an image. A CNN-based image classifier, specifically VGG-16, was trained on these images to predict the music genre based solely on the spectrogram. The second approach consisted of extracting time and frequency domain features from the audio signals, followed by training traditional machine learning classifiers based on these features. Among the feature-based classifiers, XGBoost was determined to be the best; the most important features were also identified. Our results indicate that the CNN-based deep learning models outperformed the feature-engineered models. Additionally, ensembling the CNN and XGBoost models proved to be beneficial. It is worth noting that the dataset used in this study consisted of audio clips from YouTube videos, which are generally very noisy. Future studies could focus on pre-processing this noisy data before feeding it into a machine-learning model to achieve better performance.

VI. Creativity of the problem

One of the creative aspects of this project is the utilization of the Audioset data, which is a large-scale human annotated database of sounds. This dataset provides a rich source of audio clips that are annotated with detailed labels based on an ontology covering 527 classes of sounds. By leveraging this dataset, our project aims to improve the accuracy of music genre classification, which is a task of great interest to audio streaming services like Spotify and iTunes.

The approach of using a CNN-based image classifier to classify music genres based on spectrograms is an innovative and novel solution to this problem. Spectrograms are a visual representation of the frequency content of an audio signal over time, and treating them as images allows us to leverage the power of convolutional neural networks, which have been shown to be highly effective in image classification tasks.

Similarly, the use of traditional machine learning classifiers like XGBoost to classify music genres based on time and frequency domain features extracted from audio signals is also a creative solution to the problem. This approach allows us to use a wide range of feature engineering techniques and statistical models to improve the accuracy of the classification task.

Overall, the creativity of this problem lies in the combination of different approaches and techniques to classify music genres, which can lead to more accurate and automated classification of music in large-scale audio databases.

VII. Statement of Contribution

1 Audio Retrieval, Plot Spectrogram : Jay
2 VGG Model Transfer Learning, Fine Tuning and FFB : Vishnuvardhan
3 Model Building : Arpita
4 Report Writing : Vishnuvardhan and Arpita
5 Backgroung Research : Jay
6 Video Creation : Vishnuvardhan

VIII. Research Paper

Music Genre Classification Research paper

IX. References

1. Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. IEEE/ACM Transactions on audio, speech, and language processing 22(10):1533–1545.

2. Yali Amit and Donald Geman. 1997. Shape quantization and recognition with randomized trees. Neural computation 9(7):1545–1588.

3. Leo Breiman. 1996. Bagging predictors. Machine learning 24(2):123–140.

4. Leo Breiman. 2001. Random forests. Machine learning 45(1):5–32.

5. Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. Machine learning 20(3):273–297.

6. Steven B Davis and Paul Mermelstein. 1990. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In Readings in speech recognition, Elsevier, pages 65–74.