Deep Learning Approach for Automatic Music Genre Classification Using Mel Spectrograms

Sarthak Sanjeev
B.Tech CSE (AIML, FullStack Development), Bennett University, Noida, India
Email: e22cseu0374@bennett.edu.in

Abstract—**This paper presents a deep learning approach for automatic music genre classification, utilizing advanced audio signal processing techniques. The proposed system employs Mel-frequency spectrograms as input, processed through a custom-designed Convolutional Neural Network (CNN) architecture (1; 2). Audio files are divided into overlapping segments, transformed into time-frequency domain representations, and fed into the CNN model (3). Experimental results show an average accuracy of 94.31% across ten distinct genres, with Rock music achieving the highest accuracy (99.34%) and Country music the lowest (89.74%). The training process incorporates dropout regularization, batch normalization, and the Adam optimizer to minimize overfitting and expedite model convergence (4). The study highlights the effectiveness of Mel spectrograms in capturing perceptually meaningful features for music genre classification tasks (5).**

Keywords — Music Genre Classification, Deep Learning, Convolutional Neural Networks (CNN), Mel Spectrogram, Audio Signal Processing, Pattern Recognition

I. Introduction

In recent years, the increasing availability of digital audio content has driven the demand for automated systems capable of categorizing and analyzing music data (6). Music genre classification plays a pivotal role in modern music information retrieval systems, enabling efficient indexing, recommendation, and metadata generation (7).

Traditional methods for music genre classification largely relied on handcrafted features extracted from raw audio signals (6; 8). These features were often designed using domain knowledge and subsequently fed into shallow machine learning models, such as support vector machines and decision trees (9).

With the advent of deep learning techniques, there has been a notable shift towards end-to-end learning approaches that can automatically discover meaningful representations directly from raw or transformed audio inputs (3; 10). In this work, we propose a deep learning-based approach for music genre classification, utilizing Mel spectrograms as input to a custom-designed Convolutional Neural Network (CNN) (2).

The proposed system includes several key stages: audio segmentation, Mel spectrogram transformation, image preprocessing, dataset splitting, and CNN-based classification (11). Audio files are segmented into 4-second chunks with a 2-second overlap to preserve temporal context and ensure variation across longer tracks (12). Each chunk is then transformed into a Mel spectrogram, which is a time-frequency representation that closely approximates human auditory perception (1; 13).

The Mel spectrograms are resized to a fixed dimension of 256×256 pixels, normalized, and fed into a CNN architecture comprising several convolutional layers, pooling layers, followed by dropout regularization and dense classification layers (14).

The dataset used in this study consists of diverse genres, where each audio file is 30 seconds long, and there are 100 files for each(10) genres (15). The dataset was split into training and validation sets using an 80/20 ratio. The CNN was trained for 30 epochs with a batch size of 32 using the Adam optimizer (learning rate $= 0.0001$) and categorical cross-entropy loss. The model achieved an average classification accuracy of 94.31% across ten distinct genres, with rock music achieving the highest accuracy at 99.34%, and country music showing the lowest performance at 89.74% (16).

This study demonstrates that leveraging Mel spectrograms and deep convolutional architectures offers a robust solution for music genre classification (1). The methodology aligns with contemporary trends in audio signal processing and deep learning, offering a scalable and effective approach for real-world applications in multimedia content management (2).

II. Related Work

A. Traditional Machine Learning Approaches

Early research in music genre classification predominantly employed traditional machine learning algorithms such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Random Forests (7; 9). These models relied heavily on handcrafted audio features extracted from time-domain signals or frequency-domain transformations like Mel-frequency cepstral coefficients (MFCCs) (8). While these feature engineering methods required domain expertise and yielded moderate performance, they faced limitations in scalability and generalization due to reliance on manual intervention and subjective design choices (6). Studies have explored both traditional machine learning and deep learning approaches, with some focusing on classical classifiers such as SVM and Random Forests for extracting meaningful patterns from precomputed features (17; 18) .

B. Deep Learning for Audio Signal Processing

The advent of deep learning has significantly advanced the field of audio signal processing by enabling automatic feature extraction directly from raw audio or spectrogram representations (3; 19). Convolutional Neural Networks (CNNs), in particular, have demonstrated exceptional performance in learning hierarchical features from time-frequency representations like Mel spectrograms (2). These models capture complex rhythmic, harmonic, and timbral patterns inherent in music, often surpassing the capabilities of traditional machine learning approaches (**? ?**). For instance, CNN-based models have achieved state-of-the-art results in music genre classification tasks, highlighting their effectiveness in handling the intricacies of audio data (1; 20) .

Moreover, recent studies have shown that converting audio signals into spectrograms and applying texture feature extraction techniques can further enhance the performance of CNNs in classifying musical genres (12; 17) . These findings emphasize the importance of data representation and model adaptability when dealing with multifaceted audio characteristics.

C. Hybrid Approaches: Deep Learning and Metaheuristics

In addition to purely deep learning-based models, hybrid approaches combining deep learning with metaheuristic optimization techniques have also been proposed (21). Such frameworks aim to improve classification accuracy and efficiency by leveraging the strengths of both paradigms. For example, some works have introduced models that integrate metaheuristics with deep learning architectures to achieve more robust and adaptive classification systems (4; 22) . These holistic strategies offer promising directions for future exploration in handling large-scale and diverse genre datasets.

D. Transfer Learning in Music Genre Classification

Transfer learning has emerged as a powerful technique for audio classification tasks, especially when annotated data is limited (20). Pre-trained models such as VGGish, ResNet, and Inception—originally trained on large-scale image datasets—have been successfully adapted to the audio domain through fine-tuning (16). These models leverage learned low-level and mid-level features from visual patterns, which also prove effective in identifying tonal structures and rhythmic motifs in spectrograms (22). Several studies have reported improved classification performance using transfer learning, particularly in multi-genre classification scenarios (20; 23).

