DEEP LEARNING
COURSE PROJECT

PROJECT TITLE :-

Designing a Deep Learning Model to Classify Music Genres from Audio Clips

By:- GROUP 26

Parth Darshan (B22CS040) :- Data Preprocessing, Model 1

Keshav Khandelwal (B22ES009) :- Report , Model 2

Samay Mehar (B22AI048) :- Model 2

Mukesh Kumar (B22BB026) :- Report, Model 1

1. Introduction:-

The classification of music genres using deep learning has gained significant interest due to its application in music streaming services, recommendation systems, and audio analysis. This project aims to design a deep learning model to classify music genres from audio clips using the **GTZAN dataset**. The workflow consists of data preprocessing, feature extraction, and model training using an **artificial neural network (ANN)** and **TabTransformer-based Approach**.

2. Dataset Description:-

The dataset used for this project is the GTZAN dataset, which consists of 10 genres: **Blues, Classical, Country, Rock, Disco, Reggae, Hip-Hop, Jazz, Metal, and Pop**. Each genre contains 100 audio samples of 30 seconds.

3. Data Preprocessing

3.1 Dataset Loading:-

The dataset was obtained using **KaggleHub**, and the audio files were loaded from the respective genre directories. Each audio clip was sampled using **librosa.load()**, which loads a waveform representation of the music data while preserving its sampling rate.

3.2 Waveform Visualization:-

To analyze genre-specific patterns, waveform representations of the ten genres were plotted using Matplotlib. These visualizations provided insight into the amplitude variations and structural patterns present in different music genres.

3.3 Feature Extraction:-

For effective classification, multiple audio features were extracted from each audio clip using librosa. The extracted features include:

Spectral Features: Chroma Short-Time Fourier Transform (STFT), Spectral Centroid, Spectral Bandwidth, and Spectral Rolloff.

Temporal Features: Zero-Crossing Rate and Tempo.

