Enhancing Music Tagging Accuracy: A CNN-Based Approach

Sanjeev Ghosh1, a, Payel Saha1, b, Manoj Chavan1, c and Deepak S. Shete1, d

**Abstract.** Accurate classification of unlabeled music, automatic music tagging can significantly enhance search recommendations. This work explores the effects on tagging accuracy of combining several Convolutional Neural Network (CNN) models, each trained on separate audio features. Often leveraging spectrogram-based characteristics including MFCC, Mel spectrograms, and STFT, CNNs are extensively used for the classification of musical genres. Applying the Librosa library to extract Mel spectrograms, MFCCs, and STFT spectrograms, the paper makes use of the 1000 songs across 10 genres GTZAN dataset. Using GPU resources on Kaggle to speed training, these characteristics are used to train separate CNN models with TensorFlow and Keras. Using ensemble learning via majority voting, the predictions of the models are aggregated to increase general accuracy. With one reaching 94.71% accuracy, the MFCC-based models proved to be the best among them. Models learnt on Mel and standard spectrograms, however, were more likely to overfit. Combining all models let the ensemble system achieve a rather better accuracy of 97.88%. While some genres, including Rock and Disco, were sometimes misclassified, the ensemble model always exceeded the individual models. This reveals that combining several CNN models increases resilience and tagging accuracy. Presenting a possible scalable, real-time genre classification technique for the digital music market, the paper also underlines the generalizability and stability of MFCC-based models. The ensemble helps basic models overcome their weaknesses and capitalize on their strengths. Mel/Spectrograms' various features and MFCCs' robustness provide strong generalization across datasets and reduce variance and overfitting in standalone models. Mel Spectrogram and Spectrogram variations had testing accuracies of 68–71% and struggled with genre overlaps, but the ensemble model had 97.89%. Its F1-scores above 0.90 for 9 of 10 genres showed its balanced precision and recall across a wide range of musical styles.

# Introduction

The Internet has become the most important source of information for all kinds of media, from text to music, to videos. Music plays a very important role in our lives and is abundantly available on the internet, offering a wide range of choices to users for their listening. Music in the digital form on the internet has seen an exponential rise over the last few years and has led to the development of efficient and accurate music discovery and organization mechanisms. The Internet search engines are very good at retrieving information based on text. Using text-based queries to search for music is not efficient as music is an auditory medium. As a way out, music information has to be augmented with metadata like genre, mood, theme, artist, instruments, etc. Music Information Retrieval (MIR), which includes music tagging and retrieval has seen a transition from manual annotation and classification techniques to more sophisticated models relying on machine learning and deep learning. Initially, music tagging and retrieval was carried out with the help of rule-based systems and metadata annotations. Human experts were entrusted the task of manually labelling music tracks based on genre, mood or instruments. With the exponential increase in the size of the digital music libraries, manual tagging became next to impossible. This lead to the development of software based feature extraction techniques that used signal processing. These systems used spectral analysis, mel-frequency cepstral coefficients (MFCCs), and chroma features to classify music tracks [1]. Over the years, significant improvement in computational power and development of machine learning and deep learning models have led to a revolution in this field with newer scalable, efficient and high accuracy music tagging and retrieval systems.

Automatic music tagging is the process of attaching relevant labels such as genre, mood or instrument to musical tracks. This step is vital for the development of efficient MIR, personalisation and recommendation systems. The availability of accurate tags or labels (metadata) with the clean music tracks determines the effectiveness of these recommendation systems. Studies have shown that Convolutional Neural Networks (CNNs) have been successfully used in other domains like computer vision and natural language processing [2],[3]. This has led to the use of these architectures being for developing MIR systems. The use of deep learning has helped in surpassing the previous performance of music tagging systems. More and more MIR researchers have started proposing models based on deep learning architectures. As a result, the previously used manual feature based approaches have been replaced by machine learning and deep learning based approaches in most recent automatic music tagging research.

Regarding the evolution of MIR systems, it has been found that advance in computational techniques have made a significant contribution to the enhancement in the performance of music tagging and retrieval systems. In the traditional systems, i.e., the ones where music tagging involved manually annotating music tracks by experts, it was found that scalability and adaptability were major challenges. MIR research has seen significant advances with the advent of machine learning and deep learning techniques. However, with the advent of deep learning, MIR research has made significant strides, shifting toward data-driven feature learning approaches that can automatically extract meaningful patterns from vast amounts of audio data. MIR is not just significant for individual users who want to retrieve their preferred song or music, but also a revolution for the music industry in the present scenario. Music streaming apps can use the information in the metadata to find the latest trending music, make personalized playlists and also give users exactly what they are looking for. Finding tracks that relate to different moods, genres or emotions becomes much simpler with the help of MIR [4].

This work explores the effects on tagging accuracy of combining several Convolutional Neural Network (CNN) models, each trained on separate audio features. Often leveraging spectrogram-based characteristics including MFCC, Mel spectrograms, and STFT, CNNs are extensively used for the classification of musical genres. Applying the Librosa library to extract Mel spectrograms, MFCCs, and STFT spectrograms, the paper makes use of the 1000 songs across 10 genres GTZAN dataset. Using GPU resources on Kaggle to speed training, these characteristics are used to train separate CNN models with TensorFlow and Keras. Using ensemble learning via majority voting, the predictions of the models are aggregated to increase general accuracy. In the field of audio signal processing and machine learning, study on music genre classification and tagging has been especially important. Several approaches have been suggested to handle these challenges including low-level and high-level feature extraction methods as well as classification models.

## 1.1 Feature Extraction Techniques

Low-level audio features including energy, loudness, timbre, tempo, and spectral characteristics have been extensively applied for music genre classification [5], [6]. These characteristics faithfully reflect the basic acoustic qualities of music. High-level features including Mel-Frequency Cepstral Coefficients (MFCC), Constant-Q Transform, and Short-Time Fourier Transform (STFT) provide representations that capture more complex patterns and structures in music, so fitting for genre classification and tagging [7].