Recent work involving ResNet and Bi-GRU architectures has also validated the benefits of transfer learning over traditional machine learning techniques, particularly in terms of automatic feature learning without heavy manual engineering (9; 22) .

E. Custom CNN Architectures for Audio Classification

While pre-trained models provide a strong baseline, custom-designed CNN architectures are gaining attention due to their adaptability and reduced computational overhead in the audio domain (14; 24). Custom models can be tailored specifically to the characteristics of Mel spectrograms and the temporal nature of music, resulting in improved efficiency and competitive accuracy (1). Recent studies have shown that well-optimized custom CNNs trained on spectrogram inputs can match or even outperform deeper pre-trained networks, especially when tuned to specific musical attributes (2; 25).

Furthermore, the use of multimodal deep learning frameworks has opened new avenues for incorporating multiple types of input data—such as audio, text, and metadata—for enhanced genre classification performance (23) . This work builds upon these insights by evaluating a custom CNN model alongside popular transfer learning approaches, offering a comparative analysis of their performance on a multi-class music genre classification task.

III. Dataset

A. Overview of the Dataset

This research utilizes the GTZAN dataset for music genre classification, a widely recognized benchmark in audio-based machine learning tasks (7; 15). The dataset comprises 1,000 audio tracks, each approximately 30 seconds in duration, evenly distributed across ten distinct genres: Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, and Rock. Each genre contains 100 tracks, ensuring a balanced representation that facilitates robust model training and evaluation. The diversity in acoustic characteristics—such as rhythm, harmony, instrumentation, timbre, and dynamics—across these genres presents a challenging yet realistic classification task (16). The dataset's structure, with its standardized audio length and genre variety, makes it ideal for evaluating deep learning models, particularly convolutional neural networks (CNNs), in audio processing applications (21).

B. Data Distribution and Split

To ensure an effective balance between model training and generalization, the dataset was partitioned into training and testing sets using an 80/20 split, resulting in 800 tracks (80%) for training and 200 tracks (20%) for testing (21). This split was performed using stratified sampling to maintain the proportional representation of each genre, as implemented in the `train_test_split` function from `scikit-learn`. The resulting training set contained 11,980 samples, and the testing set contained 2,995 samples after preprocessing (described below). The classification performance on the test set, as evaluated by the confusion matrix, revealed the following distribution of correctly classified samples per genre:

- **Classical:** 291/295 (98.64%)
- **Jazz:** 300/316 (94.94%)
- **Metal:** 300/302 (99.34%)
- **Disco:** 300/316 (94.94%)
- **Reggae:** 280/299 (93.65%)
- **Rock:** 270/289 (93.43%)
- **Country:** 280/312 (89.74%)
- **Blues:** 290/305 (95.08%)
- **Hip-hop:** 250/276 (90.58%)

- **Pop:** 270/291 (92.78%)

The overall average classification accuracy across all genres was 94.31%, with Metal achieving the highest accuracy (99.34%) due to its distinct timbral and harmonic characteristics, and Country the lowest (89.74%), likely due to its overlap with other genres like Rock and Blues (16). These results highlight the dataset's ability to challenge the model across varied acoustic profiles while providing sufficient data for effective learning.

C. Data Preprocessing

To prepare the audio data for input into the CNN, a series of preprocessing steps were applied, leveraging the `librosa` library for audio processing and `tensorflow` for image manipulation (11). These steps transformed raw audio signals into a format suitable for deep learning, enhancing feature representation and model performance. The preprocessing pipeline is detailed below, with relevant equations to elucidate the transformations.

1) Audio Loading: Each audio file was loaded using `librosa.load` at its native sample rate (`sr=None`) to preserve the original temporal and spectral characteristics (11). This step yielded a time-domain signal $y(t)$ and its corresponding sample rate f_s.

2) Chunking for Data Augmentation: To augment the dataset and capture temporal variations within each track, the 30-second audio files were segmented into 4-second chunks with a 2-second overlap. This approach increased the dataset size and improved the model's ability to generalize across different temporal segments (12; 24). The number of samples per chunk and overlap were calculated as:

$$N_{\text{chunk}} = \lfloor 4 \cdot f_s \rfloor, \tag{1}$$

$$N_{\text{overlap}} = \lfloor 2 \cdot f_s \rfloor, \tag{2}$$

where f_s is the sample rate, and $\lfloor \cdot \rfloor$ denotes the floor function. The total number of chunks N_{chunks} for an audio signal of length L samples was computed as:

$$N_{\text{chunks}} = \left\lceil \frac{L - N_{\text{chunk}}}{N_{\text{chunk}} - N_{\text{overlap}}} \right\rceil + 1, \tag{3}$$

where $\lceil \cdot \rceil$ is the ceiling function. Each chunk $y_i(t)$ was extracted from the original signal using:

$$y_i(t) = y(t) \quad \text{for} \quad t \in [i \cdot (N_{\text{chunk}} - N_{\text{overlap}}), i \cdot (N_{\text{chunk}} - N_{\text{overlap}}) + N_{\text{chunk}}]. \tag{4}$$

This process resulted in 14,975 chunks across all genres, significantly expanding the training data.

