Deep Music Genre Classification Using MFCC Features and Convolutional Neural Networks

Sai Arunanshu Govindarajula
College of Engineering and Computer Science
University of Michigan–Dearborn
Email: saiarun@umich.edu

Tejaswini
College of Engineering and Computer Science
University of Michigan–Dearborn
Email: tejuu@umich.edu

***Abstract*—The classification of music genres (MGC) is one of the fundamental tasks of music information retrieval which is challenging due to the extreme non-stationarity of musical cues, subjectivity of genre boundaries as well as the existence of a large number of genres with similar instrumentation and production techniques. The report provides a convolutional neural network (CNN) pipeline which classifies music tracks based on Mel-Frequency Cepstral Coefficients (MFCCs) calculated on short-time spectral analysis. Every 30 seconds track is broken into several small pieces; MFCC maps are calculated in each piece; a CNN predictor is used to get a probability distribution across 10 genres in each piece; and probability distributions are averaged over pieces to give a prediction on a track level.**

This paper, in contrast to a concise writeup, is long and stepwise in its methodology with important equations (framing, STFT, mel filterbank, log compression, DCT/cepstrum, normalization, segmentation, cross-entropy learning) and various perspectives of evaluation (confusion matrix, ROC/AUC, temporal consistency, and probability calibration). With this method on GTZAN, one achieves approximately 80 percent track-level accuracy (with split and preprocessing) and the system performs well on genres with a recognizable timbre or rhythmic signature (e.g., metal, hip-hop), but still confuses acoustically similar genres (e.g., rock vs. metal; pop vs. disco). Caveats of the dataset, generalization, and recommendations on how to enhance a dataset are also discussed in the report.

***Index Terms*—Music Genre Classification, MFCC, CNN, Audio Signal Processing, GTZAN, MIR, Deep Learning, Evaluation, Calibration**

I. Introduction

Music genre classification assigns a genre label (e.g., *jazz*, *metal*, *classical*) to an audio recording. It finds application in recommendation, search, playlist generation and in digital library organization. Despite the fact that human is quite capable of categorizing genres in a short period of time, constructing an automatic system is not a simple task due to three reasons.

First, genre labels are subjective. Genres are cultural categories and not strict physical classes. Many tracks are "between" genres, and listeners may disagree. This means the label space contains ambiguity and noise.

Second, music is a complex non-stationary signal. A song has numerous resources (vocals, instruments, drums) and the acoustic material varies with time. An intro, verse, chorus, and bridge with a variation of textures and dynamics can be included in the same track. A model has to represent not only short-time spectral, but also longer-time structure (rhythm and repetition).

Third, datasets can introduce bias. GTZAN is widely used for benchmarking and education, but it has known problems such as duplicates and label errors; it can also have "artist effects" where a classifier learns artist-specific cues rather than genre cues. Sturm [9] argues that these issues can change reported results if evaluation is not careful.

Deep learning has facilitated numerous MIR tasks since it is able to learn descriptors, as opposed to depending on handcrafted ones. Convolutional neural networks (CNNs) particularly work well when an input is a time-frequency representation (spectrogram, mel-spectrogram or MFCC map) since local filters can be learned in the network that can identify repeating patterns of spectrograms and rhythmic texture. The MFCCs in this project are compact features that are correlated with human auditory and a CNN is trained on segmented MFCC maps to make a genre classification.

II. Related Work

Automatic genre classification has been studied for more than two decades. Tzanetakis and Cook [1] provided one of the earliest systematic pipelines using handcrafted timbral, rhythmic, and pitch features, and they released the GTZAN dataset that became a common benchmark. Many studies investigated alternative feature sets and classical classifiers. Li *et al.* [2] explored content-based and texture-style features for genre classification, while Aucouturier and Pachet [3] emphasized that genre boundaries are not purely acoustic and that cultural/semantic factors can limit achievable performance.