## 1.2 Classification Models

Originally intended for image processing, Convolutions Neural Networks (CNNs), have found use in the classification of musical genres. Mel spectrograms and other spectrograms are handled as images, which lets CNNs identify music depending on patterns found in these visual depictions [8]. Deep convolutional neural networks (CNNs) have been embraced in the field of computer vision since their capacity to replicate the functioning of the human visual system. These networks can develop hierarchical characteristics that allow them to show object localization invariance and model [9] distortion and translation resilience. For similar reasons, CNNs have also found use in audio-related tasks; they exhibit innovative performance in tasks including music segmentation [11] and speech recognition [10]. Support Vector Machine (SVM): Low-level features have been used with non-linear kernel equipped SVMs to best classify music genres [12]. Higher-dimensional and non-linear data especially fit SVMs, which help to segment music into several genre "blobs" on a graph.

## 1.3 Aggregation at Sample Levels

One strategy to guarantee relevant tag predictions is to take samples from the song and aggregate the tags produced for every sample since the defining traits of a genre or mood may not last throughout an entire song [8]. This method raises the strength of genre categorization.

## 1.4 Generation of Metadata and Tagging in Music

Automatic music tagging is the study of music to find patterns and project its genre and other characteristics. This system helps with music classification and organization as well as improves user search recommendations [7]. Usually using audio processing methods for feature extraction, music tagging systems then train a tagging model using convolutional neural networks (CNNs [7]. Websites can be used to host these models so users may enter songs and get expected tags as result. Practical applications like real-time music genre classification and tagging depend on improvements in computational efficiency, which especially in the framework of music streaming services depends on [13]. Combining several models, ensemble learning presents a potential path for even more accuracy and robustness enhancement of music genre classification and tagging systems [14].

# Methodology

Python programming language was used to create the complete end-to-end application which takes the audio file as input and gives the genre tag as output. To implement a Music tagging system, the methodology followed is given below:

## 2.1. Data Collection, Pre-processing, and Feature Extraction

The music dataset should be a diverse collection of music spanning different genres, including both popular songs and instrumental pieces. The pre-tagging of each song in the dataset was ensured with relevant genre labels. The GTZAN dataset was used for training the data. It consists of selected pieces of music that correspond to a particular style or genre. The GTZAN dataset appears in at least 100 published works and is the most-used public dataset for evaluation in machine listening research for music genre recognition (MGR) [15]. The audio files consist of pieces from popular songs from the genre as well as pieces of instrumental music that represent the genre. The GTZAN dataset, contains 10 different music genres, each represented by 100 songs, totaling 1000 songs in the dataset. The dataset can be considered a comprehensive one that can be used to make a model that will classify most of them accurately.

Python language has been used to pre-process the audio data. This involves tasks such as audio file loading, feature extraction, and normalization. Librosa, a library that is written in C++ and provides support for music and audio data analysis, has also been used [16]. It provides various functions that help with information retrieval from music streams. Since the dataset lacks exclusive validation samples, an 80-20 train-test split is performed using the ‘train\_test\_split’ function from the ‘sklearn’ library. This ensures a robust validation process. The training and validation split has to be done before any other mathematical operation to prevent data leakage which may lead to inaccurate validation. The validation data remains untouched for later evaluation. Librosa's feature extraction functions are utilized to create feature arrays for each audio file. These values are then converted to a dB scale and normalized to generate spectrograms for various feature extraction techniques.

Spectrograms as features can be given as inputs to the CNN model for the classification and tagging of music files [17], [18]. A set of relevant features have been extracted from the audio data, including the Short Time Fourier Transform (STFT) spectrogram, Mel frequency spectrogram, and Mel frequency Cepstral Coefficient (MFCC) spectrogram CNN can be effectively used to extract patterns from this data and assign relevant tags. The major advantage that CNN provides is that it can learn and capture both local and global patterns from the data. It can identify short-term patterns in the music as well as long-term patterns in the music which can then be associated with the different tags.

## 2.2. Model Selection and Training

For training the model the first most important step is to check the file format. Audio files are checked to ensure they are in WAV format, as the Librosa library used for audio processing can only read this format. Each audio file is tagged with its respective genre, and the genre information is saved in a data frame for later use. Convolutional Neural Networks (CNNs) have been used as the primary model for music genre classification [19]. CNNs are well-suited for processing spectrogram data [20], [21]. Multiple CNN models have been created in the proposed work, each tailored to the specific input shape of the extracted features (e.g., STFT, Mel spectrogram, MFCC). Convolutional Neural Network (CNN) models are created using TensorFlow and a library it contains called Keras. The model architectures are designed according to the specific input feature type (e.g., STFT, Mel spectrogram, MFCC).

The experimentation was done with various model architectures, hyperparameters, and training strategies to achieve the best performance. The GPU resources were utilised through Kaggle Notebooks, to expedite training and reduce computational time. Kaggle Notebooks is an online, notebook-based cloud environment that provides computing resources based on a subscription pricing model. For the work, the platform was used for GPU resources, as training the model is faster when using GPU, and using it based on subscription also reduces the overall cost of buying a GPU. Ensemble learning is a machine-learning technique that involves combining multiple models to improve the accuracy and robustness of predictions. The idea behind ensemble learning is that a group of diverse models when combined, can achieve better performance than any individual model [22]. Ensemble learning is beneficial because it helps to reduce the impact of overfitting and bias in individual models. In the proposed work, ensemble learning techniques have been implemented to combine the predictions of multiple CNN models trained on different features. Ensemble models can enhance prediction accuracy and robustness. Depending on the input feature (e.g., STFT, Mel spectrogram, MFCC), the CNN model architecture is adapted accordingly. The architecture should be tailored to maximize performance for each feature type.