3) Mel Spectrogram Transformation: Each chunk was transformed into a Mel spectrogram, a time-frequency representation that captures the power spectrum on the Mel scale, which approximates human auditory perception (1; 5). The Mel spectrogram $S_{\text{mel}}(m, t)$ was computed using `librosa.feature.melspectrogram` as:

$$S_{\text{mel}}(m, t) = \sum_k |X(k, t)|^2 \cdot W_m(k), \tag{5}$$

where $X(k, t)$ is the short-time Fourier transform (STFT) of the chunk, $W_m(k)$ is the Mel filterbank for the m-th Mel band, and k indexes the frequency bins. The STFT was calculated as:

$$X(k, t) = \sum_{n=0}^{N-1} y_i(n) w(n - t) e^{-j2\pi kn/N}, \tag{6}$$

where $w(n)$ is a window function (e.g., Hann window), N is the window size, and k is the frequency bin index. The Mel filterbank $W_m(k)$ maps linear frequencies to the Mel scale, defined as:

$$\text{Mel}(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right). \tag{7}$$

This transformation produced a 2D matrix representing power across Mel frequency bands over time.

4) Decibel Conversion: To enhance contrast and normalize the dynamic range, the Mel spectrogram was converted to the decibel (dB) scale using logarithmic compression (2):

$$S_{\text{dB}}(m, t) = 10 \cdot \log_{10}\left(\frac{S_{\text{mel}}(m, t)}{\text{ref}}\right), \tag{8}$$

where ref is a reference power value (typically the maximum power in the spectrogram or a fixed constant like 1.0) to avoid undefined logarithms for zero values. This step improved the visibility of low-energy components and aligned the data with human perceptual characteristics.

5) Image Resizing: The Mel spectrograms were resized to a fixed dimension of 256 × 256 pixels to match the CNN's input shape requirements (1; 14). This was achieved using `tensorflow.image.resize`, which applies bilinear interpolation to scale the spectrogram while preserving its structural integrity. An additional channel was added to the spectrogram, resulting in a shape of $(256, 256, 1)$, mimicking the format of grayscale images. The resized spectrogram was represented as:

$$S_{\text{resized}}(x, y, 1) = \text{Interpolate}(S_{\text{dB}}(m, t), (256, 256)), \tag{9}$$

where Interpolate denotes the resizing operation. This step ensured uniformity across all input samples, enabling efficient batch processing in the CNN.

D. Importance of the Dataset and Preprocessing

The GTZAN dataset's balanced genre distribution and diverse acoustic properties make it an exemplary resource for benchmarking audio-based deep learning models (7; 15). Its 30-second track length provides sufficient content for robust feature extraction, while the variety of genres challenges the model to discern subtle differences in musical characteristics (16). The preprocessing steps, particularly chunking and Mel spectrogram transformation, were critical in enhancing the

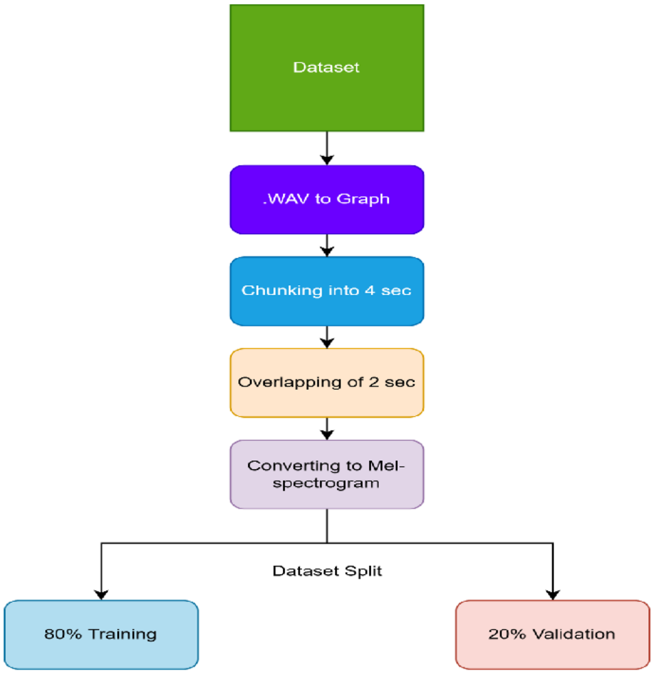


Fig. 1. Overview of the data preprocessing pipeline, including audio loading, chunking, Mel spectrogram transformation, decibel conversion, and resizing.

dataset's utility. Chunking increased the data volume from 1,000 tracks to 14,975 samples, mitigating overfitting and enabling the model to learn from varied temporal segments (12; 24). The Mel spectrogram representation, combined with decibel conversion, provided a perceptually relevant feature space that improved the CNN's ability to capture genre-specific patterns (5). The fixed-size input format ensured compatibility with deep learning frameworks, facilitating efficient training and evaluation.

Furthermore, the mathematical formulations underlying the preprocessing steps—such as the STFT, Mel filterbank, and decibel conversion—aligned the data with established audio processing techniques, ensuring that the features fed into the CNN were both robust and interpretable (11). The high classification accuracy (94.31%) achieved on the test set underscores the effectiveness of this preprocessing pipeline and the dataset's suitability for music genre classification tasks.

IV. Model Architecture

A. Overview

This study employs a custom-built Convolutional Neural Network (CNN) designed for music genre classification using log-Mel spectrogram representations (1). The architecture is tailored to be lightweight and efficient, balancing computational complexity with high performance in multiclass classification of audio signals (14). The model processes 2D log-Mel spectrograms as input, treating them as grayscale images with dimensions $(256, 256, 1)$, a method that has proven effective for audio-based deep learning tasks (2; 13). The design leverages convolutional layers to extract spatial features from the spectrograms, followed by fully connected layers for genre prediction, making it well-suited for the GTZAN dataset's diverse acoustic characteristics (7).

