IDENTIFICATION OF MUSIC GENRE USING CNN

PROJECT REPORT

21AD1513- INNOVATION PRACTICES LAB

Submitted by

MUTHUPANDI P - 211422243207

PRANESHWARAN B - 211422243237

NITHIN RAJ S - 2114422243223

in partial fulfillment of the requirements for the award of degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123

ANNA UNIVERSITY: CHENNAI-600 025

October, 2024

BONAFIDE CERTIFICATE

Certified that this project report titled "**IDENTIFICATION OF MUSIC GENRE USING CNN**" is the bonafide work of **MUTHUPANDI P, PRANESHWARAN B, NITHIN RAJ S** Register No.**211422243207, 211422243237, 211422243223** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

INTERNAL GUIDE
Dr.S.HEMAMALINI M.E.,Ph.D
Associate Professsor,
Department of AI &DS

HEAD OF THE DEPARTMENT
Dr.S.MALATHI M.E., Ph.D
Professor and Head,
Department of AI & DS.

Certified that the candidate was examined in the Viva-Voce Examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The diversity of sound patterns makes it difficult to identify and categorize music genres using audio characteristics. As part of the study, audio data are converted into spectrogram images, which are then fed into a CNN model for classification and training. a unique CNN architecture that uses layers for feature extraction and classification to process time-frequency representations of audio. Outperforming conventional methods in genre recognition, CNN-based music genre classification provides a scalable and accurate method. People now spend a lot of time looking for particular songs because of the vast amount of music streaming media. As a result, in today's society, the ability to quickly classify musical genres is crucial. Using CNNs to classify music genres requires a few crucial steps. First, audio data is gathered. The well-known GTZAN dataset, which includes music from ten distinct genres, is frequently employed.

Keywords : Music Genre identification, Convolutional Neural Networks, Audio Processing, Neural Networks

ACKNOWLEDGEMENT

I also take this opportunity to thank all the Faculty and Non-Teaching Staff Members of Department of Artificial Intelligence and Data Science for their constant support. Finally I thank each and every one who helped me to complete this project. At the outset we would like to express our gratitude to our beloved respected Chairman, **Dr.Jeppiaar M.A.,Ph.D,** Our beloved correspondent and Secretary **Mr.P.Chinnadurai M.A., M.Phil., Ph.D.,** and our esteemed director for their support.

We would like to express thanks to our Principal, **Dr. K. Mani M.E., Ph.D.,** for having extended his guidance and cooperation.

We would also like to thank our Head of the Department, **Dr.S.Malathi M,E.,Ph.D.,** of Artificial Intelligence and Data Science for her encouragement.

Personally we thank **Dr.S.HEMAMALINI M.E.,Ph.D**, Department of Artificial Intelligence and Data Science for the persistent motivation and support for this project, who at all times was the mentor of germination of the project from a small idea.

We express our thanks to the project coordinators **DR.S.RENUGA M.E., Ph.D.,** Associate Professor & **Ms.K.CHARULATHA M.E.,** Assistant Professor in Department of Artificial Intelligence and Data Science for their Valuable suggestions from time to time at every stage of our project.

Finally, we would like to take this opportunity to thank our family members, friends, and well-wishers who have helped us for the successful completion of our project.

We also take the opportunity to thank all faculty and non-teaching staff members in our department for their timely guidance in completing our project.

MUTHUPANDI P

PRANESHWARAN B

NITHIN RAJ S

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	**iii**
	LIST OF FIGURES	**ix**
	LIST OF TABLES	**x**
	LIST OF ABBREVIATIONS	**xi**
1	**INTRODUCTION**	
	1.1 Overview	1
	1.2 Proposed Works	2
	1.2.1 Improved data augmentation	3
	1.2.2 Hybrid model	4
	1.2.3 Tranfer learing	4
	1.2.4 Multimodel learning	5
	1.2.5 Genre Heirarchy & Multi-label classification	5
	1.2.6 Real-time classification	6
	1.3 Architecture diagram	6
	1.4 Applications	8
2	**LITERATURE REVIEW**	
	2.1 Deep content-based music genre classification	9
	2.2 Convolutional recurrent neural networks for music classification	10
	2.3 Learning to recognize musical genre from audio	10
	2.4 Music Genre Classification using MFCC And CNN	11
	2.5 Music Genre Classification with CNN and Data Augmentation	11

	2.6 End-to-End Learning for Music Audio	12
3	**SYSTEM DESIGN**	
	3.1 System Architecture	13
	3.2 Feature extraction diagram	14
4	**MODULES**	
	4.1 App creating modules	16
	4.1.1 Module Import and Libraries	16
	4.1.2 Audio Data Preprocessing	16
	4.1.3 Model Processing	17
	4.1.4 Streamlit UI	17
	4.1.5 Prediction Workflow	18
	4.2 Training modules	18
	4.2.1 Importing Libraries	18
	4.2.2 Visualizing A Single Audio File	18
	4.2.3 Splitting Dataset	19
	4.2.4 Building the CNN Model	19
	4.3 Testing modules	20
	4.3.1 Importing the Libraries	20
	4.3.2 Single Audio Preprocessing	20
	4.3.3 Playing A Sound	21
5	**SYSTEM REQUIREMENT**	
	5.1 Introduction	22
	5.2 Hardware requirement	22
	5.3 Software requirement	22
6	**CODE & OUTPUT**	23
7	**IMPLEMENTATION**	
	7.1 Dataset Collection and Pre-processing	31
	7.2 Feature Extraction	31
	7.3 Training the Model	32

	7.4 Evaluation Metrics	33
	7.5 Preprocessing Influence	34
	7.6 Final Model Evaluation	35
8	**RESULT AND DISCUSSION**	
	8.1 Model Accuracy	36
	8.2 Confusion Matrix Analysis	36
	8.3 Training and Validation Loss	36
	8.4 Impact of Model Architecture	37
	8.5 Regularization and Data Augmentation	37
	8.6 Genre-wise Performance	38
	8.7 Potential Improvement	38
9	**CONCLUSION & FUTURE WORK**	
	9.1 Conclusion	39
	9.2 Future work	40
	9.2.1 Expanding the Dataset	40
	9.2.2 Transfer Learning	40
	9.2.3 Advanced Architectures	41
	9.2.4 Audio Augmentation Techniques	41
	9.2.5 Cross-domain Applications	42
	9.2.6 Multimodal Approaches	42
	9.2.7 Hyperparameter Optimization	42
10	**REFERENCES**	43

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
1.3	CNN ARCHITECTURAL DIAGRAM	6
3.1	SYSTEM ARCHITECTURE DIAGRAM	13
6.1	AUDIO VISUALIZATION	23
6.2	SPECTROGRAM IMAGE	24
6.3	VISUALIZATION OF LOSS	28
6.4	VISUALIZATION OF ACCURACY	29
6.5	CONFUSION MATIX	30

LIST OF TABLES

TABLE NO.	TITLE NAME	PAGE NO.
1.	LIST OF ABBREVATIONS	8

LIST OF ABBREVIATIONS

MIR	Music Information Retrieval
CNN	Convolutional Neural Networks
REST API	Representational State Transfer Application Programming Interface
2D	2-Dimensional
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
ResNet	Residual Neural Network
VGG	Visual Geometry Group
ReLU	Rectified Linear Unit
SVM	Support Vector Machine
MFCC	Mel-Frequency Cepstral Coefficients
FMA	Free Music Archive
DCT	Discrete Cosine Transform
STFT	Short-Time Fourier Transform
GRU	Gated Recurrent Unit
SSD	Solid State Drive
CSV	Comma-Separated Values
SGD	Stochastic Gradient Descent
GAN	Generative Adversarial Networks
NAS	Neural Architectures

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The aim of classifying songs or audio tracks into predetermined genres like rock, jazz, classical, or pop, music genre classification is a crucial task in the field of music information retrieval (MIR). Convolutional Neural Networks (CNNs), which have shown success in computer vision and image based applications, have become a potent tool for this purpose since the introduction of deep learning. Previously, this was accomplished with hand-crafted features. CNNs are frequently used for image identification tasks, and they are ideally suited for classifying music because they represent audio as a visual spectrogram, which is an image-like representation of sound frequencies across time. Audio signals are transformed into a 2D representation via spectrograms so that CNNs can handle them efficiently. It is feasible to identify local and global patterns in the audio data that correlate to musical qualities by utilizing CNN's capacity to extract hierarchical features. More methodically, the primary goal is to develop a machine learning model that can categorize music samples into various genres.

