Music Genre Classification using GTZAN Dataset

Siddhartha Kaushik
M.Eng. Software Engg., ECE,
University of Western Ontario

London, ON, Canada
skaush9@uwo.ca

James Zhong
M.Eng. Software Engg., ECE,
University of Western Ontario

London, ON, Canada
pzhong22@uwo.ca

Harshit Kamboj
M.Eng. Software Engg., ECE,
University of Western Ontario

London, ON, Canada
hkamboj6@uwo.ca

Syed Mohammad Zeeshan Hyder
M.Eng. Software Engg., ECE,
University of Western Ontario

London, ON, Canada
shyder26@uwo.ca

***Abstract*— This report presents a comprehensive study on Music Genre Classification utilizing the GTZAN dataset, employing four distinct machine learning models: Random Forest, XGBoost, Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN). The aim of the project is to explore the effectiveness of these models in classifying music genres based on audio features, with a focus on both raw audio data and Mel spectrograms. Each model is evaluated based on various performance metrics including accuracy, precision, recall, and F1 score. The analysis encompasses the comparison of these metrics across the models, shedding light on their strengths and weaknesses in handling music genre classification tasks.**

The study provides insights into the suitability of different machine learning approaches for music genre classification, contributing to the understanding of the application of these algorithms in audio processing tasks.

Keywords—Music Genre Classification, XGBoost, Random forest, LSTM, CNN, GTZAN.

I. Introduction

Music genre classification plays a pivotal role in various applications such as music recommendation systems, content organization, and personalized user experiences. With the advent of machine learning and deep learning techniques, researchers have explored innovative approaches to automate the process of genre classification using audio data.

In this report, we delve into a comprehensive study on music genre classification utilizing the GTZAN dataset, a widely recognized benchmark dataset in the field of music information retrieval. Our study focuses on four distinct machine learning models: Random Forest, XGBoost, Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN). These models are chosen for their diverse architectures and capabilities in handling different types of data, ranging from raw audio samples to spectrogram representations.

The primary objective of this project is to evaluate and compare the performance of these models in classifying music genres, considering evaluation metrics such as accuracy, precision, recall, and F1 score. By systematically assessing the strengths and limitations of each model, we aim to provide valuable insights into the effectiveness of various machine learning approaches for music genre classification tasks.

Furthermore, our study contributes to the broader field of music information retrieval by offering implications for the development of robust and scalable music classification systems. The findings presented in this report can inform the design of advanced music recommendation algorithms, content tagging mechanisms, and automated genre labelling tools, ultimately enhancing the user experience in digital music platforms.

Through this investigation, we seek to advance the understanding of machine learning techniques applied to audio processing tasks, with a specific focus on music genre classification. By leveraging the capabilities of modern machine learning algorithms, we aim to address the challenges and opportunities in automating the classification of music genres, paving the way for future advancements in this exciting domain.

II. Background

A. GTZAN Dataset

The dataset comprises a diverse collection of 10 music genres, each containing 100 audio files, each lasting 30 seconds. It is reminiscent of the GTZAN dataset, often dubbed the MNIST of sounds. Additionally, the dataset includes visual representations for each audio file, created as Mel Spectrograms to facilitate classification using neural networks, particularly Convolutional Neural Networks (CNNs). Furthermore, there are two CSV files containing audio file features, one computed for each 30-second song and another for 3-second segments of the same songs, effectively increasing the data volume for classification models. The GTZAN dataset is a pivotal resource in machine listening research for music genre recognition, inspires this dataset's design and evaluation criteria.

B. Random Forest

Random Forest is an ensemble learning method used for classification and regression tasks. It operates by constructing multiple decision trees during training and outputs the mode of the classes or mean prediction of the individual trees. Random

forests correct for decision trees' habit of overfitting to their training set.

Random Forest can handle high dimensional datasets well and can manage both numerical and categorical data. For music genre classification, where features might include a variety of signal characteristics such as tempo, beat, rhythm, and melody extracted from audio files, Random Forest can efficiently handle such complexity. Its ensemble nature makes it robust against overfitting, often resulting in high accuracy.

C. XGBoost

XGBoost (Extreme Gradient Boosting) is an implementation of gradient boosted decision trees designed for speed and performance. It is a scalable and accurate implementation of gradient boosting machines, which are used for supervised learning tasks.

XGBoost is known for its performance and speed, especially in structured or tabular datasets. It can handle sparse data and is effective in dealing with a wide range of data types, making it suitable for music genre classification where the dataset contains various feature types extracted from audio signals. Its ability to automatically handle missing data and its use of regularization to prevent overfitting can lead to superior predictive performance.