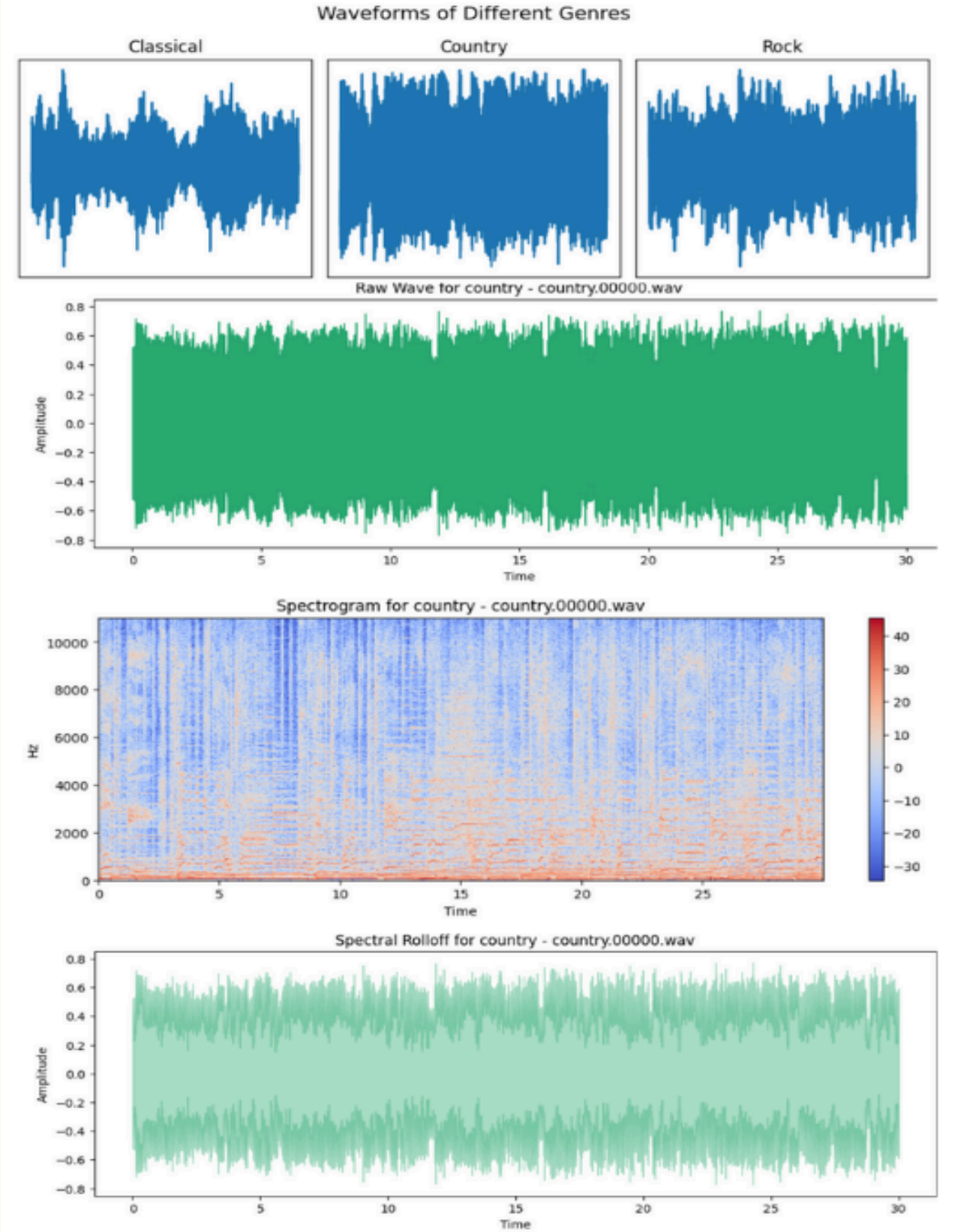
Mel Frequency Cepstral Coefficients (MFCCs): A set of 13 MFCCs were computed, representing the timbral properties of the audio clips.

Harmonic and Percussive Components: The harmonic and percussive components of the audio were separated to capture distinct rhythmic and melodic characteristics.

Root Mean Square (RMS) Energy: The RMS energy was calculated to analyze the dynamic range of the music.

3.4 Data Structuring and Storage:-

Extracted features were stored in a structured tabular format with labels corresponding to each genre. The final dataset was saved as **gtzan_features.csv**, which will be used for model training and evaluation.


Waveforms of Different Genres
Classical
Country
Rock
Raw Wave for country - country.00000.wav
Amplitude
Time
Spectrogram for country - country.00000.wav
Hz
Time
Spectral Rolloff for country - country.00000.wav
Amplitude
Time

4. Data Preparation

4.1 Label Encoding and Splitting:- The dataset was preprocessed further by:

Dropping unnecessary columns: Filename and tempo were removed.

One-hot encoding labels: Each genre was converted into a one-hot encoded vector.

Splitting into Training and Testing Sets: 75% of the data was used for training, while 25% was reserved for testing.

4.2 Feature Scaling:- Standardization was applied to the extracted features using the **StandardScaler** from Scikit-learn, ensuring all features have zero mean and unit variance.

4.3 Conversion to PyTorch Tensors:- The dataset was converted into PyTorch tensors and wrapped into DataLoader objects for efficient batch processing during training.