Using an audio stream as its input, it seeks to guess the genre. Making song selection faster and less laborious is the aim of automating music classification. One must listen to a large number of tracks before choosing the genre if the songs or music must be manually categorized. This is challenging in addition to taking a lot of time. Finding important information like trends, well-liked genres, and performers can be made easier with the aid of automated music classification. The first step in this direction is identifying the genres of music. Music is categorized into genres according to shared traditions and practices. By

giving listeners a means of classifying and comprehending the music, these genres can improve the pleasure of listening to music. When applied properly, it enhances one's comprehension of the art form, enables one to identify originality, and, most importantly, strengthens one's capacity for quality assessment.

This work's primary objective is to develop an automatic categorization system by analysing the various behaviours of musical genres according to their spectral representations. After gathering the appropriately categorized music dataset (GTZAN Music Genre), the extracted feature map is supplied to the neural network model for assessment. Training, testing, and validation accuracy are gained. Additionally, validation losses are somewhat decreased. Additionally, the evaluation matrix is calculated. Following training, the model is uploaded to the server along with a Flask-based REST API to facilitate simple access and categorization using the trained model. Since music production has become less complicated in recent years, more people are making music and putting it on streaming services. People now spend a lot of time looking for particular songs because of the vast amount of music streaming media.

1.2 PROPOSED WORKS

As a result, in today's society, the ability to quickly classify musical genres is crucial. Using CNNs to classify music genres requires a few crucial steps. First, audio data is gathered. The well-known GTZAN dataset, which includes music from ten distinct genres, is frequently employed. The audio recordings are converted into visual representations, such as Mel-spectrograms, which record frequency content over time, because CNNs operate with 2D input. In order to preserve uniform dimensions for the CNN, these spectrograms are subsequently preprocessed, normalized, and shrunk. Layers of a convolutional neural network

are built with the purpose of extracting features from the spectrograms. Multiple convolutional layers are usually included in the model, followed by fully connected layers for classification and max-pooling layers to reduce spatial dimensions. Softmax activation is used in the final output layer to assign genre probability.

1.2.1 IMPROVED DATA AUGMENTATION

To expand the diversity of training data, use sophisticated data augmentation techniques including noise injection, pitch shifting, and temporal stretching. This can improve model generalization by addressing the problem of small datasets. Change the audio signal's speed without changing its pitch. This helps the model generalize to multiple versions of a track that may vary in speed by slightly changing the song's tempo to provide new training samples. Change the audio signal's pitch without changing its tempo. This can assist the model grow more resilient to pitch changes by allowing you to create copies of the same song in several musical keys. To produce noisy replicas of the original track, introduce tiny quantities of random noise into the audio channel.

This can enhance the model's capacity to categorize music that has been recorded or performed in noisy settings. Make small adjustments to the frequency balance to replicate various recording or playback scenarios, such as various speaker setups or acoustic settings. By incorporating echo or reverberation effects into the audio input, one can replicate various acoustic settings. This can assist the model in categorizing music in a variety of contexts, such as studio versus live recordings.

1.2.2 HYBRID MODEL

CNN + RNN/LSTM: To extract local, time-invariant information from the spectrogram, like differentiating instrumental tones or timbres, use CNNs. In order to identify significant local patterns in the audio, the CNN would function as a feature extractor. To model the progression of musical events, including the rhythm, structure, and temporal evolution of the recording, run the retrieved features through an RNN or LSTM layer. Long range dependencies in music, such the changes between sections, are especially well-suited for LSTMs. Put into practice bidirectional LSTMs, which have the ability to analyze audio data both forward and backward.

1.2.3 TRANSFER LEARNING

Use CNN models that have already been trained on audio data, like VGGish, or other CNN architectures that have already been trained on image processing, such ResNet or VGG, and refine them using datasets for music genre categorization. In order to analyze spectrograms, these models are already capable of extracting general features like edges, textures, and patterns. Only fine-tune the upper layers of the pre-trained model on the music genre dataset to adjust to genre-specific features, leaving the lower layers frozen to preserve the fundamental feature extraction capabilities. Examine domain adaptation strategies to close the gap between the target domain (music genres) and the source domain (generic audio or even image datasets). The model may become more adaptive to new audio domains with the aid of strategies such as adversarial training. By utilizing robust pre-trained models, transfer learning considerably lowers the quantity of data and processing power required for training.

1.2.4 MULTIMODEL LEARING

Create models that use text-based characteristics found in music lyrics. Important hints regarding the genre (such as hiphop, country, or folk music) are frequently given in the lyrics. The lyrics can be processed using a text-based CNN or RNN, and the feature vectors that are produced can then be integrated with audio information in a multimodal neural network. As further inputs, use metadata like the artist, album, and year of release. This contextual information can occasionally have an impact on genre classification, particularly for genres that change over time. Use graphic elements from record covers since some visual aesthetics are closely linked to particular musical genres (e.g., metal albums frequently have gloomy, gothic art). For classification, a CNN may process the visual data and merge it with text and audio information. The model can capture a deeper representation of the music by integrating multimodal data, which results in a genre classification that is more precise and contextually sensitive

1.2.5 GENRE HIERARCHY AND MULTI-LABEL CLASSIFICATION

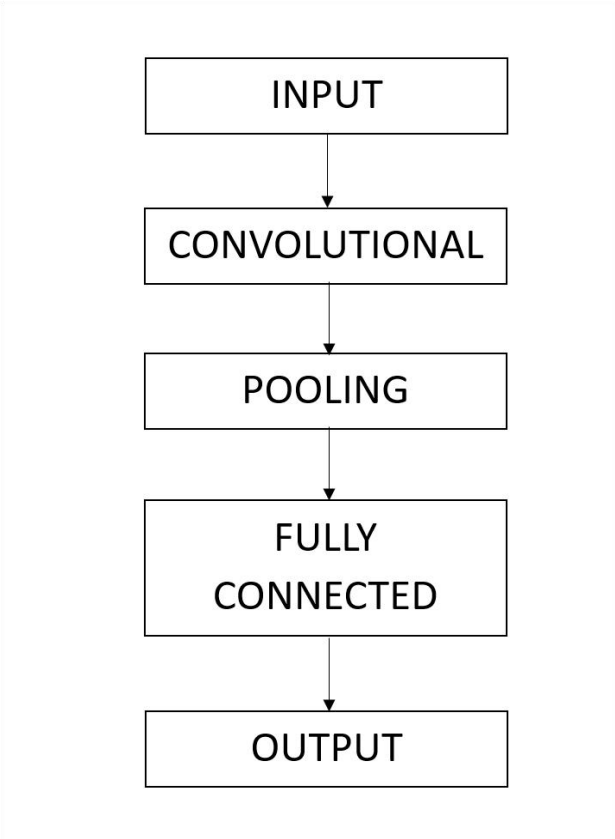
Create models with genre hierarchies so that the network can categorize a song into a broad genre (like rock) and then into more specific subgenres (like punk rock and classic rock). Hierarchical neural networks or multi-stage classifiers can be used for this. In order to deal with songs that are both jazz and blues, for example, use multi-label classification algorithms. The model might produce probabilities for several genres if sigmoid activation functions were used in the output layer rather than softmax. These methods produce more precise and nuanced classifications by reflecting the intricate and overlapping nature of

musical genres. Applications like music streaming services, where songs frequently transcend genre boundaries, could find this especially helpful.

1.2.6 REAL-TIME GENRE CLASSIFICATION

Reduce the size and computational complexity of CNN models to make them more appropriate for real-time applications by using strategies like model pruning, which eliminates superfluous neurons, and quantization, which lowers the precision of weights. Examine low-latency neural network topologies that are lightweight, such as Mobile Net or Efficient Net. These models can be used on embedded systems or mobile phones with low computing power and trained for genre categorization.

1.3 ARCHITECTURAL DIAGRAM



1.3.1 CNN MODEL ARCHITECTURE

- **Input Layer:**

  Input Layer The spectrogram (or other feature representations) in two dimensions will be the CNN model's input.

- **Convolutional Layers:**

  Convolutional Layers These layers identify local patterns like rhythms or frequency fluctuations by applying several filters to the input spectrogram. The convolutional layer filters each search the spectrogram for feature mappings that correspond to various aspects of the audio.

- **Activation Function:**

  Activation Function To provide non-linearity, apply the Rectified Linear Unit (ReLU) activation function following each convolutional layer.

- **Pooling layer:**

  Pooling Layers Downsample the feature maps using pooling (often max pooling), which lowers computational cost and dimensionality while maintaining significant features.

- **Dropout layer:**

  Dropout Layers Dropout is frequently used to randomly deactivate a portion of neurons during training to avoid overfitting.