D. Long-Short Term Memory (LSTM)

LSTM networks are a special kind of Recurrent Neural Network capable of learning long-term dependencies. They are explicitly designed to avoid the long-term dependency problem, remembering information for long periods as part of their default behavior.

LSTMs are particularly suited for classification tasks where temporal dynamics are crucial, such as in time-series data or sequence data. Music tracks, which are essentially time-series data, contain temporal structures and patterns unique to each genre. LSTMs can learn these patterns over time, making them particularly effective for capturing the temporal characteristics of music for genre classification.

E. Convolution Neural Network (CNN)

CNNs are a class of deep neural networks, most commonly applied to analyzing visual imagery. They use a mathematical operation called convolution in at least one of their layers. CNNs are known for their ability to automatically and adaptively learn spatial hierarchies of features from images.

While traditionally used for image data, CNNs can also be applied to audio data by converting audio signals into spectrograms, which are visual representations of the spectrum of frequencies in a sound or other signal as they vary with time. This transformation allows the CNN to treat audio data similarly to image data, capturing patterns and features like rhythms, beats, and harmonies that are distinctive to different music genres. Their ability to learn local patterns makes CNNs exceptionally good at capturing the essence of music genres from spectrograms or other audio-derived visual representations.

III. Methodology

Each member of the group trained one of the four models and generated results to compare.

A. Random Forest

1) Preprocessing and Model

The StandardScaler from sklearn.preprocessing is used to standardize features by removing the mean and scaling to unit variance. The LabelEncoder is used to encode target labels with value between 0 and n_classes-1. The dataset is partitioned into training and test subsets, utilizing the "train_test_split" function with a 33% allocation for testing and a defined random state to ensure consistent results.

The "RandomForestClassifier" model is configured using Tensorflow with 100 trees (estimators) and a set random state to guarantee the reproducibility of results.

2) Hyperparameter Tuning

The model underwent hyperparameter tuning with a grid search over a combination of 216 different parameter sets, across 5-fold cross-validation, resulting in a total of 1080 fits.

Parameter	Value
bootstrap	False
max_depth	30
min_samples_leaf	1
min_samples_split	2
n_estimators	300

B. XGBoost

1) Preprocessing and Model

The data preprocessing step is identical to that of the Random Forest. The "XGBClassifier" is initialized with specific settings to disable the use of label encoder and to use 'logloss' as the evaluation metric, aiming to circumvent deprecation notices and align with current best practices.

2) Hyperparameter Tuning

The model was subject to hyperparameter tuning over 243 candidate parameter sets, with 5-fold cross-validation, totaling 1215 fits.

Parameters	Value
colsample_bytree	0.5
learning_rate	0.2
max_depth	5
n_estimators	300
subsample	1

C. Long-Short Term Memory (LSTM)

1) LSTM Model Design

Two LSTM models were designed: one to handle features from 30-second clips and another for features from 3-second clips. The 30-second clip model was designed with a look-back of 1, considering each 30-second clip as a single sequence. For the 3-second clips, a look-back of 10 was used to treat ten consecutive 3-second clips as one sequence, increasing the data granularity. Both models included dropout layers to prevent overfitting and dense layers for classification.

2) Hyperparameter Tuning

Keras Tuner was utilized to fine-tune the LSTM models, focusing on the number of LSTM units, dropout rate, and the optimizer's learning rate. The tuning process aimed to identify the optimal configuration that maximizes classification accuracy.

3) Early Stopping

To mitigate overfitting and optimize training time, we implemented an early stopping mechanism in our LSTM model training. This technique halts training when the validation loss doesn't improve for a set number of epochs, indicated by the "patience" parameter. By monitoring validation loss, early stopping ensures the model doesn't learn noise from the training data, which could degrade performance on unseen data. It effectively balances learning and computational efficiency, ceasing training at an optimal juncture to maintain model generalization capabilities.

D. Convolution Neural Network (CNN)

1) Dataset and Preprocessing

The Mel Spectrogram representation of the audio files are stored as PNG format images for 10 different classes in separate directories. The dimensions of the input images are 432x288, with a batch size of 32. The ImageDataGenerator class from Keras is used to load and preprocess images. Data augmentation techniques such as normalization and horizontal flipping are applied to increase the diversity of the training dataset, which helps improve model generalization. The flow_from_directory method of ImageDataGenerator is used to load images from the directory and split them into training and validation sets with 20% of the data used for validation.