**TABLE 1**. ARCHITECTURAL PARAMETERS FOR MEL SPECTROGRAM MODEL (SEQUENTIAL)

|  |  |  |
| --- | --- | --- |
| Layer (type) | Output Shape | Param # |
| conv2d (Conv2D) | (None, 128, 1293, 8) | 80 |
| max\_pooling2d (MaxPooling2D) | (None, 32, 324, 8) | 0 |
| conv2d\_1 (Conv2D) | (None, 32, 324, 16) | 1168 |
| max\_pooling2d\_1 (MaxPooling2D) | (None, 8, 81, 16) | 0 |
| conv2d\_2 (Conv2D) | (None, 8, 81, 32) | 4640 |
| max\_pooling2d\_2 (MaxPooling2D) | (None, 2, 21, 32) | 0 |
| conv2d\_3 (Conv2D) | (None, 2, 21, 64) | 18496 |
| max\_pooling2d\_3 (MaxPooling2D) | (None, 1, 6, 64) | 0 |
| conv2d\_4 (Conv2D) | (None, 1, 6, 64) | 36928 |
| max\_pooling2d\_4 (MaxPooling2D) | (None, 1, 2, 64) | 0 |
| flatten (Flatten) | (None, 128) | 0 |
| dense (Dense) | (None, 64) | 8256 |
| dense\_1 (Dense) | (None, 10) | 650 |
| **Total params:** 70,218  **Trainable params:** 70,218  **Non-trainable params:** 0 | | |

Table 1 shows the model architecture and the number of parameters in the CNN model for training on the Mel spectrogram. The Mel spectrogram is created by first applying STFT to the audio signal and then transforming the frequency bins to corresponding bins in the Mel scale. The size of the array obtained after applying the Mel transform is 128\*1293 where 128 is the number of Mel bins and 1293 is the number of time bins. The values of amplitude are converted to dB value and the values are normalized so that the data is in a readable form and can be processed easily by the CNN model. The model consists of 5 convolution layers and 5 pooling layers ending with a dense network having 2 layers with 128 and 64 inputs and the last layer is a SoftMax function with 10 outputs corresponding to the 10 genres. Each convolution layer is followed by a max pooling layer. The first convolution layer gives 8 feature maps as output of size 3\*3 which are passed to a max pooling layer of size 4\*4. A similar arrangement is followed for the next 4 times but with the number of feature maps changed to 16, 32, 64, and 64 in the given order. The activation function ReLU is used for all the convolution layers and dense layers. The total number of parameters for this architecture is 70,218 parameters.

***TABLE 2****. ARCHITECTURAL PARAMETERS FOR MFCC MODEL (SEQUENTIAL)*

|  |  |  |
| --- | --- | --- |
| Layer (type) | Output Shape | Param # |
| conv2d (Conv2D) | (None, 120, 600, 16) | 160 |
| batch\_normalization (BatchNorm) | (None, 120, 600, 16) | 64 |
| max\_pooling2d (MaxPooling2D) | (None, 30, 100, 16) | 0 |
| conv2d\_1 (Conv2D) | (None, 30, 100, 32) | 4640 |
| batch\_normalization\_1 (BatchNorm) | (None, 30, 100, 32) | 128 |
| max\_pooling2d\_1 (MaxPooling2D) | (None, 8, 17, 32) | 0 |
| conv2d\_2 (Conv2D) | (None, 8, 17, 64) | 18496 |
| batch\_normalization\_2 (BatchNorm) | (None, 8, 17, 64) | 256 |
| max\_pooling2d\_2 (MaxPooling2D) | (None, 2, 3, 64) | 0 |
| flatten (Flatten) | (None, 384) | 0 |
| dense (Dense) | (None, 256) | 98560 |
| dense\_1 (Dense) | (None, 64) | 16448 |
| dense\_2 (Dense) | (None, 10) | 650 |
| **Total parameters:** 139,402  **Trainable parameters:** 139,178 **Non-trainable parameters:** 224 | | |

Table 2 shows the architecture of the CNN model used for training on MFCC data. The MFCC spectrogram is created by first applying STFT to the signal and transforming to the logarithmic Mel scale. This step is similar to creating a Mel spectrogram, but an additional step is involved for MFCC where cosine transform has to be applied to the signal as the last step. For this project, 10 coefficients are used and a 2048 size window is used for FFT which gives an output shape of 10\*1293 where 1293 is the number of frames. Since the output has only 10 rows it is a very inconvenient size to apply convolution filters as there will be a large reduction in size at every step compared to the columns. Therefore, the spectrogram is resized so that the representation of audio is the same but with an image size that is convenient to pass through a CNN. The first convolution layer receives an input of the spectrogram with shape 600\*120 and gives an output of 16 feature maps created by using filters of size 3\*3. The convolution layer is followed by a batch normalization and a max pooling layer of size 4\*6. This arrangement is repeated 2 more times to give 32 and 64 feature maps in the given order. After the last max pooling layer is a dense network with 3 layers of sizes 256, 64, and 10 where the last layer is a SoftMax function. The total number of trainable parameters for the model is 139,178 and there are 224 nontrainable parameters from the batch normalization steps.

Table 3 shows the model architecture used for training on the spectrogram. The spectrogram is a visual representation of the frequencies in time frames of an audio sample. The spectrogram is created by first applying STFT to the audio signal. The power values are converted to the dB scale and the values are normalized. The model has a series of 5 convolution and max pooling layers. The feature maps generated in each layer are 8, 16, 32, 64, and 64 in the given order where 3\*3 filters are used for convolution. Window size 4\*4 is used for max pooling layers. A dense network is added after the last max pooling with 2 layers of input size 128 and 64 followed by SoftMax function of size 10. Total number of trainable parameters for this model is 103,114.