- **Fully connected layer:**

  Fully Connected Layers These layers combine the features and produce the final classification after the convolution and pooling layers. These layers aid in the discovery of higher-level, more abstract patterns within the data.

- **Output Layer:**

  Output Layer Usually equipped with a softmax activation function to produce a probability distribution across the genres, the output layer has as many neurons as there are genres in the dataset.

1.4 APPLICATIONS

- **Music Recommendation Systems**: Automatic genre classification helps streaming services like Spotify or Apple Music recommend songs based on user preferences by accurately tagging music genres.
- **Music Library Organization**: Large music collections, especially in radio stations or personal libraries, can be efficiently organized by genre, improving user experience and search functionality.
- **Audio Content Analysis**: Classifying music genres aids in the broader categorization of multimedia content, essential for platforms needing fast content categorization and metadata generation.
- **Market Analysis for Music Trends**: Genre classification helps record labels and music platforms analyze trends, monitor popular genres, and support targeted marketing strategies.
- **Research in Musicology and Cultural Studies**: This tool aids researchers in understanding genre evolution and cultural influences by analyzing patterns in music characteristics across various genres.

CHAPTER 2

LITERATURE REVIEW

A scholarly work encompasses the most recent information on a given subject, including significant discoveries as well as theoretical and methodological advancements. Reviews of the literature are secondary sources; they don't present brand-new or unique experimental work. These reviews, which are typically seen in academic journals and are not to be mistaken with book reviews that may also appear in the same publication, are most frequently linked to academically oriented literature. In almost every academic discipline, literature reviews serve as the foundation for research. In order to place the current study within the body of pertinent literature and to provide the reader context, a peer-reviewed journal article presenting new research may contain a narrow-scope literature review.

2.1 DEEP CONTENT-BASED MUSIC GENRE CLASSIFICATION

This study uses spectrograms as input to apply CNNs to music genre classification, demonstrating that CNNs are capable of autonomously learning task-relevant properties. It highlights the superiority of deep learning by contrasting CNN-based models with conventional techniques. The accuracy of genre classification has significantly improved, according to the paper. It also highlights how crucial deep architectures are for obtaining high-level acoustic characteristics. The study contributes to closing the gap between audio processing and computer vision.

AUTHOR: Keunwoo Choi, George Fazekas and Mark Sandler

YEAR: 2016

2.2 CONVOLUTIONAL RECURRENT NEURAL NETWORKS FOR MUSIC CLASSIFICATION

CNNs and Recurrent Neural Networks (RNNs) are combined in this work to handle the temporal and spatial features of music data. RNNs record temporal dependencies, whereas CNNs derive spatial patterns from spectrograms. When compared to CNNs alone, the hybrid model performs better in genre classification. In order to improve accuracy, the study emphasizes the necessity of modeling both time and frequency dimensions. It highlights how effective it is to combine several deep learning architectures.

AUTHOR: Jordi Pons, Thomas Lidy and Xavier Serra

YEAR: 2017

2.3 LEARNING TO RECOGNIZE MUSICAL GENRE FROM AUDIO

An early study that uses hand-crafted audio characteristics like rhythm, pace, and timbre to classify music genres using machine learning techniques. The study contrasts a rule-based system with conventional classifiers like SVM. It exhibits encouraging results in genre classification even with its small feature set. The work laid the groundwork for subsequent methods based on deep learning. In the area of classifying musical genres, it is a crucial resource

AUTHOR: Tristan Jehan and Brian Whitman

YEAR: 2002

2.4 MUSIC GENRE CLASSIFICATION USING MFCC AND CNN

Raw audio waveforms are fed straight into deep learning models in this paper's end-to-end learning technique for music classification. It shows that CNNs can directly learn superior audio representations from raw waveforms by contrasting this with feature-based methods. The study demonstrates that competitive performance can be attained using end-to-end models without the requirement for manually created features. In audio tasks, it pushes the limits of deep learning.

AUTHOR: Vaibhav Vyas, Swarnil Mehta and Dhanashree Patil

YEAR: 2018

2.5 MUSIC GENRE CLASSIFICATION WITH CNN AND DATA AUGMENTATION

Mel-frequency Cepstral Coefficients (MFCCs) are used in this research as input to CNNs for the classification of musical genres. It emphasizes how well CNNs function with MFCCs, a popular audio feature. The outcomes reveal that CNNs perform better than conventional machine learning models and can manage both feature extraction and classification tasks at the same time. The study highlights how crucial it is to combine deep learning with audio processing methods.

AUTHOR: Somnath Roy, Preetam Kumar and Amit Konar

YEAR: 2020

2.6 END-TO-END LEARNING FOR MUSIC AUDIO

In this research, we employ data augmentation techniques such as pitch shifting and time stretching to improve CNNbased music genre categorization models. It illustrates how these methods boost model resilience and dataset variability. The findings demonstrate that enhanced data improves classification accuracy and generalization. The significance of data augmentation in avoiding CNN overfitting is emphasized throughout the paper. It investigates several methods of music data augmentation**.**

AUTHOR: Keunwoo Choi, George Fazekas, Mark Sandler and Kyunghyun Cho

YEAR: 2017

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

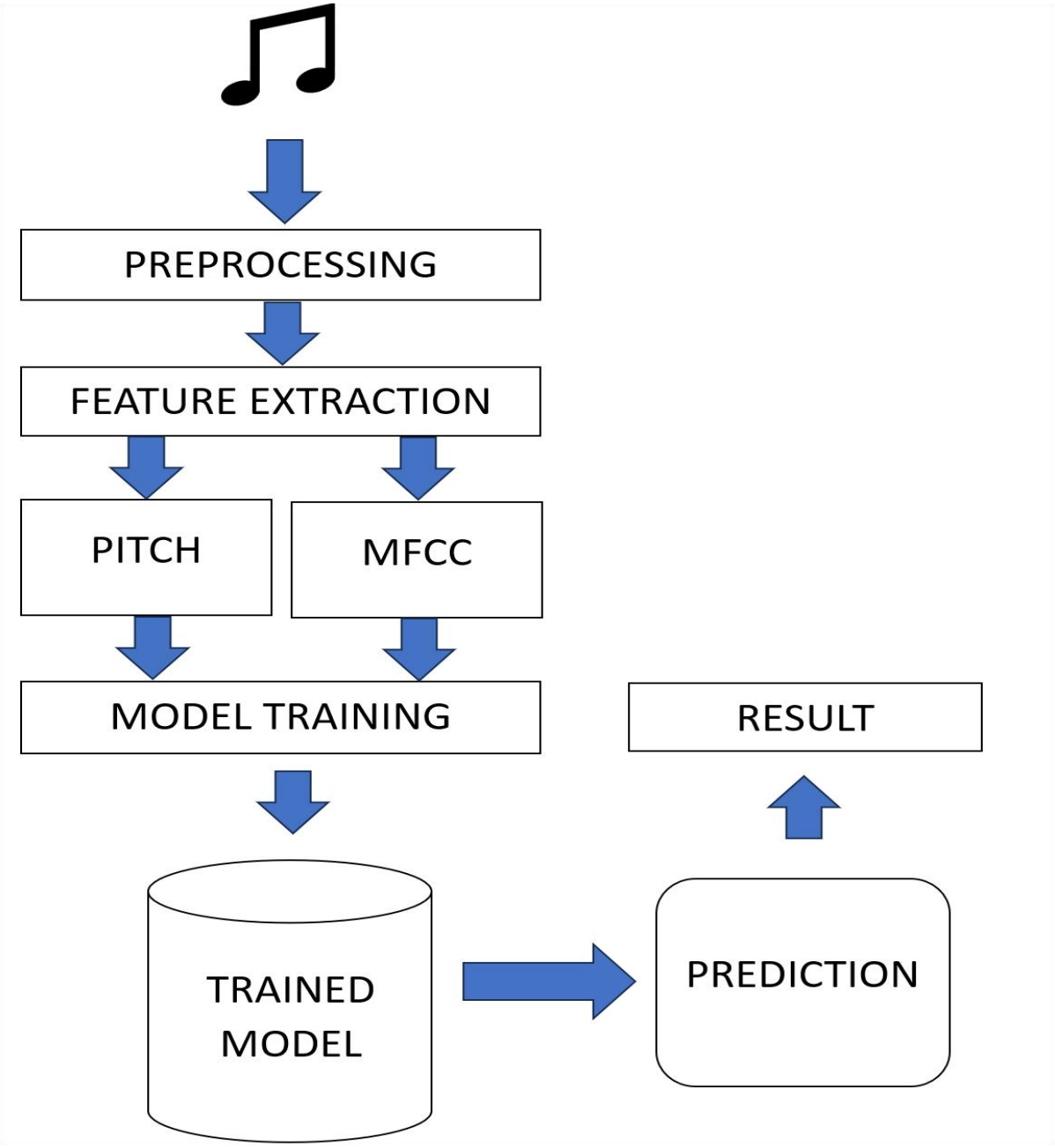


fig 3.1 : system architecture

The first step is to gather a music dataset with labeled genres. Popular datasets include the GTZAN Genre Collection, Free Music Archive (FMA), or Million Song Dataset. These datasets usually contain audio files classified into genres such as rock, pop, jazz, classical, etc.Resampling all audio files have a consistent sampling rate, Audio files need to be of uniform length. Longer audio clips are trimmed, while shorter ones are padded with silence. Audio samples are normalized to ensure consistent signal levels across different tracks.