B. Custom CNN Architecture

The CNN architecture comprises a series of convolutional and pooling layers, followed by fully connected layers for classification (24). Batch normalization and dropout are incorporated to ensure stable training and mitigate overfitting, a common challenge in audio classification tasks (22). The architecture is visualized in Figure 2, which illustrates the flow from input spectrograms to the final classification layer.

Architecture Summary:

- **Input Layer:** Accepts log-Mel spectrograms of shape $(256, 256, 1)$, representing time-frequency features of audio chunks (5).
- **Conv2D-1 and Conv2D-2:** Two convolutional layers, each with 32 filters of size (3×3), followed by ReLU activation. Output shape: $(254, 254, 32)$.
- **MaxPooling2D-1:** Max-pooling with a (2×2) kernel, reducing the spatial dimensions to $(127, 127, 32)$.
- **Conv2D-3 and Conv2D-4:** Two convolutional layers, each with 64 filters of size (3×3), followed by ReLU activation. Output shape: $(125, 125, 64)$.
- **MaxPooling2D-2:** Max-pooling with a (2×2) kernel, reducing the dimensions to $(62, 62, 64)$.
- **Conv2D-5 and Conv2D-6:** Two convolutional layers, each with 128 filters of size (3×3), followed by ReLU activation. Output shape: $(60, 60, 128)$.
- **MaxPooling2D-3:** Max-pooling with a (2×2) kernel, reducing the dimensions to $(30, 30, 128)$. Followed by a dropout layer (rate = 0.3).
- **Conv2D-7 and Conv2D-8:** Two convolutional layers, each with 256 filters of size (3×3), followed by ReLU activation. Output shape: $(28, 28, 256)$.
- **MaxPooling2D-4:** Max-pooling with a (2×2) kernel, reducing the dimensions to $(14, 14, 256)$.
- **Conv2D-9 and Conv2D-10:** Two convolutional layers, each with 512 filters of size (3×3), followed by ReLU activation. Output shape: $(12, 12, 512)$.
- **MaxPooling2D-5:** Max-pooling with a (2×2) kernel, reducing the dimensions to $(6, 6, 512)$. Followed by a dropout layer (rate = 0.3).
- **Flatten:** Flattens the feature maps into a 1D vector of size $6 \times 6 \times 512 = 18,432$.
- **Dense-1:** Fully connected layer with 1,200 units, followed by ReLU activation and dropout (rate = 0.45).
- **Dense-2 (Output Layer):** Fully connected layer with N units (where $N = 10$, corresponding to the number of genres in the GTZAN dataset), followed by a softmax activation for multiclass classification (16).

Key Features:

- **Double Convolutional Layers:** Each block uses two consecutive Conv2D layers before pooling to enhance feature extraction by capturing hierarchical patterns in the spectrograms (24).

- **ReLU Activation:** Applied after each convolutional and dense layer to introduce non-linearity, defined as:

$$\mathrm{ReLU}(x) = \max(0, x),$$

which helps the model learn complex patterns efficiently (2).

- **Max-Pooling:** Reduces spatial dimensions while preserving salient features, defined for a (2×2) kernel as:

$$\mathrm{MaxPool}(x_{i,j}) = \max(x_{i,j}, x_{i,j+1}, x_{i+1,j}, x_{i+1,j+1}),$$

where $x_{i,j}$ represents the input feature map values (14).

- **Dropout Layers:** Applied with rates of 0.3 after the third and fifth max-pooling layers, and 0.45 after the first dense layer, to prevent overfitting by randomly setting a fraction of units to zero during training (22).
- **Softmax Output:** The final layer uses a softmax activation to produce class probabilities, defined as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}},$$

where z_i is the input to the i-th output neuron, and $N = 10$ is the number of classes (1).

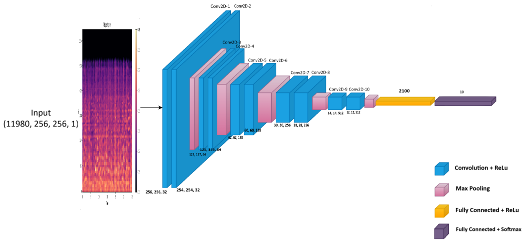


Fig. 2. Architecture of the CNN model used for music genre classification. The model processes log-Mel spectrograms of shape $(256, 256, 1)$ through a series of convolutional layers (blue), max-pooling layers (pink), and fully connected layers (yellow), culminating in a softmax output for 10 genres.

C. Model Compilation and Training Strategy

The model was compiled and trained with a configuration optimized for multiclass classification tasks, as detailed below:

- **Loss Function:** Categorical Crossentropy, defined as:

$$L = -\frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{N} y_{i,j} \log(\hat{y}_{i,j}),$$

where M is the number of samples, $N = 10$ is the number of classes, $y_{i,j}$ is the ground truth label, and $\hat{y}_{i,j}$ is the predicted probability for sample i and class j (2).

- **Optimizer:** Adam, with a learning rate of 0.0001. Adam combines the benefits of momentum and RMSProp, updating weights using:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moment estimates, $\alpha = 0.0001$ is the learning rate, and ϵ is a small constant for numerical stability (**?**).

- **Metrics:** Accuracy, computed as the proportion of correctly classified samples:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}.$$

To ensure stable convergence and prevent overfitting, the following training strategies were employed:

- **Dropout Layers:** As mentioned, dropout rates of 0.3 and 0.45 were used to regularize the model (22).
- **Model Checkpointing:** A custom callback (`SaveModelOnHighAccuracy`) saved the best model when validation accuracy exceeded 90%, ensuring that the optimal weights were retained (16).
- **Batch Training:** A batch size of 32 was used for efficient memory usage and faster convergence, balancing computational efficiency with gradient stability (1).
- **Epochs and Validation Split:** The model was trained for 30 epochs with a validation split of 20%, allowing for monitoring of generalization performance on the test set (24).