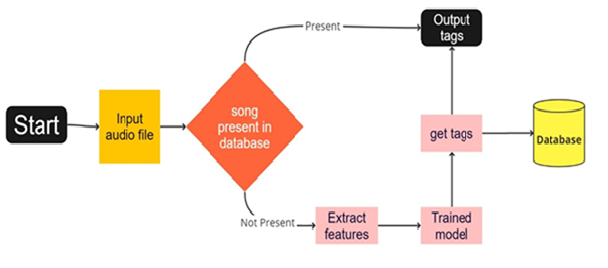
***TABLE 3****. ARCHITECTURAL PARAMETERS FOR SPECTROGRAM MODEL (SEQUENTIAL)*

|  |  |  |
| --- | --- | --- |
| Layer (type) | Output Shape | Param # |
| conv2d (Conv2D) | (None, 1025, 1293, 8) | 80 |
| max\_pooling2d (MaxPooling2D) | (None, 257, 324, 8) | 0 |
| conv2d\_1 (Conv2D) | (None, 257, 324, 16) | 1168 |
| max\_pooling2d\_1 (MaxPooling2D) | (None, 65, 81, 16) | 0 |
| conv2d\_2 (Conv2D) | (None, 65, 81, 32) | 4640 |
| max\_pooling2d\_2 (MaxPooling2D) | (None, 17, 21, 32) | 0 |
| conv2d\_3 (Conv2D) | (None, 17, 21, 64) | 18496 |
| max\_pooling2d\_3 (MaxPooling2D) | (None, 5, 6, 64) | 0 |
| conv2d\_4 (Conv2D) | (None, 5, 6, 64) | 36928 |
| max\_pooling2d\_4 (MaxPooling2D) | (None, 2, 2, 64) | 0 |
| flatten (Flatten) | (None, 256) | 0 |
| dense (Dense) | (None, 128) | 32896 |
| dense\_1 (Dense) | (None, 64) | 8256 |
| dense\_2 (Dense) | (None, 10) | 650 |
| **Total parameters:** 103,114  **Trainable parameters:** 103,114  **Non-trainable parameters: 0** | | |

The CNN models are trained on the training dataset using Tensor Flow's training function. Parameters such as the number of epochs and other hyperparameters are optimized through experimentation to achieve high training accuracy while avoiding overfitting [23]. After training, the performance of each model is evaluated using the validation dataset. Predictions are made on the validation songs, and accuracy scores are calculated. If overfitting is detected (i.e., high training accuracy but low validation accuracy), model adjustments are made iteratively until satisfactory results are obtained. Once a satisfactory model is achieved, it is saved as a file for later use in making predictions. For making predictions, the same feature extraction and normalization techniques are applied to generate feature spectrograms for audio files. There is no data splitting since predictions are made for all input audio files. The extracted features are passed through the corresponding saved models, and predictions are recorded for each model. An ensemble model is created by taking a majority vote among all the predictions, ensuring a robust final prediction.

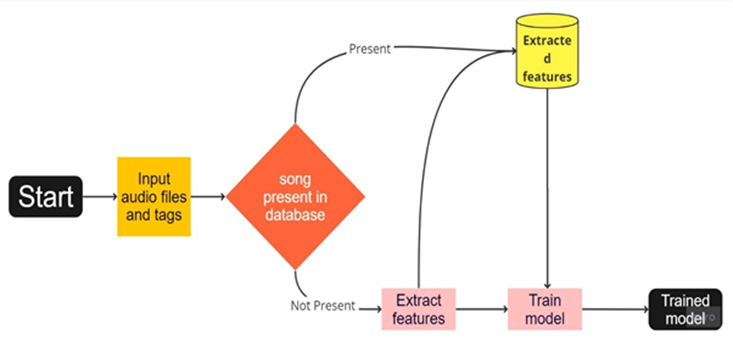
## 2.3. Tagging Model and Database

After feature extraction, the extracted features are fed into the trained CNN models for prediction. The generated tags and corresponding music files are then stored in a database for efficient retrieval and to reduce processing time for recurring queries [24]. Periodically the model needs to be updated using new audio files and their corresponding tags. This retraining process can help the model adapt to evolving music trends and improve its accuracy over time. The methodology can be summarized as follows:



***FIGURE 1****. SYSTEM FOR TAGGING APPLICATION*

The users can interact with the tagging model through an interface, as shown in Figure 1, that takes the music file as the input and gives the output as tags corresponding to the input. When the application gets the input, the audio file first goes through the pre-processing stage where all the feature extraction operations are applied, and different features are generated that can be passed to the model. The extracted features are then passed to model that had been trained earlier and the model makes the classifications based on the features to generate relevant tags. The music and the generated tags are stored in a separate database that consists of the name of the songs and their generated tags. This step allows to reduce the time taken for generating tags by retrieving generated tags from the database for a query on the same music that has been encountered earlier.

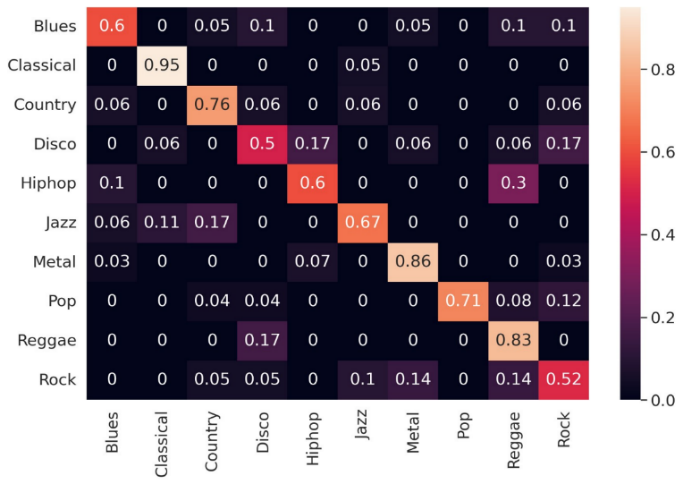


***FIGURE 2.*** *SYSTEM FOR MODEL TRAINING*

For training the prediction model required to generate tags for given music, the system will take input as audio files and corresponding tags as shown in Figure 2. The input consists of the audio files and a CSV file having the tags for each audio file. The pre-processing of the audio files will be done to extract the different features and these features will be stored in a separate database having the song name, features, and tags. Storing the extracted features in a database will reduce the cost of computation required during retraining as feature extraction will not have to be performed again on all the songs. Using the extracted features and tags from the database, the neural network model is trained to find the best parameters. The parameters are saved for the model giving the best performance so that the model can be used again easily when making predictions on new songs.