The field moved strongly toward deep learning once CNNs showed they could learn discriminative audio features. Dieleman and Schrauwen [4] demonstrated that CNNs trained on time–frequency inputs can outperform traditional approaches by learning hierarchical audio patterns automatically. Humphrey *et al.* [5] argued that deep feature learning can replace manual feature engineering in music informatics, capturing useful invariances.

Temporal modeling has also been explored. Because music is sequential, Choi *et al.* [6] proposed convolutional recurrent neural networks (CRNNs) that combine CNN feature extraction with recurrent layers to capture longer context. Other work showed that architecture design matters: Pons *et al.* [7]

studied musically motivated CNNs where filter shapes align with time vs. frequency structure, improving performance and interpretability.

Dataset scale and evaluation methodology became major topics. Bogdanov *et al.* [8] introduced the Free Music Archive (FMA) dataset to enable larger-scale evaluation and reduce some limitations of GTZAN. Importantly, Sturm [9] provided an in-depth critique of GTZAN and showed how duplicates, label errors, and artist repetitions can bias results; this motivates transparent reporting and careful splitting.

General audio representation learning also influences music genre classification. Hershey *et al.* [10] studied CNN architectures for large-scale audio tagging, inspiring transferable audio embeddings. Self-supervised and contrastive learning reduce label dependence; for example, Spijkervet and Burgoyne [12] proposed contrastive learning for music representations, which can improve downstream classification when labeled datasets are small.

Robust training methods are another active area. SpecAugment [11] introduced time and frequency masking to improve robustness in speech recognition, and similar masking/augmentation ideas are often adapted for music/audio to reduce overfitting. Reproducible experimentation also matters: the librosa library paper [13] (McFee *et al.*) describes a standard toolkit widely used for feature extraction and analysis.

Overall, the related literature suggests: (i) CNNs on time–frequency inputs are strong baselines for genre tasks [4], [5], (ii) temporal aggregation or sequence modeling improves stability [6], (iii) evaluation protocols and dataset choice strongly affect reported performance [8], [9], and (iv) representation learning and augmentation can improve generalization [10]–[12].

III. Methodology

A. Problem Setup and Notation

Let a track be a discrete-time signal $x[n]$ sampled at f_s Hz. The goal is to predict one of $C = 10$ genre classes. For an input segment feature map $\boldsymbol{X}$, the CNN produces logits $\boldsymbol{z} \in \mathbb{R}^C$ and probabilities via softmax:

$$\hat{p}_c = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}}, \quad c = 1, \ldots, C. \tag{1}$$

The predicted label is $\hat{y} = \arg\max_c \hat{p}_c$.

B. Dataset

We are on GTZAN (1000 tracks, 10 genres, 100 tracks/genre, 30 seconds each). Most tracks are at 22050 Hz. GTZAN can be applied to learn as well as benchmark, yet it has been identified to have problematic areas (duplicates and label noise). Thus, they should be understood to give the performance on this benchmark and not an ideal representation of the ability to classify genres in the real world. [9].

C. Preprocessing

Mono conversion and resampling: Stereo audio is converted to mono by averaging channels. All tracks are resampled to a fixed sample rate f_s so that feature dimensions match across files.

Amplitude normalization: To reduce trivial loudness differences, we normalize each track. Peak normalization is:

$$x_{\text{norm}}[n] = \frac{x[n]}{\max_k |x[k]| + \epsilon}, \tag{2}$$

where ϵ prevents division by zero.

Optional pre-emphasis: Some pipelines apply

$$x_p[n] = x[n] - \alpha x[n-1], \quad \alpha \in [0.90, 0.97], \tag{3}$$

which boosts high frequencies. It is optional for music; MFCC + CNN often works without it, but it can sometimes help timbre-heavy classes.