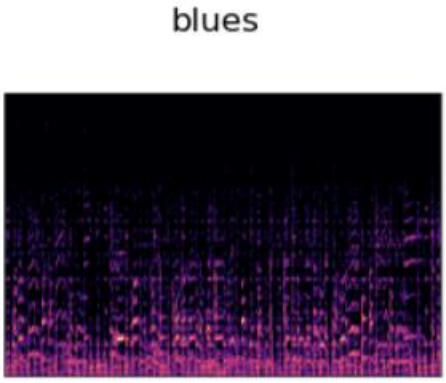

Fig. 1. Example of a Mel Spectrogram image generated from audio file

2) CNN model

The CNN model consists of several convolutional layers followed by max-pooling layers. Each convolutional layer extracts features from the input images using filters (kernels) of size 3x3 with ReLU activation. Max-pooling layers down sample the feature maps obtained from convolutional layers, reducing the spatial dimensions. The final layers of the model are densely connected (fully connected) layers, which perform classification based on the extracted features. The output layer consists of 10 neurons with softmax activation, representing the probability distribution over the 10 classes. Below is the CNN configuration.

Layer	Output Shape	Kernel	Stride
conv2d_1 (Conv2D)	(None, 430, 286, 64)	3x3	1x1
max_pooling2d_1 (MaxPooling2D)	(None, 215, 143, 64)	2x2	2x2
conv2d_2 (Conv2D)	(None, 213, 141, 128)	3x3	1x1
max_pooling2d_2 (MaxPooling2D)	(None, 106, 70, 128)	2x2	2x2
conv2d_3 (Conv2D)	(None, 104, 68, 256)	3x3	1x1
max_pooling2d_3 (MaxPooling2D)	(None, 52, 34, 256)	2x2	2x2
conv2d_4 (Conv2D)	(None, 50, 32, 256)	3x3	1x1
max_pooling2d_4 (MaxPooling2D)	(None, 25, 16, 256)	2x2	2x2
conv2d_5 (Conv2D)	(None, 23, 14, 512)	3x3	1x1
max_pooling2d_5 (MaxPooling2D)	(None, 11, 7, 512)	2x2	2x2
flatten (Flatten)	(None, 39424)	-	-
dense (Dense)	(None, 256)	-	-
dense_1 (Dense)	(None, 128)	-	-
dense_2 (Dense)	(None, 64)	-	-
dense_3 (Dense)	(None, 10)	-	-

3) Hybrid CNN model

The hybrid model built upon the above model by adding data from ‘features_30_sec.csv’ file into the dense layer trying to classify the genre more effectively. Below is the configuration difference between Hybrid and simple CNN.

flatten (Flatten)	(None, 39424)	-	-
dense (Dense)	(None, 256)	-	-
Input_layer_1	(None, 57)	-	-
Concatenate	(None, 313)	-	-
dense_1 (Dense)	(None, 128)	-	-
dense_2 (Dense)	(None, 64)	-	-
dense_3 (Dense)	(None, 10)	-	-

4) Model compilation and hyperparameter selection

The model is compiled using the Adam optimizer and categorical cross-entropy loss function, which is suitable for multi-class classification problems. Accuracy is chosen as the

evaluation metric to monitor during training. Early stopping and learning rate reduction callbacks are applied to monitor the validation loss and adjust the learning rate dynamically during training, preventing overfitting and speeding up convergence.

IV. Results

A. Random Forest

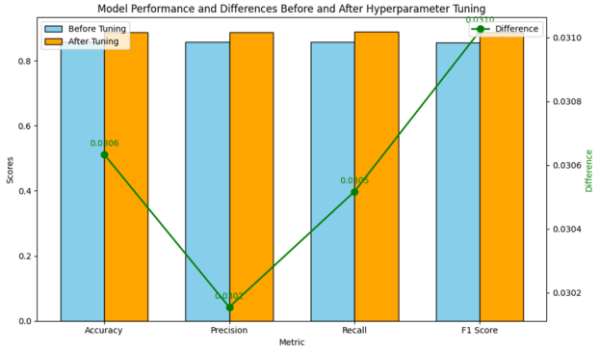

Fig. 2. Ramdon Forest result

	Before Hyperparameter Tuning	After Hyperparameter Tuning
Accuracy	0.8571	0.8877
Precision	0.8572	0.8873
Recall	0.8576	0.8881
F1 Score	0.8559	0.8869