# Results and Discussion

Using a pre-processing process akin to that used on the training data, the system guarantees that the audio fragment is converted into numerical data fit for the CNN algorithm. Longer songs are broken out into 30-second chunks since the models are made to manage audio clips of up to 30 seconds in length. The system generates therefore (song length/30 seconds) \* 6 outputs. One uses an ensemble technique to get at a final estimate. This method aggregates the model outputs to find among the segments the most often occurring genre label. After that, the expected genre tag shows on the website so that users may enter later audio files for genre prediction.



**FIGURE 3.** CONFUSION MATRIX FOR MEL SPECTROGRAM MODEL

Figure 3 shows the confusion matrix for the model educated on the Mel spectrogram feature's predictions. Especially with regard to accuracy for every genre category, the matrix offers understanding of the classification performance of the model. Some noteworthy notes include: blues, disco, rock, and hip-hop Sort: These categories show rather less accuracy. For example, 12% of songs categorized as Rock are classified as Pop, and almost 17% of them as Disco. One theory for this could be the impact of Rock music on Disco and Pop genres, which resulted in tempo and beat similarities. Sometimes Blues is found as Disco, Reggae, and Rock, suggesting difficulties separating these styles.

|  |  |
| --- | --- |
| A picture containing text, screenshot, square, number  Description automatically generated | A picture containing text, screenshot, square, number  Description automatically generated |
| (a) | (b) |

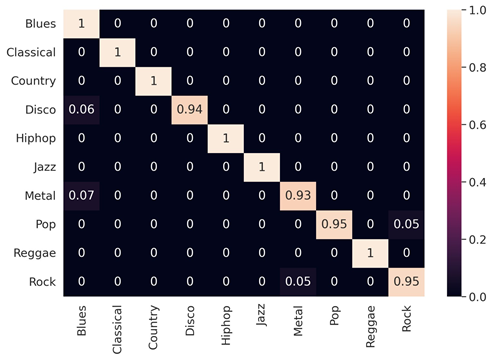
**FIGURE 4.** (A) CONFUSION MATRIX FOR MFCC MODEL 1 AND (B) CONFUSION MATRIX FOR MFCC MODEL 2

Figure 4(a) and 4(b) show confusion matrices for MFCC models. The model has accurate classifications with a low number of misclassifications in all categories except Classical, where the accuracy is 100%.

|  |  |
| --- | --- |
|  | A picture containing text, screenshot, number  Description automatically generated |
| (a) | (b) |

**FIGURE 5**. (A) CONFUSION MATRIX FOR SPECTROGRAM MODEL 1 AND (B) CONFUSION MATRIX FOR SPECTROGRAM MODEL 2

Figure 5(a) and Figure 5(b) show the confusion matrices for the models trained on the spectrogram. The misclassifications are similar to that of the Mel spectrogram model where Rock has been the most misclassified as Metal, Disco, Blues, and Country. The classifications are spread out in all categories and no specific pattern can be observed.



**FIGURE 6**. CONFUSION MATRIX FOR ENSEMBLE MODEL

Figure 6 shows the confusion matrix for the predictions of the ensemble of all models. The ensemble model gives the best confusion matrix compared to the individual models. Most of the categories are predicted correctly except for the categories of Rock, Pop, Metal, and Disco as they are inspired by other forms which may have led to minor inaccuracies. Based on the results it can be said the ensemble model can tag the genres correctly.

Table 4 lists the accuracies of every single model as well as the ensemble one. It is clear from comparison that the ensemble model performs better than the individual ones. Extremely high training accuracies (~100%), Mel Spectrogram (Row 1) and Spectrogram Models (Rows 5 & 6) show greatly lower testing accuracies (~68–71%). This discrepancy points to overfitting: the models fail to generalize to unseen data even while they memorize training data rather effectively. High model complexity, lack of regularizing and inadequate data augmentation or variation in training data are most likely the causes of this. With just minor variation between the two, MFCC Models (Rows 2–4) have training and testing accuracy in the 91–94% range. With 94.04% training and 94.71% testing, MFCC Model 3 stands out as indicating outstanding generalization and most likely the best single model. Here MFCCs seem to produce more stable models and are a generally trusted feature set for speech/audio tasks. Much above all individual models, the ensemble model achieves 98.68% training and 97.89% testing accuracy. This implies that the ensemble minimizes the shortcomings of the base models and combines strengths rather successfully. The combination of Mel/Spectrogram feature diversity and MFCC-based robustness by the ensemble It achieves strong generalizing over datasets and lowers variance and overfitting from individual models.