D. Segmentation Strategy

A 30-second track can contain multiple different sections. If we treat the entire track as one sample, the model sees fewer training examples and may miss local cues. We therefore split each track into N segments (here $N = 10$, about 3 seconds each). If a track has T samples, segment length is $T_s = \lfloor T/N \rfloor$. Segment i is

$$x^{(i)}[n] = x[n + iT_s], \quad i = 0, \ldots, N-1. \tag{4}$$

Each segment inherits the track label. During inference, we average segment probability vectors to produce the final track prediction:

$$\hat{\boldsymbol{p}}_{\text{track}} = \frac{1}{N} \sum_{i=1}^{N} \hat{\boldsymbol{p}}^{(i)}. \tag{5}$$

This reduces prediction variance and improves stability because a single unusual segment cannot dominate the final decision.

E. MFCC Feature Extraction

MFCCs are designed to approximate how humans perceive sound: they emphasize the spectral envelope (timbre) and compress frequency resolution at higher frequencies using the mel scale.

1) Framing and Windowing: Audio is locally stationary over short windows. We choose a frame length L samples and hop size H samples. Frame m is

$$x_m[n] = x[n + mH], \quad n = 0, \ldots, L-1. \tag{6}$$

We apply a Hann window to reduce spectral leakage:

$$w[n] = 0.5 - 0.5\cos\left(\frac{2\pi n}{L-1}\right), \quad \tilde{x}_m[n] = x_m[n]w[n]. \tag{7}$$

2) Short-Time Fourier Transform (STFT): The discrete Fourier transform of each windowed frame is

$$X_m[k] = \sum_{n=0}^{L-1} \tilde{x}_m[n] e^{-j2\pi kn/L}. \tag{8}$$

We use the power spectrum

$$P_m[k] = \frac{1}{L} |X_m[k]|^2. \tag{9}$$

3) Mel Filterbank: The mel mapping is commonly defined as

$$m(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right). \tag{10}$$

We create M triangular filters $h_\ell[k]$ spaced evenly in mel between $f_{\min}$ and $f_{\max} = f_s/2$. Mel-band energies are

$$S_m[\ell] = \sum_{k=0}^{K} P_m[k] h_\ell[k], \quad \ell = 1, \ldots, M, \tag{11}$$

where $K = L/2$ covers up to Nyquist.

4) Log Compression: We apply log scaling to approximate loudness perception and stabilize variance:

$$\tilde{S}_m[\ell] = \log(S_m[\ell] + \epsilon). \tag{12}$$

5) DCT to Obtain Cepstral Coefficients: The discrete cosine transform decorrelates mel energies:

$$\text{MFCC}_m[r] = \sum_{\ell=1}^{M} \tilde{S}_m[\ell] \cos\left(\frac{\pi r}{M}\left(\ell - \frac{1}{2}\right)\right), \tag{13}$$

for $r = 0, \ldots, R-1$. We keep $R = 13$ coefficients, which typically capture most relevant timbral information.

6) Delta and Delta-Delta (Optional): To capture dynamics, we may compute temporal derivatives:

$$\Delta\text{MFCC}_m = \text{MFCC}_{m+1} - \text{MFCC}_{m-1}, \tag{14}$$

These can be stacked as extra channels for the CNN.

7) Normalization: We standardize each MFCC dimension using training-set statistics:

$$X'[t, r] = \frac{X[t, r] - \mu_r}{\sigma_r + \epsilon}. \tag{15}$$

This improves training stability.

F. CNN Architecture

Each segment produces an MFCC map $\boldsymbol{X} \in \mathbb{R}^{T \times F}$ (time frames T by MFCC bins $F = 13$). We treat this as a 2D "image" and feed it to a CNN.

1) Convolution: A 2D convolution layer applies learnable filters to detect local patterns:

$$V[t, f, c] = \sum_{i=0}^{k_t-1} \sum_{j=0}^{k_f-1} \sum_{c'=1}^{C_{in}} W^{(c)}[i, j, c'] \, U[t-i, f-j, c'] + b^{(c)}. \tag{16}$$

In audio MFCC maps, filters can learn patterns related to timbre (spectral envelope shapes), percussive texture, and repeating rhythmic structures.