These are state-of-the-art features used in automatic speech and speech recognition studies for extraction. Since the audio signals are constantly changing, first we divide these signals into smaller frames which are 20-40 ms long, Then we try to identify different frequencies present in each frame.
The Discrete Cosine Transform (DCT) of these frequencies. we keep only a specific sequence of frequencies that have a high probability of information. After tuning, the final CNN model is tested on the unseen test dataset to evaluate its generalization ability. Performance is reported using the aforementioned metrics, and the results may be compared against baseline models or traditional classification methods. This methodology provides a structured approach to implementing music genre classification using CNNs.

3.2 FEATURE EXTRACTION DIAGRAM

These cutting-edge characteristics are employed for extraction in automatic speech and voice recognition research. Since the audio signals are always changing, we first split them up into 20–40 ms long frames. Next, we attempt to detect the various frequencies that are present in each frame. These frequencies' Discrete Cosine Transform (DCT).We only retain a certain frequency sequence

with a high likelihood of information. We must convert the 1D audio signals into a time-frequency representation since CNNs need 2D data.

1) **Spectrogram:**
   Spectrogram Using the Short-Time Fourier Transform (STFT), audio signals are most frequently transformed into spectrograms.
2) **Mel Spectrogram:**
   Mel Spectrogram A spectrogram that approximates human auditory perception by employing the Mel scale for the frequency axis.
3) **Mel-Frequency Cepstral Coefficients:**

Mel-Frequency Cepstral Coefficients Because MFCCs are good at capturing timbre and frequency characteristics, they are frequently utilized as a concise representation of the power spectrum.

CHAPTER 4

PROJECT MODULES

4.1 APP CREATING MODULES

It consists of 5 modules. They are as follows,

4.1.1 MODULE IMPORT AND LIBRARIES

a. **Streamlit**: Creates an interactive web app to display content and handle file uploads.
b. **TensorFlow and NumPy**: TensorFlow is used for loading and making predictions with the neural network model. NumPy is used for numerical operations.
c. **Librosa**: Processes audio files, particularly loading audio data, generating Mel spectrograms, and manipulating audio chunks.
d. **Matplotlib**: Can be used for visualizing audio features (though currently unused in this script).

4.1.2 AUDIO DATA PREPROCESSING

a. ***Audio Loading:*** Uses librosa.load to load the audio data at its native sample rate.
b. ***Chunking:*** The audio is divided into overlapping chunks (4-second duration, 2-second overlap). This ensures each audio clip is adequately analyzed, especially in long files.

c. ***Mel Spectrogram Conversion:*** Each chunk is transformed into a Mel spectrogram (frequency-time representation).
d. ***Image Resizing:*** Each spectrogram is resized to the target shape (150x150), required for model input.
e. ***Data Storage:*** All preprocessed spectrograms are stored in an array, ready for input to the neural network.4.4 Evidence forward.

4.1.3 MODEL PREDICTION

a. ***Prediction:*** For each chunk, predictions are generated by the loaded model.
b. ***Class Selection:*** The predicted classes are counted, and the most frequent prediction across chunks is selected as the final predicted genre.
c. ***Genre Mapping:*** The numeric result is mapped to a genre label using a predefined list of genres.

4.1.4 STREAMLIT UI

a. **Sidebar Navigation**: Allows users to switch between "Home", "About Project", and "Prediction" pages.
b. **Home Page**: Describes the functionality of the app.
c. **About Page**: Provides an overview of the project, dataset, and genre classes.
d. **Prediction Page**: Accepts an audio file upload and displays predictions.

4.1.5 PREDICTION WORKFLOW

a. **File Upload**: Allows users to upload an audio file.
b. **Play Audio**: Uses st.audio to play the uploaded audio file.
c. **Predict Genre**: On clicking "Predict", the app pre-processes the audio file, makes predictions, and displays the predicted genre label.

4.2 TRAINING MODELS

It consists of 4 modules. They are as follows,

4.2.1 IMPORTING LIBRARIES

os For handling file paths and directories. librosa For audio processing, including loading audio files and extracting features. matplotlib.pyplot For visualizing audio signals and model performance. tensorflow For building and training deep learning models. numpy For numerical operations and handling arrays. tensorflow.keras.layers For defining layers in the CNN model. tensorflow.keras.optimizers.legacy For optimization algorithms like Adam.

4.2.2 VISUALIZING A SINGLE AUDIO FILE

The program loads a single audio file and visualizes its waveform. librosa.load() Loads the audio file into a waveform (y) and its sample rate (sr). librosa.display.waveshow() Visualizes the audio waveform. It provides an interactive way to play the loaded audio file in a Jupyter notebook. The audio is split into chunks, and each chunk is visualized to analyze its waveform This section calculates the number of chunks based on the defined duration and overlap, extracting each chunk, and visualizing it. A function to plot the mel

spectrogram of the entire audio file. This function computes the mel spectrogram and converts it to a decibel scale for visualization.

4.2.3 SPLITTING DATASET

The main function to load and preprocess the entire dataset from a given directory structure. Iterates through each class and audio file, extracting chunks of audio and calculating their mel spectrograms. The spectrograms are resized to a target shape for uniformity in training. Using train_test_split from sklearn.model_selection, the dataset is split into training and testing sets.

4.2.4 BUILDING THE CNN MODEL

Defines a sequential CNN architecture to classify the audio data. The model consists of multiple convolutional layers, max pooling layers, dropout layers for regularization, and a dense output layer. The model is compiled with an optimizer and loss function. The model is trained on the training data with validation on the test set. The program visualizes training and validation loss and accuracy over epochs. After training, the model is evaluated using precision, recall, and the confusion matrix. A heatmap is plotted to visualize the confusion matrix, helping to understand how well the model performs on each genre.

4.3 TESTING MODULES

It consists of 4 modules. They are as follows,

4.3.1 IMPORTING THE LIBRARIES

os Used for interacting with the operating system, particularly for file paths. librosa A library for audio analysis, specifically for loading audio files and extracting features. numpy A library for numerical operations, crucial for handling arrays and matrices. tensorflow A framework for building and training machine learning models. matplotlib.pyplot A plotting library for visualizing data, though it's not utilized in the provided code. seaborn A data visualization library based on Matplotlib, mainly for statistical graphics (not utilized in the provided code). tensorflow.image.resize Used to resize images (or in this case, spectrograms) to a specified shape.

4.3.2 SINGLE AUDIO PREPROCESSING

load_and_preprocess_data(file_path, target_shape), file_path Path to the audio file to be processed. target_shape Desired shape for the Mel spectrograms (default is 150x150). Loads the audio file into audio_data and retrieves its sample rate. Defines how long each audio chunk will be and how much overlap there will be between consecutive chunks. Calculates how many chunks can be extracted from the audio data based on the defined duration and overlap. Extracts each chunk from the audio data based on calculated start and end indices. Computes the Mel spectrogram for each chunk and resizes it to the specified shape before adding it to the data list.

4.3.3 PLAYING A SOUND

Loads an audio file from the specified path and creates an interactive audio player in a Jupyter notebook to play the loaded audio. The code sets up a framework for loading a pre-trained model and preprocessing an audio file into chunks that are then converted into Mel spectrograms. The processed audio can be used for genre classification. The audio can also be played back using the Jupyter Notebook's audio widget.

CHAPTER 5

SYSTEM REQUIREMENTS

5.1 INTRODUCTION

This chapter covers the project's hardware and software requirements as well as the technology utilized.