***TABLE 4.*** *MODEL ACCURACIES*

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Technique | Training Accuracy | Testing Accuracy |
| 1 | Mel Spectrogram | 100.00 % | 70.90 % |
| 2 | MFCC Model 1 | 93.64 % | 93.65 % |
| 3 | MFCC Model 2 | 93.51 % | 91.53 % |
| 4 | MFCC Model 3 | 94.04 % | 94.71 % |
| 5 | Spectrogram Model 1 | 99.87 % | 69.84 % |
| 6 | Spectrogram Model 2 | 100.00 % | 68.25 % |
| Total Accuracy (Ensemble) | | 98.68 % | 97.88 % |

To gain deeper insights beyond simple accuracy, we evaluated each model using class-wise Precision, Recall, and F1-Score across all 10 music genres. These metrics offer a more nuanced understanding of how well each model performs in identifying both common and less frequent genres. While accuracy reflects overall correctness, precision measures the reliability of positive predictions, recall captures the model's sensitivity to true positives, and F1-score balances both precision and recall. This detailed evaluation helps identify specific genre-level strengths and weaknesses across models. It also highlights cases where models may overfit or underperform due to genre overlaps or feature limitations. Comparing these metrics enables a fairer assessment of model generalization and robustness, particularly for complex, overlapping genres like Rock and Disco. Precision, recall, and F1-scores for all genres across the models were computed and are presented in Table 5, Table 6, and Table 7, respectively.

***TABLE 5.*** *GENRE-WISE PRECISION COMPARISON ACROSS MODELS*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Genre | Mel  Spectogram | MFCC-1 | MFCC-2 | MFCC-3 | Spec-1 | Spec-2 | Ensemble |
| Blues | 0.71 | 0.88 | 0.86 | 0.69 | 0.69 | 0.70 | 0.90 |
| Classical | 0.95 | 1.00 | 1.00 | 0.64 | 0.64 | 0.76 | 1.00 |
| Country | 0.85 | 1.00 | 0.77 | 0.67 | 0.67 | 0.68 | 1.00 |
| Disco | 0.71 | 0.88 | 0.62 | 0.60 | 0.60 | 0.50 | 0.90 |
| HipHop | 0.72 | 0.87 | 0.85 | 0.92 | 0.92 | 0.59 | 0.89 |
| Jazz | 0.74 | 1.00 | 0.80 | 0.95 | 0.50 | 0.65 | 0.97 |
| Metal | 0.95 | 0.86 | 0.85 | 0.89 | 0.89 | 0.74 | 0.92 |
| Pop | 0.70 | 0.95 | 0.82 | 0.60 | 0.60 | 0.70 | 0.93 |
| Reggae | 0.86 | 0.96 | 0.78 | 0.76 | 0.76 | 0.59 | 0.95 |
| Rock | 0.57 | 0.90 | 0.63 | 0.46 | 0.46 | 0.44 | 0.89 |

Table 5 shows that the precision values are very different between models and genres. However, MFCC Model 1 always has the highest precision in most categories. For example, the genres Classical, Country, and Jazz all had perfect or almost perfect precision (1.00, 1.00, and 1.00, respectively) in MFCC-1, which means that the predictions were very reliable and there were very few false positives. Mel Spectrogram and raw Spectrogram models struggled to match this performance, especially for Disco, Rock, and Jazz, when precision values decreased below 0.44. Similar tempos and instruments led these models to combine genres. After integrating predictions and resolving errors in weaker models, the ensemble model improved accuracy in tough genres like Rock (0.89), Disco (0.90), and HipHop (0.89). Ensemble voting eliminates unconvincing forecasts, improving accuracy across genres.

Recall values emphasized the sensitivity of each model in accurately identifying true instances of each genre. Table 6 demonstrates that MFCC-1 excelled in recall, attaining perfect scores in genres including Classical, Country, and Jazz, along with high scores in Rock (0.95) and Reggae (0.96).Mel and Spectrogram models, on the other hand, showed lower recall, especially for genres with overlapping auditory characteristics

Recall values highlighted the sensitivity of each model in correctly identifying true instances of each genre. As seen in Table 6, MFCC-1 once again led in recall, achieving perfect scores in genres like Classical, Country, and Jazz, and very high scores in others such as Rock (0.95) and Reggae (0.96). On the other hand, Mel and Spectrogram models exhibited lower recall, especially for genres with overlapping acoustic features. For example, the Mel model's Rock recall was only 0.52, and Spectrogram-2's Disco recall was only 0.41; this shows that these models often failed to find correct genre names. The ensemble method fixed these problems by getting memory scores of more than 0.88 for all types of music, even those that had low recall scores before. This shows that the ensemble can improve sensitivity by combining the best features of base models and reducing false negatives. This is especially useful in genres that are hard to consistently record.

***TABLE 6.*** *GENRE-WISE RECALL COMPARISON ACROSS MODELS*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Genre | Mel  Spectogram | MFCC-1 | MFCC-2 | MFCC-3 | Spec-1 | Spec-2 | Ensemble |
| Blues | 0.60 | 0.94 | 0.78 | 0.84 | 0.84 | 0.78 | 0.95 |
| Classical | 0.95 | 1.00 | 1.00 | 0.87 | 0.87 | 1.00 | 1.00 |
| Country | 0.76 | 1.00 | 0.65 | 0.71 | 0.71 | 0.65 | 1.00 |
| Disco | 0.50 | 0.88 | 0.41 | 0.47 | 0.47 | 0.41 | 0.88 |
| HipHop | 0.60 | 0.82 | 0.64 | 0.93 | 0.93 | 0.64 | 0.88 |
| Jazz | 0.67 | 1.00 | 0.71 | 0.53 | 0.53 | 0.71 | 0.96 |
| Metal | 0.86 | 0.80 | 0.79 | 0.95 | 0.95 | 0.79 | 0.91 |
| Pop | 0.71 | 0.95 | 0.67 | 0.60 | 0.60 | 0.67 | 0.92 |
| Reggae | 0.83 | 0.96 | 0.63 | 0.73 | 0.73 | 0.63 | 0.94 |
| Rock | 0.52 | 0.95 | 0.44 | 0.46 | 0.46 | 0.44 | 0.91 |