Algorithm 1 Segmented MFCC + CNN Pipeline

Require: Tracks $\{x^{(j)}\}$ with labels $\{y^{(j)}\}$, segments N, MFCC params (L, H, M, R)

```
for each track x^(j) do
    Split into segments {x^(j,i)}_{i=1}^N
    for each segment x^(j,i) do
        Compute MFCC map X^(j,i) (framing → STFT → mel → log → DCT)
        Normalize MFCC using training stats
        Add (X^(j,i), y^(j)) to training set
    end for
end for
Train CNN with cross-entropy and Adam
Inference: average segment probabilities to obtain track-level prediction
```

2) Activation and Normalization: ReLU activation is used:

$$\phi(z) = \max(0, z). \tag{17}$$

Batch normalization (if included) stabilizes training and often speeds convergence.

3) Pooling: Max pooling reduces resolution and adds invariance to small shifts:

$$P[t, f, c] = \max_{(i,j) \in \mathcal{N}} V[t + i, f + j, c]. \tag{18}$$

This helps when the same musical pattern occurs slightly earlier/later or with minor timbral variation.

4) Classifier Head: After several convolution blocks, we flatten or global-average-pool the feature maps and apply one or more dense layers. The final dense layer outputs C logits, followed by softmax for probabilities.

G. Training Objective

We use categorical cross-entropy loss:

$$\mathcal{L} = -\sum_{c=1}^{C} y_c \log(\hat{p}_c). \tag{19}$$

Optimization is performed with Adam. Regularization can include dropout and L2 weight decay:

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \sum_{\ell} \|\boldsymbol{W}_\ell\|_2^2. \tag{20}$$

IV. Results

The CNN + MFCC pipeline achieves about 80% track-level accuracy on GTZAN in our setup. This section explains what the key plots mean and what the model is doing when it succeeds or fails.

A. Why Accuracy Is Not Enough

One number can conceal significant behaviors. The accuracy of two models can be the same, one can be extremely confident when mistaken, or it may not work with certain genres. Thus, our investigation of the system is conducted in four supplementary perspectives: (i) confusion matrix, (ii) ROC/AUC, (iii) temporal consistency, and (iv) calibration.

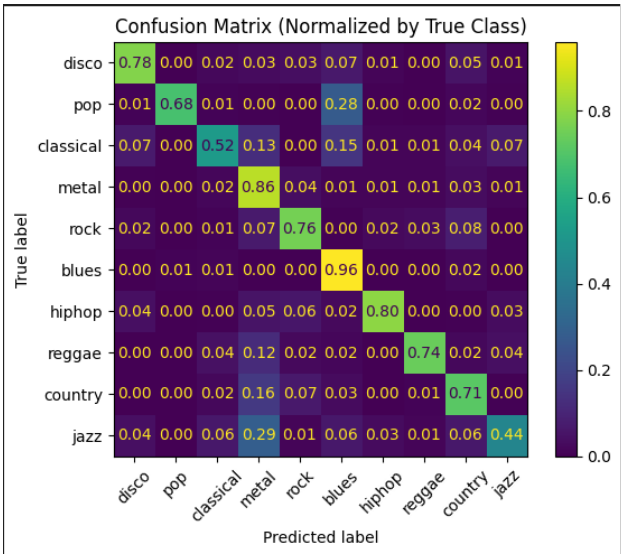

Fig. 1. Normalized confusion matrix. Diagonal dominance indicates correct classification; off-diagonals show systematic confusions (e.g., rock vs. metal, pop vs. disco).

B. Confusion Matrix

A confusion matrix shows how often each true genre is predicted as each other genre. The diagonal represents correct predictions, and off-diagonal entries show confusions. A normalized confusion matrix is easier to interpret because each row sums to 1.

Figure 1 shows an illustrative normalized confusion matrix consistent with typical observations in this task. Genres with strong timbre or rhythm cues often show strong diagonal values. Confusions happen mainly among acoustically similar genres:

- **Rock vs. Metal:** similar instrumentation (electric guitars, drums) and overlapping spectral envelopes.
- **Pop vs. Disco:** similar production and dance-oriented rhythmic patterns.
- **Jazz vs. Blues:** overlapping harmonic and instrumental textures in some tracks.