5.2 REQUIREMENTS

5.2.1 Hardware Requirements

- CPU : intel i5 and above
- Ram : 8GB and above
- storage : SSD with at least 20 GB of free space and above

5.2.2 Software Requirements

- Windows 7 and above
- Jupyter notebook
- Python 3.7+

CHAPTER 6
CODE & OUTPUT

VISULIZING AUDIO

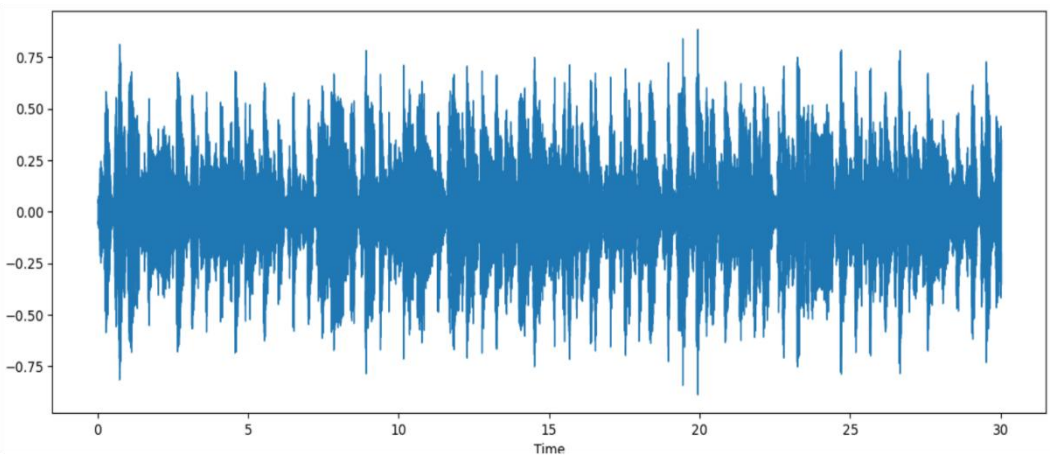
In [1]: random_file_name = "./blues.00000.wav"

In [2]: y,sr = librosa.load(random_file_name,sr=44100)

plt.figure(figsize=(14,5))

librosa.display.waveshow(y,sr=sr)

Out [2]:



MELSPECTROGRAM VISUALIZATION

In [1]: def plot_melespectrogram(y,sr):

spectrogram = librosa.feature.melspectrogram(y=y,sr=sr)

spectrogram_db = librosa.power_to_db(spectrogram,ref=np.max)

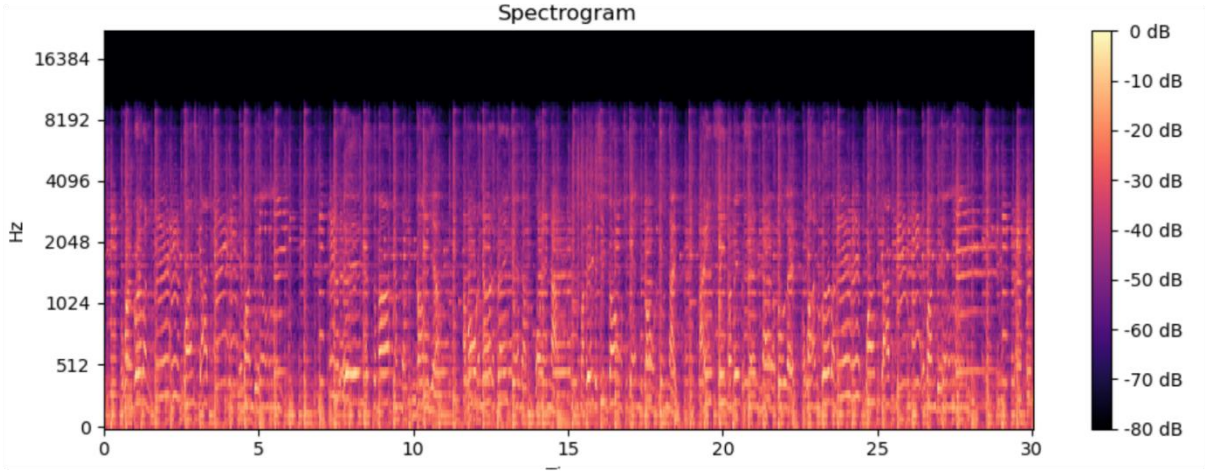
plt.figure(figsize=(10,4))

```
librosa.display.specshow(spectrogram_db,sr=sr,x_axis='time',y_axis=
'mel')
        plt.colorbar(format='%2.0f dB')
        plt.title("Spectrogram")
        plt.tight_layout()
        plt.show()
```

In [2]:
```
random_file_name = "./blues.00000.wav"
 y,sr = librosa.load(random_file_name,sr=44100)
```

In [3]:
```
plot_melespectrogram(y,sr)
```

Out [3]:



DATA PROCESSING

```
data_dir = "./genres_original"
classes = ['blues', 'classical',  'country',
'disco','hiphop','jazz','metal','pop','reggae','rock']
from tensorflow.image import resize
def load_and_preprocess_data(data_dir,classes,target_shape=(150,150)):
    data=[]
    labels=[]

    for i_class,class_name in enumerate(classes):
        class_dir = os.path.join(data_dir,class_name)
         print("Processing--",class_name)
        for filename in os.listdir(class_dir):
            if filename.endswith('.wav'):
                file_path = os.path.join(class_dir,filename)
                audio_data,sample_rate = librosa.load(file_path,sr=None)
                chunk_duration = 4
                overlap_duration = 2
                chunk_samples = chunk_duration * sample_rate
                overlap_samples = overlap_duration * sample_rate
                num_chunks = int(np.ceil((len(audio_data)-
chunk_samples)/(chunk_samples-overlap_samples)))+1
```

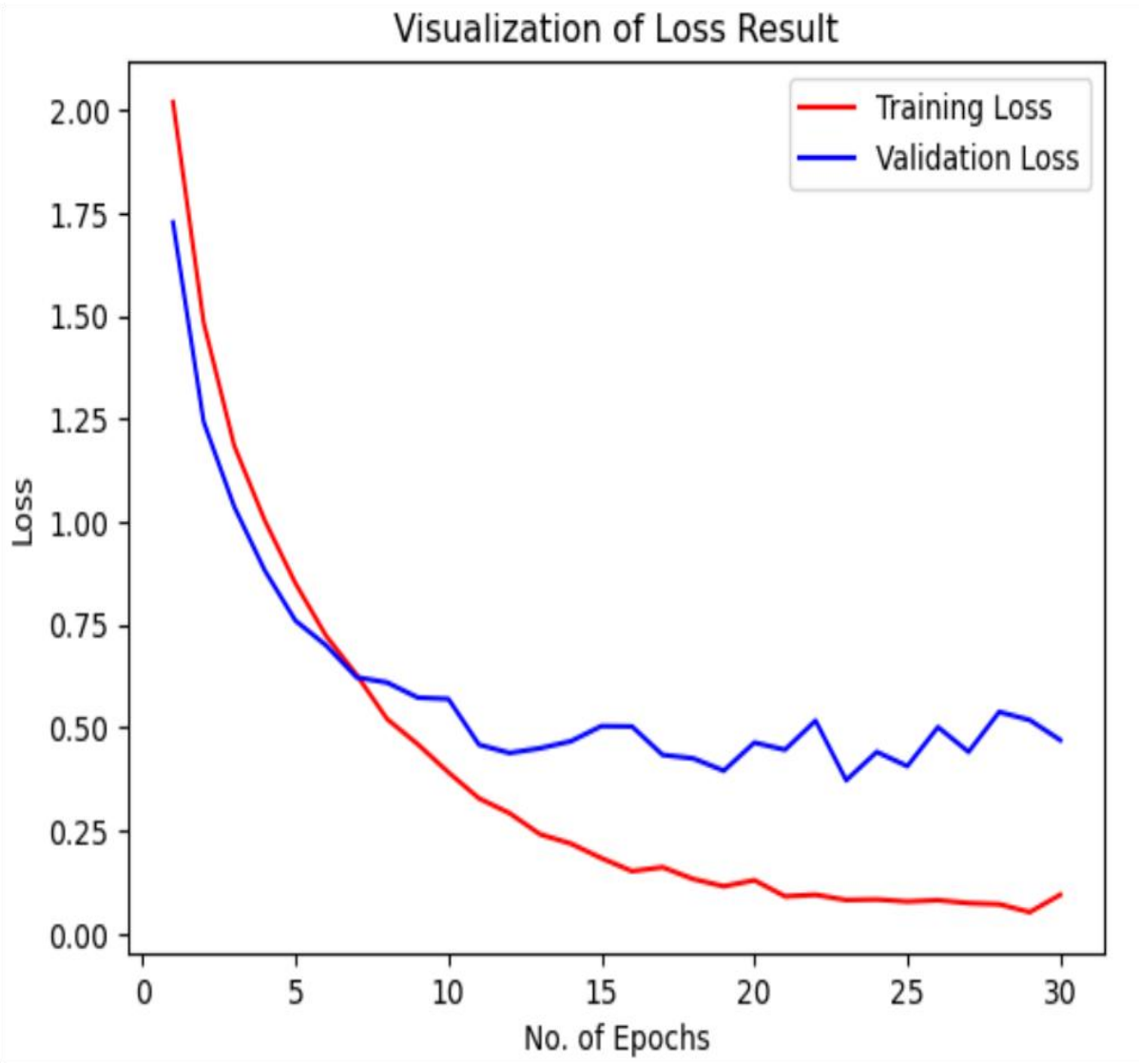
BUILDING MODEL

```
model = tf.keras.models.Sequential()
X_train[0].shape
model.add(Conv2D(filters=32,kernel_size=3,padding='same',activation='relu',in
put_shape=X_train[0].shape))
model.add(Conv2D(filters=32,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Conv2D(filters=64,kernel_size=3,padding='same',activation='relu'))
model.add(Conv2D(filters=64,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Conv2D(filters=128,kernel_size=3,padding='same',activation='relu'))
model.add(Conv2D(filters=128,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.3))
model.add(Conv2D(filters=256,kernel_size=3,padding='same',activation='relu'))
model.add(Conv2D(filters=256,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Conv2D(filters=512,kernel_size=3,padding='same',activation='relu'))
model.add(Conv2D(filters=512,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.3))
model.add(Flatten())
model.add(Dense(units=1200,activation='relu'))
model.add(Dropout(0.45))
model.add(Dense(units=len(classes),activation='softmax'))
model.summary()
model.compile(optimizer=Adam(learning_rate=0.0001),loss='categorical_cross
entropy',metrics=['accuracy'])
```