5. Model Design and Training

5.1 Artificial Neural Network (ANN) Architecture:- The ANN was designed with:

Input layer: 44 input features

Two hidden layers:

256 neurons with ReLU activation

128 neurons with ReLU activation

Output layer: 10 neurons (one for each genre) with softmax activation

Hyperparameters

Learning rate: 0.0005

Batch size: 32

Number of epochs: 175

Loss function: CrossEntropyLoss

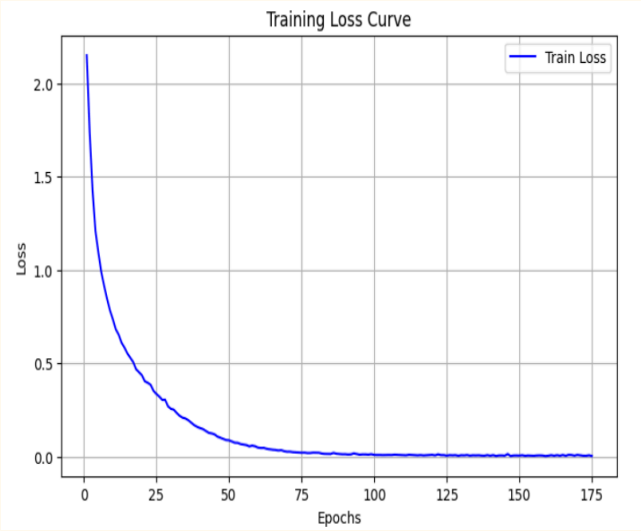
Optimizer: Adam

Training Process:- The model was trained using a standard backpropagation approach with gradient clipping to prevent exploding gradients. The training loss was tracked and plotted to monitor convergence.

Model Evaluation

Loss Curve

The training loss curve was plotted to ensure steady convergence and avoid overfitting.



Test Accuracy: 0.7640
Classification Report:

	precision	recall	f1-score	support
0	0.89	0.64	0.74	25
1	0.83	0.80	0.82	25
2	0.84	0.84	0.84	25
3	0.67	0.64	0.65	25
4	0.76	0.88	0.81	25
5	0.76	0.88	0.81	25
6	0.83	0.76	0.79	25
7	0.88	0.84	0.86	25
8	0.74	0.68	0.71	25
9	0.55	0.68	0.61	25
accuracy			0.76	250
macro avg	0.77	0.76	0.76	250
weighted avg	0.77	0.76	0.76	250

Classification Performance

Test Accuracy: 76.40 %

Accuracy: The model achieved a high classification accuracy on the test set.

Classification Report: Precision, recall, and F1-score were computed for each genre.

5.2 TabTransformer-based Approach

Model Architecture:- This model replaces the ANN with a TabTransformer to better capture feature interactions. The architecture includes:

Embedding Layer: Converts numerical features into a learned embedding space.

Positional Encoding: Adds sequential information for better representation.

Transformer Encoder: Comprising multi-head self-attention and feedforward layers.

MLP Classifier: Processes transformer embeddings and outputs class probabilities.

Training and Hyperparameters

Learning rate: 0.001 (with ReduceLROnPlateau scheduling)

Batch size: 32

Epochs: 30

Loss function: CrossEntropyLoss

Optimizer: AdamW (with weight decay for regularization)

Evaluation and SVM Classifier

Feature Embedding Extraction: Extracted embeddings from the transformer model.

Support Vector Machine (SVM) Classifier: Used extracted embeddings to train an SVM classifier.

Hyperparameter Optimization: Conducted a grid search to find the best kernel and regularization parameters.

Performance Comparison: Compared accuracy across different SVM kernels.