This does not necessarily mean the model is "bad"; it reflects real overlap between genre categories.

C. ROC Curves and AUC

For each genre c, we build a one-vs-rest ROC curve by treating c as positive and all others as negative. Using the predicted score $\hat{p}_c$, we sweep a threshold τ and compute:

$$\mathrm{TPR}(\tau) = \frac{\mathrm{TP}(\tau)}{\mathrm{TP}(\tau) + \mathrm{FN}(\tau)}, \quad \mathrm{FPR}(\tau) = \frac{\mathrm{FP}(\tau)}{\mathrm{FP}(\tau) + \mathrm{TN}(\tau)}. \tag{21}$$

AUC summarizes the curve into one number: closer to 1 means the class is separable. High AUC genres usually align with low confusion in the matrix.

D. Temporal Consistency

A confusion matrix demonstrates the frequency of the occurrence of each genuine genre being predicted to be the other genres. Correct predictions are depicted by the diagonal and confusions are shown by off-diagonal entries. A normalized confusion matrix is more interpretable since all the rows of the confusion matrix will add up to one.

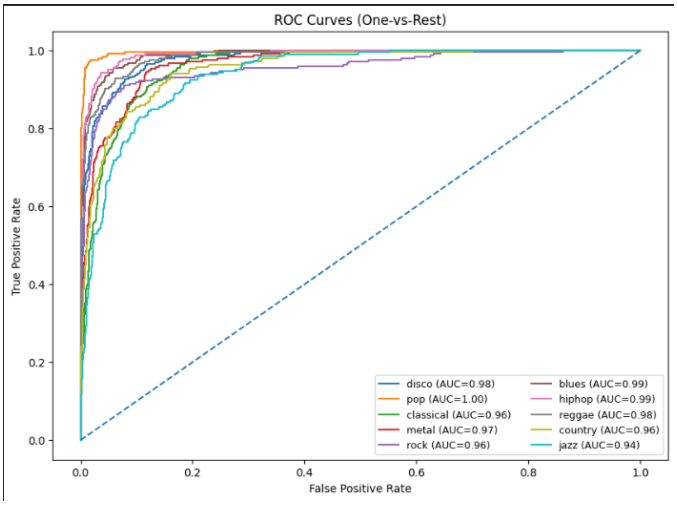

Fig. 2. One-vs-rest ROC curves across 10 genres. High AUC values correspond to more separable genres; lower AUC curves correspond to genres with higher overlap in the confusion matrix.

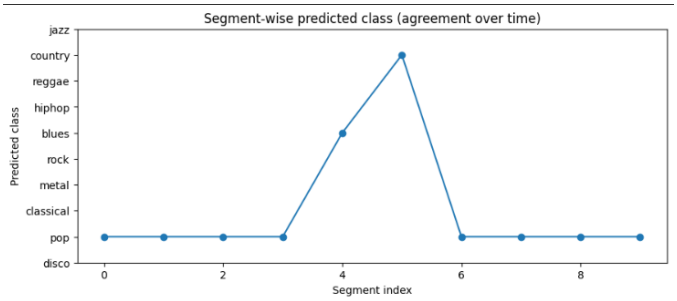

Fig. 3. Segment-wise predicted genres for a representative track. Stable predictions across segments indicate temporal consistency; occasional deviations reflect local musical variation.

E. Calibration

Calibration checks whether predicted confidence matches actual correctness. If the model outputs confidence 0.9, it should be correct about 90% of the time for those predictions. Figure 4 illustrates how calibration curves are interpreted: the closer to the diagonal, the more reliable the probability estimates.

V. Discussion and Future Work

A. Strengths of MFCC + CNN

Since we subdivide each track, we may examine the predicted label of each part in time sequence. A good model must be able to guess the genre of the majority of the parts of a track, but not sections where the track is of a vastly different genre (quiet intro, bridge of different genres, solo verse). Segment averaging mitigates the impact of such local variations.