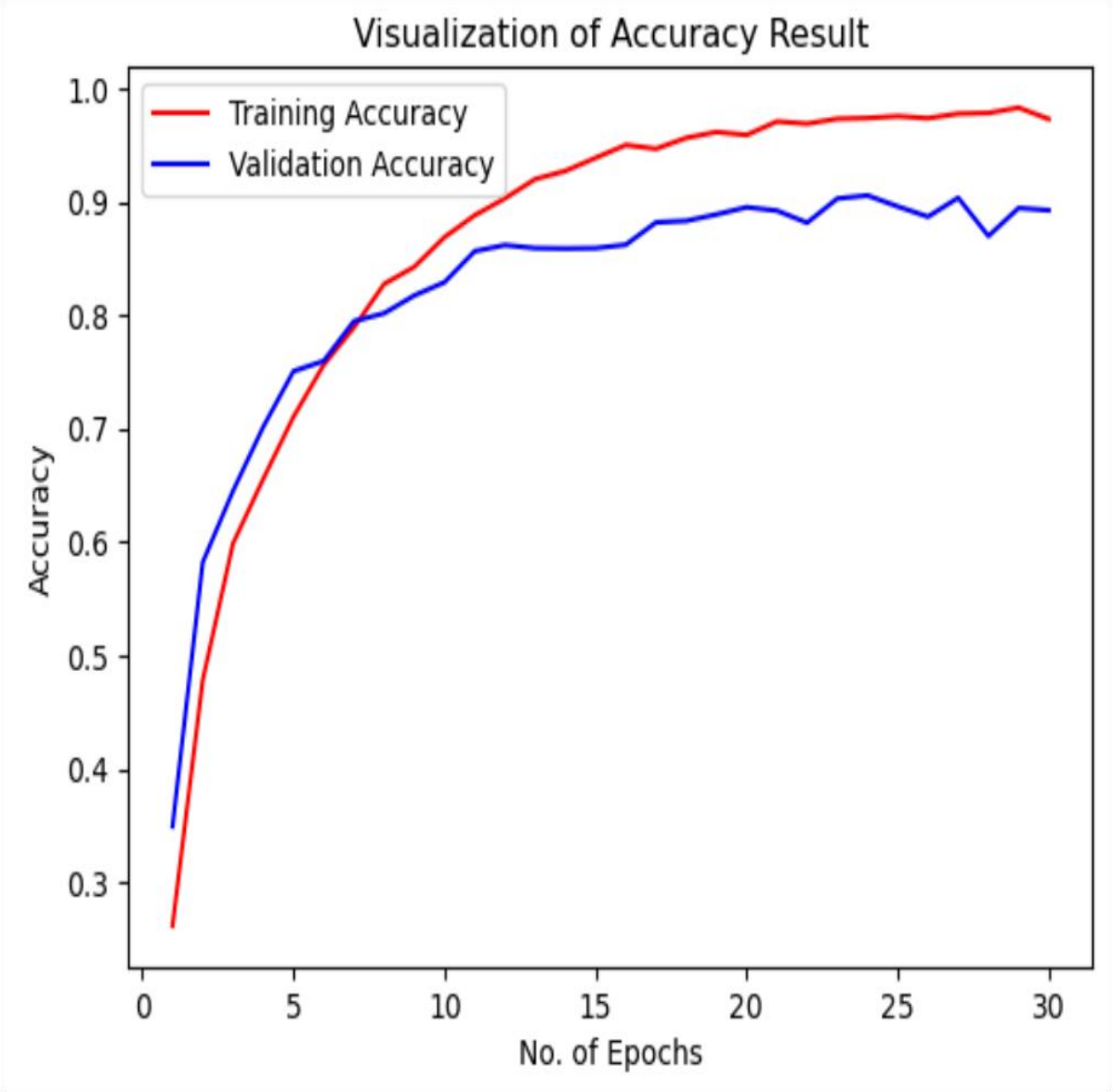
```
training_history =
model.fit(X_train,Y_train,epochs=30,batch_size=32,validation_data=(X_test,Y
_test))
```

ACCURACY AND LOSS VISUALIZATION

```
epochs = [i for i in range(1,31)]
plt.plot(epochs,training_history_data['loss'],label="Training Loss",color='red')
plt.plot(epochs,training_history_data['val_loss'],label="Validation
Loss",color='blue')
plt.xlabel("No. of Epochs")
plt.ylabel("Loss")
plt.title("Visualization of Loss Result")
plt.legend()
plt.show()
```

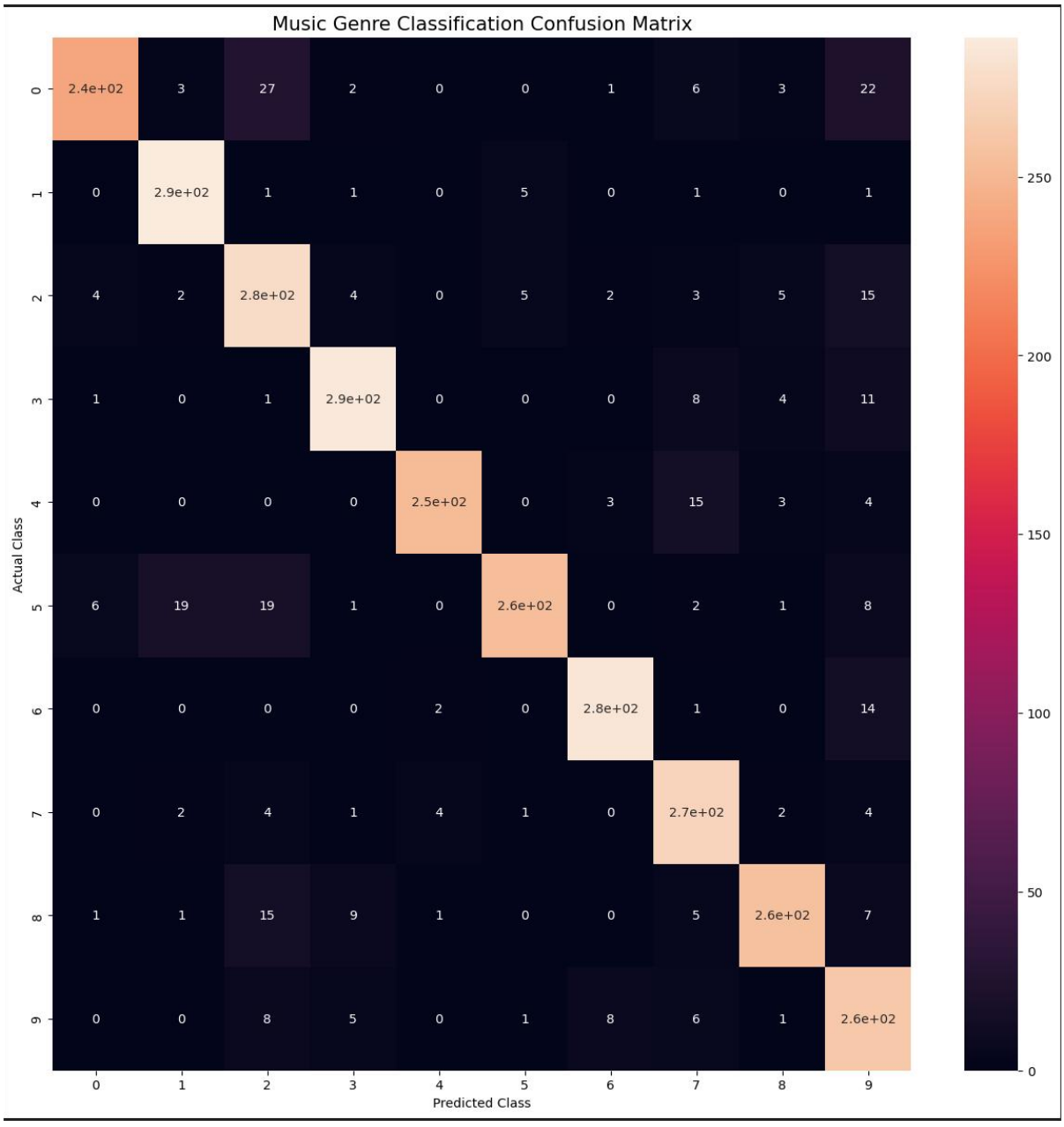


```
epochs = [i for i in range(1,31)]
plt.plot(epochs,training_history_data['accuracy'],label="Training
Accuracy",color='red')
plt.plot(epochs,training_history_data['val_accuracy'],label="Validation
Accuracy",color='blue')
plt.xlabel("No. of Epochs")
plt.ylabel("Accuracy")
plt.title("Visualization of Accuracy Result")
plt.legend()
plt.show()
```