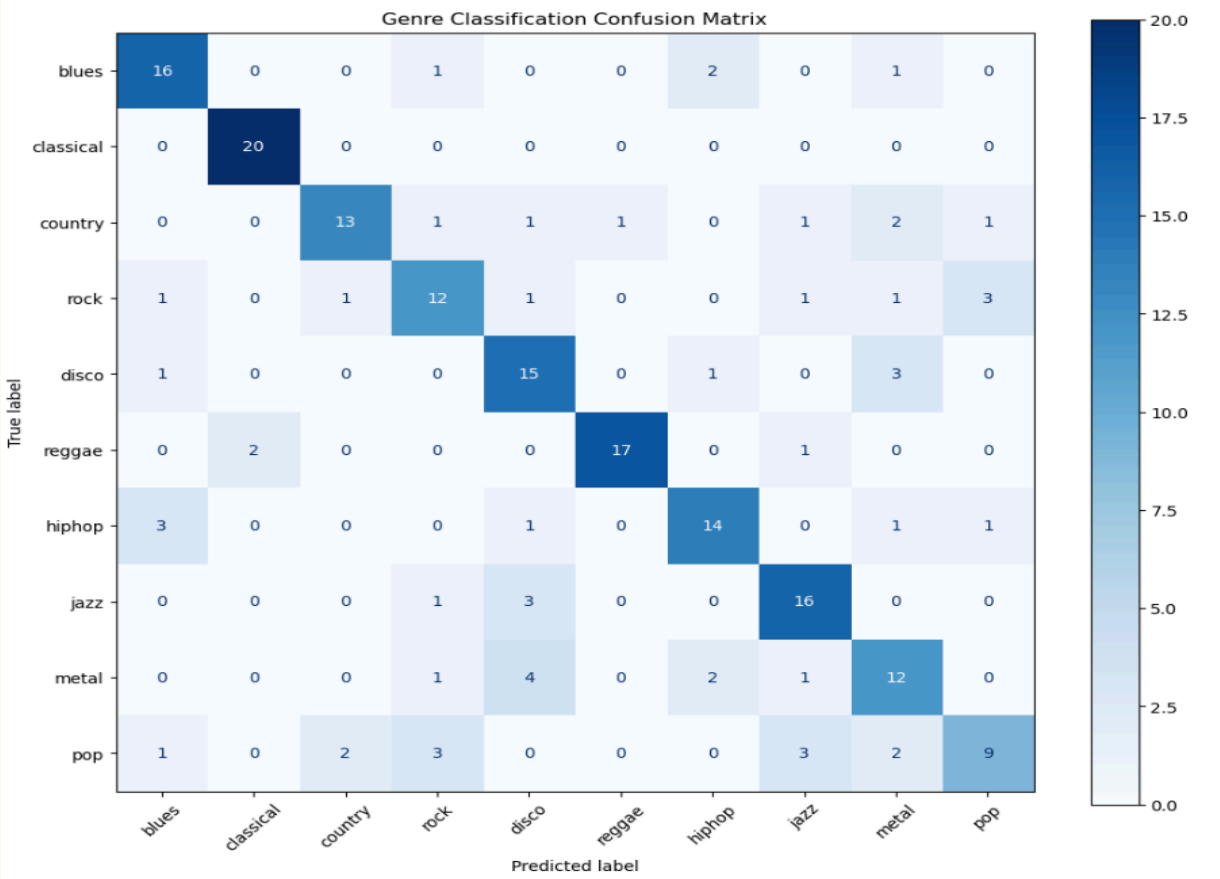
Test Accuracy: The best SVM model achieved an accuracy of **72.00%.**