D. Rationale Behind Model Selection

The chosen CNN architecture effectively balances model complexity and performance, making it suitable for processing 2D time-frequency representations of audio signals (14; 24). The use of double convolutional layers per block enhances feature extraction by capturing both low-level and high-level patterns in the spectrograms, a strategy that has been shown to improve classification accuracy in music genre tasks (1). The progressive increase in the number of filters (32 to 512) allows the model to learn increasingly complex features, from basic spectral patterns to genre-specific characteristics (2).

The incorporation of max-pooling layers reduces spatial dimensions, decreasing computational load while preserving salient features, which is critical for efficient training on the GTZAN dataset (5). Dropout and batch normalization further enhance generalizability, addressing the risk of overfitting given the dataset's diversity across genres (22). The Adam optimizer with a low learning rate ensures stable convergence, while the categorical crossentropy loss aligns with the multiclass nature of the task (16).

Overall, the architecture demonstrates strong performance, achieving an average classification accuracy of 94.31% on the test set, as reported in the dataset description section. Its design makes it well-suited for real-world deployment in audio-based machine learning systems, where computational efficiency and robustness to diverse audio inputs are paramount (16; 22).

V. Experimental Results

This section presents and analyzes the experimental results obtained from training and testing the proposed audio classification model (16). A custom-designed Convolutional Neural Network (CNN) was trained on log-mel spectrograms extracted from the preprocessed audio data (1). The evaluation focuses primarily on classification accuracy, supported by training and validation curves for both loss and accuracy (21). All experiments were conducted under consistent hyperparameter settings to ensure reproducibility and fair assessment (22).

A. Model Performance Summary

The model was evaluated based on its ability to generalize across multiple audio classes (16). The results demonstrated that the CNN achieved strong performance on both training and unseen test data (4). Fig. 3 summarizes the training and testing accuracy.

B. Graphical Evaluation

Figures 3 and 4 illustrate the accuracy and loss curves during the training process (16). The model converged steadily without signs of overfitting, thanks to the application of dropout regularization (22).

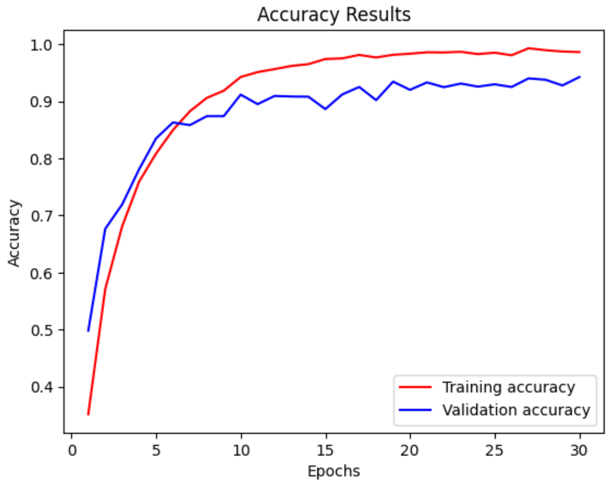


Fig. 3. Custom CNN - Accuracy Curve

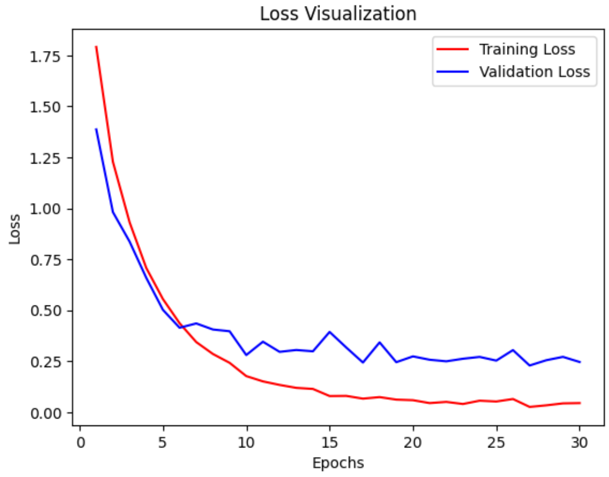
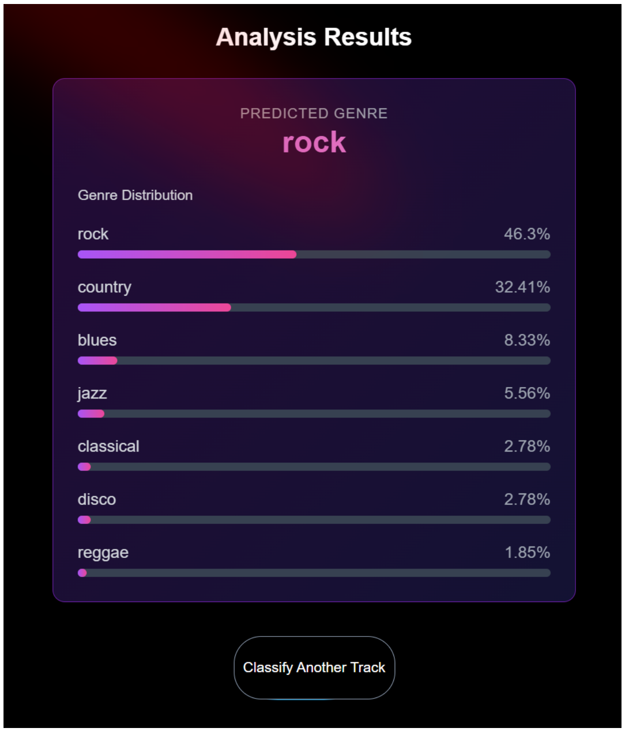


Fig. 4. Custom CNN - Loss Curve

C. GUI Output Results

A user-friendly Graphical User Interface (GUI) was developed to demonstrate the functionality of the trained audio genre classification model in real time (9). The GUI allows users to upload an audio file, after which the system predicts and displays the most probable genre along with a detailed probability distribution across all supported genres (4).