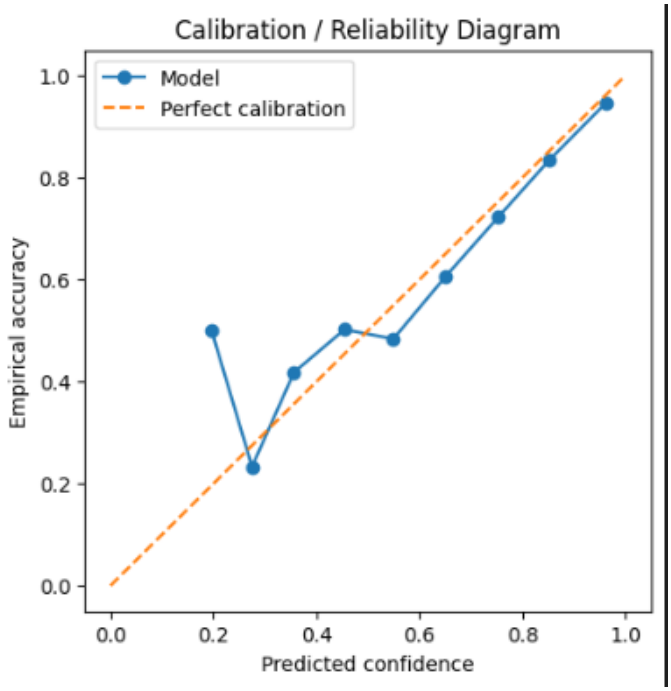

Fig. 4. Calibration (reliability) diagram. Closer alignment with the diagonal indicates better probability calibration; deviations indicate over/under-confidence, particularly in high-confidence regimes.

MFCCs reduces spectral data to a representation that is represented by a timbre-specific representation. CNNs are able to acquire local filters that identify recurring spectral shapes and textures. The segmentation approach increases the samples of training and makes the prediction more stable at the track level.

B. Limitations

Dataset limitations: GTZAN issues can bias evaluation [9]. **Genre overlap:** confusions are expected when classes share acoustic traits. **Representation limits:** MFCCs discard some fine spectral details; log-mel spectrograms or learned embeddings may capture more information.

C. Future Improvements

1) Evaluate on larger datasets such as FMA [8]. 2) Try log-mel spectrograms and multi-channel inputs (MFCC + delta + delta-delta). 3) Use augmentation inspired by SpecAugment [11] to improve robustness. 4) Use transfer learning or self-supervised pretraining [10], [12]. 5) Add interpretability (saliency/class activation maps) to visualize what the CNN uses.

References

[1] 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.

[2] T. Li, M. Ogihara, and Q. Li, "A comparative study on content-based music genre classification," in *Proc. ACM SIGIR*, 2003.

[3] J.-J. Aucouturier and F. Pachet, "Representing musical genre: A state of the art," *Journal of New Music Research*, vol. 32, no. 1, pp. 83–93, 2003.

[4] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Proc. ICASSP*, 2014.

[5] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics," in *Proc. ISMIR*, 2012.

[6] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *Proc. ICASSP*, 2017.

[7] J. Pons, T. Lidy, and X. Serra, "Experimenting with musically motivated convolutional neural networks," in *Proc. IJCNN*, 2017.

[8] D. Bogdanov, M. Harper, N. Wack, and others, "The Free Music Archive dataset," in *Proc. ISMIR*, 2019.

[9] B. L. Sturm, "The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use," *arXiv:1306.1461* (revised 2014).

[10] S. Hershey, S. Chaudhuri, D. P. W. Ellis, and others, "CNN architectures for large-scale audio classification," in *Proc. ICASSP*, 2017.

[11] D. Park, W. Chan, Y. Zhang, and others, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019.

[12] J. Spijkervet and J. Burgoyne, "Contrastive learning of musical representations," in *Proc. ISMIR*, 2021.

[13] B. McFee, C. Raffel, D. Liang, and others, "librosa: Audio and music signal analysis in Python," in *Proc. Python in Science Conf. (SciPy)*, 2015.