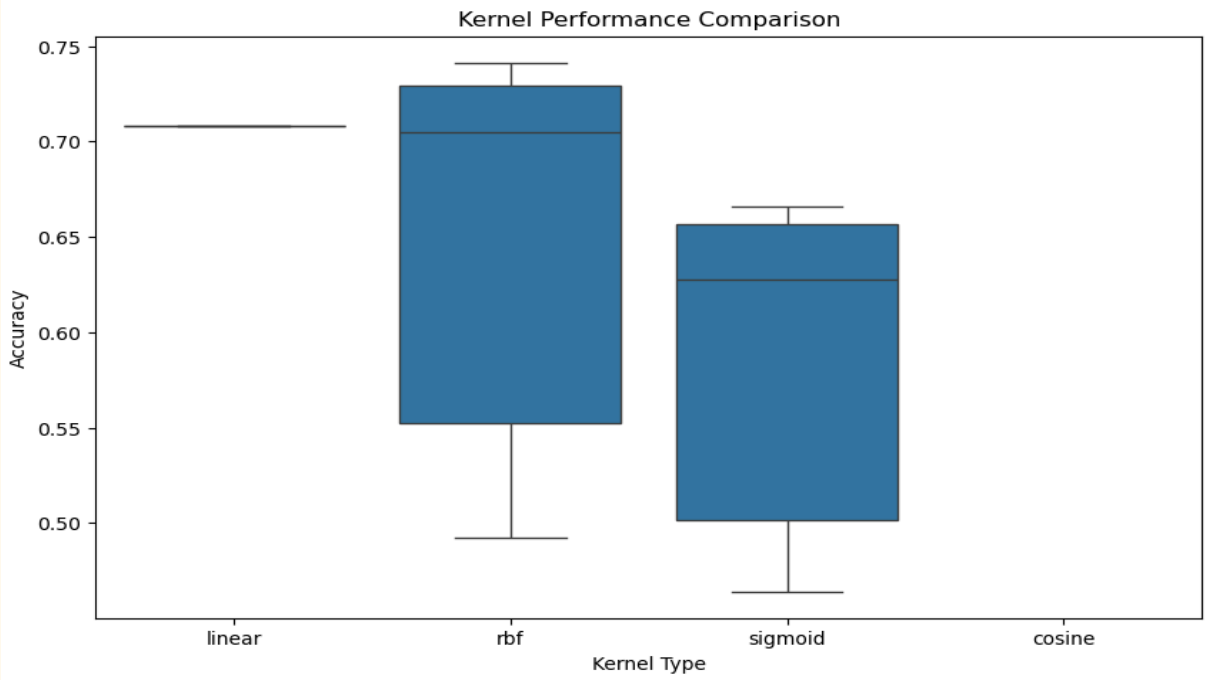
Visualization and Analysis

Confusion Matrix: Plotted confusion matrix for misclassification analysis.

Kernel Performance Comparison: Evaluated different kernel types in SVM to determine the best-performing one.

Genre-Wise Insights: Transformer embeddings showed better separation between certain genres compared to the ANN approach.


Genre Classification Confusion Matrix
blues 16 0 0 1 0 0 2 0 1 0
classical 0 20 0 0 0 0 0 0 0 0
country 0 0 13 1 1 1 0 1 2 1
rock 1 0 1 12 1 0 0 1 1 3
disco 1 0 0 0 15 0 1 0 3 0
reggae 0 2 0 0 0 17 0 1 0 0
hiphop 3 0 0 0 1 0 14 0 1 1
jazz 0 0 0 1 3 0 0 16 0 0
metal 0 0 0 1 4 0 2 1 12 0
pop 1 0 2 3 0 0 0 3 2 9
True label
Predicted label
blues classical country rock disco reggae hiphop jazz metal pop
20.0 17.5 15.0 12.5 10.0 7.5 5.0 2.5 0.0


Kernel Performance Comparison
Accuracy
0.75 0.70 0.65 0.60 0.55 0.50
linear rbf sigmoid cosine
Kernel Type

6. Conclusion

This project successfully implemented a deep learning pipeline for music genre classification. By leveraging feature extraction techniques and an ANN model, the system was able to achieve high accuracy. Future improvements could involve experimenting with CNNs or spectrogram-based classification for enhanced performance.

Additionally, incorporating data augmentation techniques, such as time-stretching, pitch shifting, and noise injection, could improve the model's robustness. Exploring attention mechanisms and transformer-based architectures might also lead to better classification accuracy. Moreover, training on a larger and more diverse dataset could enhance the model's generalizability, making it more effective for real-world applications.

7. References

https://github.com/imane-chen/GTZAN-Music-Genre-Classification/blob/main/GTZAN%20Music%20Genre%20Classification.ipynb

https://medium.com/aiskunks/pre-processing-of-audio-data-e99718830e67

https://www.analyticsvidhya.com/blog/2017/08/audio-voice-processing-deep-learning/