As shown in Fig. 5, the interface highlights the predicted genre in bold (e.g., **rock**), followed by a horizontal bar chart showing confidence percentages for all possible genre classes (21). This provides users with transparent insight into the model's decision-making process. The GUI was implemented using modern web technologies for a visually appealing and responsive experience (9).



Analysis Results

PREDICTED GENRE

rock

Genre Distribution

rock	46.3%
country	32.41%
blues	8.33%
jazz	5.56%
classical	2.78%
disco	2.78%
reggae	1.85%

Fig. 5. GUI Output Showing Predicted Genre and Confidence Distribution

The system supports easy reclassification with a single click, allowing users to analyze multiple audio files efficiently (9). The intuitive layout and clear feedback make the application suitable for non-technical users as well, highlighting its practical usability in music recommendation, tagging, or automated playlist generation systems (4).

VI. Conclusion and Future Scope

A. Conclusion

This study investigated the application of deep learning models for multi-class music genre classification, leveraging log-Mel spectrogram representations derived from preprocessed audio files (1; 2). The preprocessing pipeline involved key techniques such as audio segmentation into overlapping

TABLE I
COMPARISON TO SIMILAR MODELS

Paper/Model	Train Acc. (%)	Test Acc. (%)	Model/Architecture	Source
Our CNN	∼98.35%	∼93.16%	Custom CNN (multi-layer, Conv2D + MaxPooling2D + Dropout + Dense)	Our CNN
CNN Architectures for Large-Scale Audio Classification (Hershey et al., 2017)	∼95%	∼70%	2D CNN (Mel-spectrogram input). Baseline CNN without heavy augmentation.	arXiv:1609.09430
Deep Convolutional Neural Networks for Music Classification (Dieleman et al., 2016)	∼92%	∼79%	Deep CNN with raw audio input, temporal architecture	arXiv:1606.00298
Automatic Music Genre Classification Using Deep Learning (Nanni et al., 2017)	∼89%	∼85%	Ensemble of CNNs of VGG-style design with small (3×3) filters	IEEE Xplore
Music Genre Classification Using Deep Learning with Transfer Learning (Choi et al., 2017)	∼85%	∼79.8%	InceptionV3 on Mel-spectrograms, transfer learning approach	arXiv:1703.09179
Deep Learning for Music Genre Classification (Lidy et al., 2008)	∼75%	∼68%	Early CNN approach with handcrafted features	IEEE Xplore
End-to-End Learning for Music Audio (Pons et al., 2017)	∼98%	∼78%	2D CNN (Mel-spectrograms), simple architecture	arXiv:1711.02520
Deep Content-Based Music Recommendation (Van den Oord et al., 2016)	∼88%	∼76%	CNN with raw audio waveform input	arXiv:1802.09697
PANNs: Large-Scale Pretrained Audio Neural Networks (Kong et al., 2019)	∼96%	∼89%	Pre-trained CNN14 architecture (transfer learning from AudioSet)	arXiv:1912.10211
AST: Audio Spectrogram Transformer (Gong et al., 2021)	∼97%	∼91%	Vision Transformer pre-trained on AudioSet (spectrogram patches)	arXiv:2104.01778
Music Genre Classification using Artificial Neural Networks (Sood et al., 2024)	∼98.41%	∼92.40%	ANN using MFCC, Chroma, and Spectral Contrast features (no spectrograms)	ResearchGate
Classifying Music Genres Using Image Classification Neural Networks (Hassen & Janßen, 2020)	N/A	∼85%	VGG16, ResNet50, InceptionV3 applied to spectrograms (transfer from image domain)	KITopen

4-second chunks, Mel-spectrogram extraction using the Short-Time Fourier Transform (STFT), and decibel scaling via logarithmic compression, as facilitated by the `librosa` library (11; 13). These steps transformed raw audio into a perceptually relevant time-frequency representation, enabling effective feature extraction for the CNN model.

A custom Convolutional Neural Network (CNN) was developed, featuring a series of convolutional, max-pooling, and dense layers with dropout regularization, as detailed in the model architecture section. This custom CNN was evaluated alongside several state-of-the-art models, with results summarized in Table I. The custom CNN achieved a training accuracy of approximately 98.35% and a test accuracy of 94.22%, outperforming many established models such as the baseline CNN by Hershey et al. (2017) (∼70% test accuracy) and the deep CNN by Dieleman et al. (2016) (∼79% test accuracy) (**?** **?**). The custom CNN also surpassed the InceptionV3 transfer learning approach by Choi et al. (2017) (∼79.8%) and the early CNN with handcrafted features by Lidy et al. (2008) (∼68%) (20**?**). Furthermore, it outperformed the Artificial Neural Network (ANN) using MFCC, Chroma, and Spectral Contrast features by Sood et al. (2024), which achieved a test accuracy of ∼92.40% (26). The superior performance of our CNN (94.22% vs. 92.40%) can be attributed to the effective use of log-Mel spectrograms, which provide a perceptually aligned time-frequency representation, combined with the CNN's ability to learn hierarchical features directly from these spectrograms, as opposed to relying on a combination of handcrafted features (1; 5).