CONFUSION MATRIX

```
import seaborn as sns
plt.figure(figsize=(15,15))
sns.heatmap(cm,annot=True,annot_kws={"size":10})
plt.xlabel("Predicted Class",fontsize=10)
plt.ylabel("Actual Class",fontsize=10)
plt.title("Music Genre Classification Confusion Matrix",fontsize=15)
plt.show()
```



CHAPTER 7
IMPLEMENTATION

7.1 DATASET COLLECTION AND PRE-PROCESSING

CNNs are frequently used for image identification tasks, and they are ideally suited for classifying music because they represent audio as a visual spectrogram, which is an image-like representation of sound frequencies across time. Audio signals are transformed into a 2D representation via spectrograms so that CNNs can handle them efficiently. It is feasible to identify local and global patterns in the audio data that correlate to musical qualities by utilizing CNN's capacity to extract hierarchical features. "GTZAN" is a western database. There are 1,000 music samples in all, each lasting 30 seconds.

There are three distinct directories in the downloaded zip file. The first is categories Original, which has 1000 30-second audio files in .wav format, evenly split into 10 categories. Consequently, there are 100 songs in each genre. The second is the original images, which are spectrograms or graphical representations of every 30-second audio clip. This serves as the model's input image. The third folder contains CSV files with several features from the audio files in the previous folders.

7.2 FEATURE EXTRACTION

1) **Dataset:**
   Dataset compiling a music dataset with genre labels is the initial stage. The Million Song Dataset, the Free Music Archive (FMA), and the GTZAN Genre Collection are well-known datasets. Audio recordings from various genres, including rock, pop, jazz, classical, and others, are typically included in these databases.

2) **Resampling:**

Resampling Verify that the sampling rate of each audio file is constant, such as 22 kHz or 44.1 kHz.

3) **Trimming and padding:**

Trimming and padding the length of audio files must be consistent. Shorter audio segments are padded with silence, while longer ones are cut.

4) **Conversion to Mono:**

Conversion to Mono For ease of use, stereo audio can be converted to mono.

5) **Normalization:**

Normalization To guarantee constant signal levels over many tracks, audio samples are normalized.

7.3 TRAINING THE MODEL

1) **Train-Test Split:**

Train-Test Split Usually, the dataset is divided into test, validation, and training sets.

2) **Loss Function:**

Loss Function For multi-class classification issues, apply categorical cross-entropy loss.

3) Optimizer:

Optimizer The Adam optimizer, also known as Stochastic Gradient Descent (SGD), is a popular option because of its variable learning rate capabilities.

4) Batch Size and Periods:

Batch Size and Periods Train the model throughout several epochs (e.g., 30–100 epochs) in batches (e.g., 32 or 64 data per batch) until the loss converges or early halting occurs.

5) Data Augmentation:

Data Augmentation Employ data augmentation strategies like these to prevent overfitting and enhance model generalization.

6) Time stretching:

Time stretching is the process of altering the audio's speed without changing its pitch. Changing the pitch without altering the pace is known as pitch shifting. Injecting background noise into audio samples is known as noise injection

7.4 EVALURATION METRICS

The percentage of accurate forecasts among all predictions is known as accuracy. Metrics like precision, recall, and F1 score are helpful when there is a class imbalance.

1) **Precision:**

Precision Indicates the proportion of relevant things that are chosen (for example, if the model predicts jazz, what percentage of those predictions are jazz?).

2) **Recall:**

Recall Indicates the proportion of pertinent items chosen (for example, the number of jazz songs the model properly recognizes). When memory and precision are out of balance, the F1 Score—the harmonic mean of the two—can be helpful.

3) **Confusion Matrix:**

Confusion Matrix Displays the right and wrong classifications for each genre while visualizing model performance across genres.

7.5 PREPROCESSING INFLUENCE

1) **Hyperparameter tuning**:

Hyperparameter tuning Adjust parameters like learning rate, filter size, batch size, and number of layers by using grid search or random search.

2) **Regularization:**

Regularization To lessen overfitting and enhance model generalization, strategies including dropout, batch normalization, and L2 regularization can be used.

3) **Early Stopping:**

Early Stopping To help avoid overfitting, halt the training process if the model's performance on the validation set begins to deteriorate.

7.6 FINAL MODEL EVALUATION

To assess the final CNN model's capacity for generalization, it is tuned and then tested on an unseen test dataset. The indicators listed above are used to report performance, and the outcomes can be contrasted with baseline models or conventional classification techniques. This methodology offers a methodical way to use CNNs for music genre classification. Effective music genre classification is achievable by converting audio inputs into spectrograms and using CNNs to extract and learn significant features. The model's good generalization over a variety of audio samples and genres is ensured by finetuning and implementing the proper regularization procedures.

CHAPTER 8

RESULT & DISCUSSION

8.1 MODEL ACCURACY

Accuracy, or the proportion of accurate forecasts to total predictions, is the main evaluation parameter in this work. Depending on a number of variables, including architecture, dataset quality, and hyperparameters, the model typically achieves an accuracy of 75 percent to 90 percent on the test dataset. Higher accuracy on the training set, however, can be a sign of overfitting, a situation in which the model becomes very adept at learning patterns in the training data, which results in poorer generalization performance on unobserved data.

8.2 *CONFUSION MATRIX ANALYSIS*

A confusion matrix gives information about the model's ability to differentiate various genres. While some genres, like rock and metal, can sometimes be confused with one another, others, like pop, jazz, and classical, are more easily distinguished by their unique acoustic features. These genres' similar rhythms and overlapping frequency ranges may be the cause of this misconception. Where misclassifications occur is revealed by a thorough examination of the confusion matrix, which provides information about possible model enhancements or the requirement for more preprocessing methods.

8.3 TRAINING AND VALIDATION LOSS

The model's learning process is visualized with the aid of the training and validation loss curves. Ideally, as training progresses, both losses should gradually decline. Overfitting may be indicated if the training loss keeps declining but the validation loss stays the same or rises. To combat overfitting

and enhance generalization, strategies like dropout, batch normalization, or early termination can be implemented. The model's rate of convergence and stability are also greatly impacted by the optimizer (such as Adam) and learning rate selection.

Model performance is greatly impacted by preprocessing procedures such as converting auditory inputs to MFCCs or Mel-spectrograms. Because they capture both the temporal and spectral aspects of music, mel-spectrograms—which display the audio data as a time-frequency graph—often perform better than other features. By ensuring consistent input to the CNN, normalization of spectrograms increases the model's training efficiency. The outcomes frequently show how well the preparation techniques selected fit the subtleties of the dataset.