The F1-score, balancing precision and recall, offers a thorough assessment of each model's effectiveness in genre classification. Table 7 presents the F1 scores for all genres across the models. Among all models, MFCC Model 1 consistently attained the highest F1-scores across genres, achieving perfect scores (1.00) for Classical, Country, and Jazz, which indicates a robust capacity for accurate genre identification and prediction. The Mel Spectrogram and Spectrogram models exhibited suboptimal performance for overlapping genres such as Rock, Disco, and Pop, with F1-scores falling below 0.60 in multiple instances, thereby underscoring their limitations in managing stylistically similar categories. The ensemble model enhanced overall F1-scores, reaching values exceeding 0.90 for 9 of the 10 genres and demonstrating substantial improvements in difficult genres like Rock and Disco. This signifies that ensemble learning effectively leverages the strengths of individual models while mitigating their weaknesses. The ensemble model delivers the most reliable and equitable categorization, achieving values exceeding 0.90 for 9 out of 10 genres and demonstrating significant improvements in difficult genres such as Rock and Disco. This indicates that ensemble learning efficiently utilizes the advantages of individual models while mitigating their deficiencies. Consequently, the ensemble model provides the most dependable and equitable categorization performance across the genre spectrum.

***TABLE 7.*** *GENRE-WISE F1 SCORE COMPARISON ACROSS MODELS*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Genre | Mel  Spectogram | MFCC-1 | MFCC-2 | MFCC-3 | Spec-1 | Spec-2 | Ensemble |
| Blues | 0.65 | 0.91 | 0.82 | 0.76 | 0.76 | 0.74 | 0.92 |
| Classical | 0.95 | 1.00 | 1.00 | 0.74 | 0.74 | 0.86 | 1.00 |
| Country | 0.80 | 1.00 | 0.70 | 0.69 | 0.69 | 0.67 | 1.00 |
| Disco | 0.59 | 0.88 | 0.49 | 0.53 | 0.53 | 0.45 | 0.89 |
| HipHop | 0.65 | 0.84 | 0.73 | 0.93 | 0.93 | 0.61 | 0.89 |
| Jazz | 0.70 | 1.00 | 0.75 | 0.51 | 0.51 | 0.68 | 0.96 |
| Metal | 0.90 | 0.83 | 0.82 | 0.92 | 0.92 | 0.76 | 0.91 |
| Pop | 0.70 | 0.95 | 0.74 | 0.60 | 0.60 | 0.68 | 0.92 |
| Reggae | 0.84 | 0.96 | 0.70 | 0.74 | 0.74 | 0.61 | 0.94 |
| Rock | 0.54 | 0.92 | 0.52 | 0.46 | 0.46 | 0.44 | 0.90 |

MFCC Model 1 displayed exceptional per-genre performance, particularly for genres like Classical, Country, and Jazz, where it achieved perfect precision, recall, and F1-scores (1.00), reflecting its strong generalization and low false positive/negative rates. In contrast, Spectrogram-based models showed inconsistent behavior, especially struggling with Rock, Disco, and Jazz, where both precision and recall were significantly lower, leading to correspondingly poor F1-scores. The Mel Spectrogram model also exhibited difficulty in reliably classifying genres such as Rock, Disco, and Blues, likely due to genre overlaps and rhythmic similarities. Notably, Rock and Disco proved consistently challenging across models, often being misclassified due to hybrid subgenres like Pop-Rock and Disco-Rock. However, the ensemble model significantly improved both precision and recall for these difficult classes, leading to F1-scores above 0.90 for 9 out of 10 genres. This demonstrates the ensemble model’s ability to maintain a strong balance between sensitivity (recall) and prediction confidence (precision).

From the standpoint of both overall accuracy and granular evaluation metrics, the ensemble model stands out as the most robust and generalizable. It outperforms all individual models not just in aggregate performance, but more importantly, in genre-specific reliability—particularly where individual models falter. Mel and Spectrogram feature diversity and MFCC temporal and spectral resolution give the ensemble a richer and more discriminative representation. It also reduces overfitting, which plagued the Mel and Spectrogram models, which had great training accuracy but poor testing performance. For real-world genre classification, the ensemble approach is accurate, consistent, and dependable.

# Conclusion and Future Scope

The work presents a music tagging system based on given audio snippets that can forecast music genre labels. Two main components defined the approach: model prediction applying the CNN algorithm and audio file pre-processing. For model training, the GTZAN dataset has been used and for audio file pre-processing, the Librosa library has been used to get at this. TensorFlow was essential in guiding the proposed models across the CNN network, and an interactive website was built using Flask so users may enter songs and get genre predictions. Though high training accuracy, spectrogram and mel spectrum models suffer from overfitting, reliable and more balanced performance comes from MFCC-based models. MFCC Model 1 scored 1.00 for Classical, Country, and Jazz in a precision, recall, and F1-score evaluation across all 10 genres. Mel and Spectrogram models had trouble with genre overlaps, notably for Rock and Disco, reducing precision and recall. The ensemble model outperformed individual models by improving F1-scores for 9 genres, balancing sensitivity and specificity, and correcting class-wise anomalies in base models. The value of model diversity and ensemble learning in audio classification problems is shown by the ensemble model, which uses all models and offers the best general accuracy. The results obtained show that the ensemble model shows good performance in several music genres, proving its usefulness in the classification of different musical genres. While our current implementation uses the GTZAN dataset, we fully acknowledge its limitations and have to incorporate more diverse and contemporary datasets in our future work. Although the adopted approach shows encouraging outcomes, it is important to recognize the fast changing character of the music business. Regular new musical trends arise, and as musicians explore fresh sounds and techniques, the very definition of music genres changes. Future research could therefore include maintaining the accuracy of the proposed model by keeping the dataset current with the newest songs, so addressing this. The tag count could be increased to include themes, moods, instruments, etc., transcending mere genres. Organizing the songs according to vocalists and language will increase the capacity of the model. Song recommendation could be added to this system to offer customized recommendations depending on user preferences and past performance.