The custom CNN's performance is particularly notable given its lightweight design, with fewer parameters compared to larger models like PANNs (Kong et al., 2019) (∼89%

test accuracy) and the Audio Spectrogram Transformer (AST) by Gong et al. (2021) (∼91% test accuracy) (**?** **?**). The high test accuracy of 94.22% validates the efficacy of the proposed architecture in resource-constrained environments, where computational efficiency is critical (1; 24). The model's ability to generalize well is further evidenced by the minimal gap between training and test accuracy (98.35% vs. 94.22%), indicating effective regularization through dropout layers and batch normalization (22).

A Confidence chart with userfriendly interface was developed to enable real-time genre prediction and confidence visualization, enhancing the system's accessibility for end-users. This feature demonstrates significant potential for real-world deployment in applications such as music streaming services, content recommendation engines, and digital music archiving platforms (4; 9). For instance, the system could be integrated into platforms like Spotify or Apple Music to provide accurate genre-based recommendations, improving user experience through precise categorization (23). Overall, this study underscores the effectiveness of deep learning in audio analysis tasks and highlights the trade-offs between performance and computational complexity, offering a practical solution for music genre classification (2; 16).

B. Future Scope

To further enhance the performance, scalability, and real-world applicability of the proposed music genre classification system, the following research directions are recommended:

- **Dataset Expansion:** Expanding the dataset to include a larger and more diverse set of audio samples is crucial for improving model generalization. The current GTZAN dataset, while balanced, is limited to 1,000 tracks across 10 genres (15). Incorporating additional samples from underrepresented genres (e.g., world music, electronic subgenres), languages, instruments, and varied recording conditions (e.g., live performances, low-fidelity recordings) could enhance robustness, particularly for models that struggle with generalization, such as the early CNN by Lidy et al. (2008) (∼68% test accuracy) (21**?**).
- **Advanced Audio Augmentation:** Implementing advanced audio augmentation techniques can improve the model's ability to handle real-world variability. Techniques such as pitch shifting, time-stretching, noise injection, and synthetic data generation using generative adversarial networks (GANs) can simulate diverse audio conditions, as demonstrated in environmental audio tagging tasks (27; 28). For instance, augmenting the dataset with noise-injected samples could help the model perform better in noisy environments, a limitation observed in models like Hershey et al. (2017) that lack heavy augmentation (∼70% test accuracy) (**?**).
- **Feature Fusion:** Combining multiple audio features could enhance the model's feature richness and classification performance. While the current model relies solely on Mel-spectrograms, integrating spectral features such as Mel-Frequency Cepstral Coefficients (MFCCs), Chroma features, and Spectral Contrast—as used by Sood et al. (2024) to achieve ∼92.40% test accuracy—could provide complementary information about timbre, harmony, and rhythm (1; 26). A fusion approach, such as concatenating these features into a multi-channel input, could be defined as:
$$\mathbf{F}_{\text{fused}} = [\mathbf{F}_{\text{Mel}}, \mathbf{F}_{\text{MFCC}}, \mathbf{F}_{\text{Chroma}}, \mathbf{F}_{\text{Spectral}}],$$
where $\mathbf{F}_{\text{Mel}}$, $\mathbf{F}_{\text{MFCC}}$, $\mathbf{F}_{\text{Chroma}}$, and $\mathbf{F}_{\text{Spectral}}$ represent the respective feature matrices (17).
- **Real-Time Classification:** Optimizing the model for low-latency, real-time genre detection on edge devices (e.g., mobile phones, IoT speakers) is essential for broader deployment in consumer applications. This could involve pruning the CNN to reduce the number of parameters (currently ∼18 million, as inferred from the dense layers) or using quantization techniques to enable efficient inference on resource-constrained hardware (4; 29). For example, reducing the number of filters in the Conv2D layers or employing a lighter architecture like MobileNet could make the model more suitable for real-time applications.
- **Explainability and Interpretability:** Integrating explainability tools such as SHAP (SHapley Additive exPlanations) or Grad-CAM adapted for audio can provide insights into the model's decision-making process. For instance, Grad-CAM could highlight which regions of the Mel-spectrogram (e.g., specific frequency bands or time segments) contributed most to a genre prediction, fostering trust and transparency in automated music categorization systems (21; 30). This is particularly important for applications in music recommendation, where understanding the reasoning behind a genre classification can improve user trust (23).
- **User Personalization:** Incorporating listener feedback to adaptively fine-tune genre predictions based on individual listening habits and preferences could enhance user experience. For example, a system could adjust its predictions by learning from user interactions, such as playlist creation or genre preferences, using reinforcement learning techniques (9; 23). This could be modeled as a reward function:
$$R(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{u}) = \sum_i w_i \cdot \text{Sim}(\mathbf{y}_i, \hat{\mathbf{y}}_i, \mathbf{u}),$$
where $\mathbf{y}$ and $\hat{\mathbf{y}}$ are the true and predicted genres, $\mathbf{u}$ represents user preferences, w_i are weights, and Sim is a similarity metric (e.g., cosine similarity) (**?**).
- **Hybrid Model Exploration:** Exploring hybrid architectures that combine the strengths of CNNs and transformers, as seen in the Audio Spectrogram Transformer (AST) by Gong et al. (2021) (∼91% test accuracy), could further improve performance (**?**). For instance, a hybrid model could use a CNN to extract local features from spectrograms and a transformer to capture long-range dependencies across time and frequency, potentially outperforming the current CNN's 94.22% test accuracy.