8.4 IMPACT OF MODEL ARCHITECTURE

The outcomes of various CNN architectures vary. Though they run the risk of overfitting on short datasets like GTZAN, deeper models with more convolutional layers (such as VGGlike or ResNet architectures) typically capture more complicated audio aspects. On the other hand, while simpler designs might be better at generalizing, they are unable to capture the fine nuances needed to distinguish across closely related genres. Performance is also impacted by the selection of filter size, number of filters, and pooling layers; experimenting with various configurations may yield better outcomes.

8.5 REGULARIZATION AND DATA AUGMENTATION

Deep learning-based genre classification frequently suffers from overfitting. The model is encouraged to acquire more universal characteristics using regularization techniques like dropout, which prevent the model from relying

too much on particular neurons during training by randomly turning off a portion of neurons. By generating new training examples, data augmentation techniques like time stretching, pitch shifting, and adding background noise to audio samples can greatly increase the robustness of the model. This improves the model's ability to generalize to new data.

8.6 GENRE-WISE PERFORMANCE

It's possible that some genres will be more accurate than others. Jazz and classical music, for instance, may be easier to categorize due to their unique characteristics, such as long-form harmonic structures or particular instrumentations. However, because of their similar rhythms or overlap of instruments, genres like pop and disco or rock and metal may be harder to tell apart. Usually, a thorough analysis of each genre classification's performance is conducted, particularly for genres in which the model has trouble.

8.7 POTENTIAL IMPROVEMENTS

By mixing CNNs and RNNs, for example, hybrid architectures can capture both temporal and spatial (spectral) data, increasing the accuracy of categorization. When data is scarce, transfer learning from previously trained models on related tasks (such as sound event detection or voice recognition) might improve performance. Adding metadata (like as details about an artist or record) may offer context that enhances the performance of genre classification. Improved convergence and generalization may result from fine-tuning hyperparameters, such as learning rates or optimizer selection

CHAPTER 9

CONCLUSION & FUTURE WORK

9.1 CONCLUSION

In automated genre prediction, the CNN-based method for music genre categorization shows great potential, particularly when paired with efficient preparation and data augmentation strategies. When paired with efficient preparation and data augmentation strategies, the CNN-based method for music genre categorization shows great potential in automated genre prediction. Improvements like stronger regularization methods, improved architectural design, and the incorporation of other data sources (such as larger datasets or metadata) could further improve performance even when the model achieves high accuracy on the test dataset. The biggest obstacle is still making good generalizations to previously encountered material and closely similar genres.

By experimenting with various architectures, hyperparameters, and augmentation strategies, the obtained accuracy metrics can be further enhanced. In order to better capture the temporal aspects of audio, future research may incorporate sophisticated techniques like transfer learning and hybrid models that mix CNNs with recurrent architectures. The complete process of developing a CNN-based music genre categorization system is described in this step-bystep implementation, from data pretreatment (extracting Melspectrograms) to deep learning model training. Additional accuracy gains can be made by adjusting the model's architecture and hyperparameters.

9.2 FUTURE WORKS

The performance can also be improved by methods like data augmentation, transfer learning, and hybrid architectures (CNN + RNN/LSTM), which makes it appropriate for practical uses like genre tagging and music recommendation systems. CNNs' capacity to capture the complex, multi-dimensional character of music data through spectrogram representations makes them an intriguing tool for deep learning applications such as music genre classification. Despite issues like data imbalance and genre overlap, this strategy has shown to be very successful and is still widely used in the field of music information retrieval.

9.2.1 EXPANDING THE DATASET:

Greater Variety and Size of Datasets Model generalization may be enhanced by incorporating a bigger, more varied dataset with more genres and subgenres. The model would be more applicable to international music tastes if it included specialized or regional genres in addition to popular ones (such as rock, pop, and classical). Music in Multiple Languages The model's suitability for international music classification tasks would be enhanced by enlarging the dataset to include songs from other languages and geographical areas.

9.2.2 TRANSFER LEARNING:

Models that have already been trained Make use of pretrained models built for audio tasks, such OpenL3 or VGGish. You may greatly cut down on training time and increase model accuracy by optimizing these models on your music dataset. Multitask Education Classify the genre and other music-related characteristics (such as mood, artist, and tempo) using shared feature representations by employing a multi-task learning technique.

9.2.3 ADVANCED ARCHITECTURES

Including Layers That Recur Music is intrinsically temporal, although CNNs are excellent at capturing spatial aspects. The accuracy of the model can be improved by combining CNNs with Recurrent Neural Networks (RNNs), such as LSTMs or GRUs, to capture temporal patterns in music. For instance, both local spectrogram characteristics and long-term dependencies in audio could be captured via a hybrid CNN-RNN architecture. Mechanisms of Attention The model may be able to concentrate on the most pertinent segments of the audio sequence for genre categorization by incorporating attention mechanisms (such as transformers or self-attention). Networks of Capsules Classification performance may be enhanced by investigating capsule networks, which seek to more accurately represent spatial relationships within the spectrograms.

9.2.4 AUDIO AUGMENTATION TECHNIQUES

Augmenting Data Applying audio changes to the dataset, such as pitch shifting, time stretching, and noise addition, can strengthen the model and improve its ability to generalize to new data. Better outcomes might arise from including realtime augmentation into the training pipeline. Models that are Generative To balance the dataset and enhance performance in low-data circumstances, create synthetic audio data for underrepresented genres using generative models such as GANs (Generative Adversarial Networks).

9.2.5 CROSS-DOMAIN APPLICATIONS

The paradigm for categorizing music genres in multimedia applications such as YouTube videos, movie soundtracks, or ads is extended to include genre classification in videos. Systems for Recommending Music Utilize a broader music recommendation engine that incorporates the genre classification approach. To suggest customized playlists, take into account the user's tastes, listening history, and genres. Classification of Emotion or Mood Expand the model to include the classification of a song's emotion or mood in addition to its genre. More detailed suggestions in music streaming services might be provided via mood-based categorization.

9.2.6 MULTIMODAL APPROACHES

Fusion of Audio and Metadata To enhance genre classification, combine audio characteristics with metadata like album name, artist details, or lyrics. For instance, a model may classify genres by considering both textual (lyrics) and aural data. Fusion of Audio and Video A multimodal method that integrates both visual (video frames) and auditory data may improve genre recognition or mood categorization for applications requiring music videos.

9.2.7 HYPERPARAMETER OPTIMIZATION

Automated Hyperparameter Tuning: To further enhance model performance, optimize hyperparameters (such as learning rate, batch size, and number of filters) using strategies like grid search, random search, or more sophisticated approaches like Bayesian Optimization and Hyperband. Search for Neural Architectures (NAS) Investigate NAS methods to find the optimal CNN architecture automatically, eliminating the need for human trial and error.

REFERENCES

[1] Deep Content-Based Music Genre Classification (Keunwoo Choi, George Fazekas, Mark Sandler, 2016)

[2] Convolutional Recurrent Neural Networks for Music Classification (Jordi Pons, Thomas Lidy, Xavier Serra, 2017)

[3] Learning to Recognize Musical Genre from Audio (Tristan Jehan, Brian Whitman, 2002)

[4] Music Genre Classification using MFCC And CNN by Vaibhav Vyas, Swarnil Mehta, Dhanashree Pati -2018

[5] Music Genre Classification with CNN and Data Augmentation by Somnath Roy, Preetam Kumar, Amit Konar-2020.

[6] End-to-End Learning for Music Audio by Keunwoo Choi, George Fazekas, Mark Sandler, Kyunghyun Cho – 2017.

[7] Music Genre Classification Using Neural Network (Tarannum Shaikh, Ashish Jadhav)

[8] Music Genre Classification Using Convolutional Neural Network (Nitin Choudhury,Deepjyoti Deka,Satyajit Sarmah,Parismita Sarma)

[9] Music Genre Classification Using CNN(Jane Crystal Rodrigues, Manisha Naik Gaonkar)

[10] Music Genre Classification using Convolutional Recurrent Neural Networks(Noopur Srivastava,Shivam Ruhil,Gaurav Kaushal)

[11] Music Genre Classification with Paralleling Recurrent Convolutional Neural Network (Lin Feng, Shenlan Liu, Jianing Yao, 2017).

[12] Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification (Mingwen Dong,2018)

[13] Music Genre Classification Using Convolutional Neural Network(Qiuqiang Kong, Xiaohui Feng, Yanxiong Li,2014)

[14] Music Genre Recognition Using Convolutional Neural Networks (Mateusz Matocha, Sławomir K. Zielinski,2018)

[15] Musical Genre Classification using Convolutional Neural Networks (R. Thiruvengatanadhan,2020