# References

1. Priyanshu Rawat, Madhvan Bajaj, Satvik Vats, Vikrant Sharma, “A Comprehensive Study based on MFCC and Spectogram for Audio Classification” Journal of Information and Optimization sciences, 44(6):1057-1074, DOI: 10.47974/JIOS-1431
2. Zhao, X., Wang, L., Zhang, Y. *et al.* A review of convolutional neural networks in computer vision. *Artif Intell Rev* **57**, 99 (2024). https://doi.org/10.1007/s10462-024-10721-6
3. Won, M., Ferraro, A., Bogdanov, D., & Serra, X. (2020, June) “**Evaluation of CNN‑based Automatic Music Tagging Models”** In Proceedings of the 17th Sound and Music Computing Conference (SMC 2020).
4. Strauss, H., Vigl, J., Jacobsen, PO. *et al.* The Emotion-to-Music Mapping Atlas (EMMA): A systematically organized online database of emotionally evocative music excerpts. *Behav Res* **56**, 3560–3577 (2024). https://doi.org/10.3758/s13428-024-02336-0
5. C. C. M. Yeh, J. C. Wang, Y. H. Yang, and H. M. Wang, "Improving music auto-tagging by intra-song instance bagging," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, 2014. doi: 10.1109/ICASSP.2014.6853977.
6. A. Das, R. Bhai, S. Sachdev, T. Anand, and U. Kumar, “Automatic tagging of songs using machine learning,” International Journal of Database Theory and Application, Vol. 9, No. 5, pp. 129-140, 2016. http://dx.doi.org/10.14257/ijdta.2016.9.5.13
7. K. Choi, G. Fazekas, and M. Sandler, "Automatic tagging using deep convolutional neural networks," in 17th International Society of Music Information Retrieval Conference (ISMIR 2016), New York City, USA, 2016. https://doi.org/10.48550/arXiv.1606.00298.
8. M. Won, A. Ferraro, D. Bogdanov, and X. Serra, "Evaluation of CNN-based automatic music tagging models," in 17th Sound and Music Computing Conference (SMC2020), Torino, Italy, 2020. https://doi.org/10.48550/arXiv.2006.00751.
9. Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in M. A. Arbib, Ed., The Handbook of Brain Theory and Neural Networks. MIT Press, 1995.
10. T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 2013. doi: 10.1109/ICASSP.2013.6639347.
11. K. Ullrich, J. Schluter, and T. Grill, "Boundary detection in music structure analysis using convolutional neural networks," in Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, 2014.
12. C. Xu, N. C. Maddage, X. Shao, F. Cao, and Q. Tian, "Musical genre classification using support vector machines," in Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '03, Hong Kong, 2003. doi: 10.1109/ICASSP.2003.1199998.
13. C. Zhang and Y. Zhang, "SongNet: Real-time music classification," 2018. Available at: https://cs229.stanford.edu/proj2018/report/53.pdf. Accessed on 21 December 2023.
14. M. Chaudhury, A. Karami, and M. A. Ghazanfar, "Large-scale music genre analysis and classification using machine learning with Apache Spark," Electronics, vol. 11, no. 16, p. 2567, 2022. https://doi.org/10.3390/electronics11162567.
15. B. L. Sturm, "The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use," arXiv, 2013. Available: https://doi.org/10.48550/arXiv.1306.1461.
16. B. McFee, C. Raffel, D. Liang, and D. P. Ellis, "Librosa: Audio and music signal analysis in Python," in Proceedings of the 14th Python in Science Conference, 2015, pp. 18-24. Available: https://conference.scipy.org/proceedings/scipy2015/pdfs/brian\_mcfee.pdf.
17. C. Senac et al., "Music feature maps with convolutional neural networks for music genre classification," in Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing, 2017, pp. 1-5. https://doi.org/10.1145/3095713.3095733.
18. K. M. Athulya and S. Sindhu, "Deep learning based music genre classification using spectrogram," Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021, 2021. Available: http://dx.doi.org/10.2139/ssrn.3883911
19. Pattanaik, R. K., & Jaiswal, A. (2024), “Classification of music genre using support vector machine and convolutional neural network”inWorld Journal of Advanced Research and Reviews, 21(03), 1009–1019. https://doi.org/10.30574/wjarr.2024.21.3.0802
20. Prasanna, S. R. M., Hota, R. N., Ramesh, A., & Kumar, S. (2019), “Braided Convolutional Neural Network for Audio Signal Classification” in IET Signal Processing, 13(7), 627–633. <https://doi.org/10.1049/iet-spr.2019.0381>
21. Zheng, H., Zhang, Z., & Song, Y. (2018). A Feature Fusion Method Based on CNN for Acoustic Scene Classification. arXiv preprint arXiv:1809.01543. <https://arxiv.org/abs/1809.01543>
22. Ganaie, M. A., Hu, M., Malik, A. K., Tanveer, M., & Suganthan, P. N. (2021), “Ensemble deep learning: A review”, inEngineering Applications of Artificial Intelligence*,* 115*,* 105151. https://doi.org/10.1016/j.engappai.2022.105151
23. Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Becker, M. (2021), “Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges” **WIREs Data Mining and Knowledge Discovery**, 13(e1484). https://doi.org/10.1002/widm.1484.
24. **Xu, Y., Kong, Q., Huang, Q., Wang, W., & Plumbley, M. D. (2017), “**Convolutional Gated Recurrent Neural Network Incorporating Spatial Features for Audio Tagging”, **In 2017 IEEE International Joint Conference on Neural Networks (IJCNN)** (pp. 341–348). IEEE. https://doi.org/10.1109/IJCNN.2017.7965890*.*