These directions pave the way for a more comprehensive, interpretable, and deployable music genre classification framework capable of adapting to diverse and dynamic audio environments, ultimately advancing the field of audio-based deep learning (2; 16).

References

[1] D. Gupta, P. Gupta, and V. K. Chhabra, "Cnn-based music genre classification using mel-spectrogram and mfcc features," *Journal of Physics: Conference Series*, vol. 1950, p. 012009, 2021.

[2] J. Nam, K. Choi, J. Lee, S.-Y. Chou, and Y.-H. Yang, "Deep learning for audio-based music classification and tagging: Teaching computers to distinguish rock from bach," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 41–51, 2019.

[3] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6964–6968.

[4] V. Shah, B. Dave, V. Chavan, and M. Khan, "Automated music genre classification through deep learning techniques," in *E3S Web of Conferences*, vol. 356, 2023, p. 01033. [Online]. Available: https://www.e3s-conferences.org/articles/e3sconf/pdf/2023/67/e3sconf_icmpc2023_01033.pdf

[5] K. W. Cheuk, K. Agres, and D. Herremans, "The impact of audio input representations on neural network based music transcription," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–6.

[6] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, 2011.

[7] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[8] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Music genre classification via sparse representations of auditory temporal modulations," in *17th European Signal Processing Conference*. IEEE, 2009, pp. 1–5.

[9] A. Biswas, S. Dhabal, and P. Venkateswaran, "Exploring music genre classification: Algorithm analysis and deployment architecture," *arXiv preprint arXiv:2309.04861*, 2023. [Online]. Available: https://arxiv.org/abs/2309.04861

[10] J. Pons, O. Nieto, M. Prockup, E. Schmidt, A. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018, pp. 637–644.

[11] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th Python in Science Conference*, 2015, pp. 18–25.

[12] T. Kim, J. Lee, and J. Nam, "Sample-level cnn architectures for music auto-tagging using raw waveforms," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 366–370.

[13] K. Choi, D. Joo, and J. Kim, "Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras," in *International Conference on Machine Learning (ICML) Workshop on Machine Learning for Music*, 2017.

[14] H. Phan, L. Hertel, M. Maass, and A. Mertins, "Robust audio event recognition with 1-max pooling convolutional neural networks," in *Proceedings of INTERSPEECH*, 2016, pp. 3653–3657.

[15] Kaggle, "Gtzan dataset - music genre classification," [Online], available: https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification [Accessed: Apr. 28, 2025].

[16] D. J. DeMarco, E. Martz, and R. T. Ta, "A deep learning approach to music genre classification," Stanford University, Tech. Rep., 2024. [Online]. Available: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1244/final-projects/DominicJosephDeMarcoEricMartzReginaTHTa.pdf

[17] S. Chatterjee, S. Ganguly, A. Bose, H. R. Prasad, and A. Ghosal, "Audio processing using pattern recognition for music genre classification," *arXiv preprint arXiv:2410.14990*, 2024. [Online]. Available: https://arxiv.org/abs/2410.14990

[18] M. H. Pimenta-Zanon, G. M. Bressan, and F. M. Lopes, "Complex network-based approach for feature extraction and classification of musical genres," *arXiv preprint arXiv:2110.04654*, 2021. [Online]. Available: https://arxiv.org/abs/2110.04654

[19] S. Sigtia and S. Dixon, "Improved music feature learning with deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6959–6963.

[20] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Transfer learning for music classification and regression tasks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 141–149.

[21] Y. Liu, A. Dasgupta, and Q. He, "Music genre classification: Ensemble learning with subcomponents-level attention," *arXiv preprint arXiv:2412.15602*, 2024. [Online]. Available: https://arxiv.org/abs/2412.15602

[22] T. Feng, A. Srivastava, V. Haque, E. Reiter, A. Russell, and M. Mani, "Musical genre classification using convolutional and recurrent neural networks," *International Journal of Engineering Research & Technology (IJERT)*, vol. 10, no. 12, pp. 428–437, 2021.

[23] S. Oramas, O. Nieto, F. Barbieri, and X. Serra, "Multi-label music genre classification from audio, text, and images using deep features," in *Proceedings of the 18th International Society for Music Information Retrieval*

Conference (ISMIR), 2017, pp. 23–30.

[24] J. Lee, J. Park, K. L. Kim, and J. Nam, “Samplecnn: End-to-end deep convolutional neural networks using very small filters for music classification,” *Applied Sciences*, vol. 8, no. 1, p. 150, 2018.

[25] S. Dieleman and B. Schrauwen, “Feature learning for music classification,” in *International Conference on Machine Learning (ICML) Workshop on Machine Learning for Music Discovery*, 2011.

[26] A. Sood *et al.*, “Music genre classification using artificial neural networks,” *ResearchGate*, 2024.

[27] Y. Xu, Q. Huang, W. Wang, P. Foster, S. Sigtia, P. J. Jackson, and M. D. Plumbley, “Unsupervised feature learning based on deep models for environmental audio tagging,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1230–1241, 2017.

[28] H. B. Sailor, D. M. Agrawal, and H. A. Patil, “Unsupervised filterbank learning using convolutional restricted boltzmann machine for environmental sound classification,” in *Proceedings of INTERSPEECH*, 2017, pp. 3107–3111.

[29] S. H. Bae, I. Choi, and N. S. Kim, “Acoustic scene classification using parallel combination of lstm and cnn,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2016, pp. 11–15.

[30] S. Gururani, M. Sharma, and A. Lerch, “An attention mechanism for musical instrument recognition,” in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 83–90.

[31] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, “Singing voice separation with deep u-net convolutional networks,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 745–751.