B. XGBoost

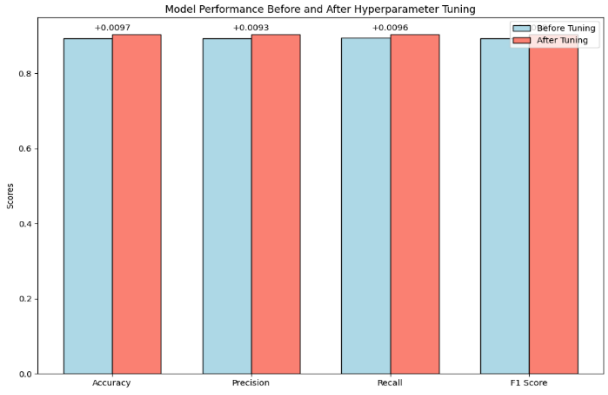

Fig. 3. XGBoost result

	Before Hyperparameter Tuning	After Hyperparameter Tuning
Accuracy	0.8932	0.9029
Precision	0.8934	0.9027
Recall	0.8937	0.9033
F1 Score	0.8930	0.9028

C. Long-Short Term Memory (LSTM)

1) Training Process

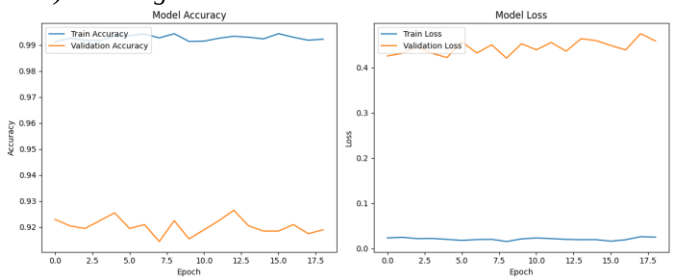

Fig. 4. 3-sec-10-lookback

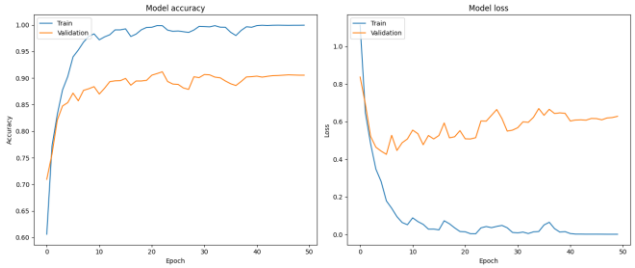

Fig. 5. 10-sec-1-lookback

2) Result

Model	Accuracy	Precision	Recall	F1 Score
3-sec-10-lookback	0.9224	0.9221	0.922	0.9217
10-sec-1-lookback	0.9054	0.9058	0.9053	0.9054

D. Convolution Neural Network (CNN)

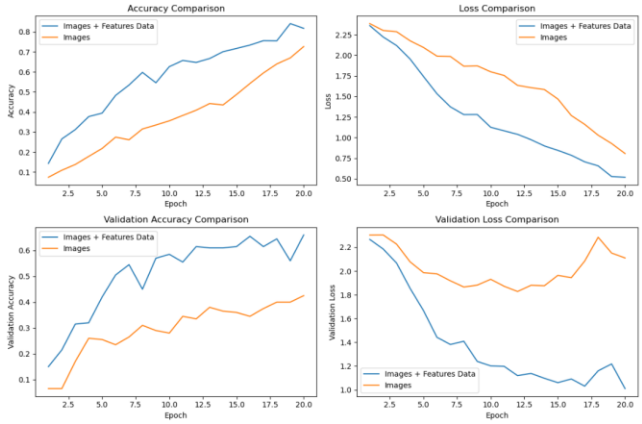

Fig. 6. Comparison between Hybrid and Simple CNN

	Hybrid	Simple
Accuracy	0.6600	0.4200
Precision	0.5614	0.4572
Recall	0.5350	0.4200

F1 Score	0.5407	0.4318

E. Comparison

1) Accuracy is the proportion of correctly classified instances out of the total instances.

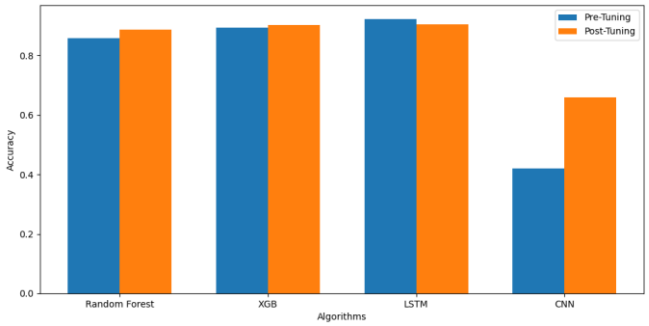

Fig. 7. Accuracy comparison of different models

Random Forest and XGBoost exhibit higher accuracy compared to LSTM and CNN, indicating that they have a higher proportion of correctly classified instances.

2) Precision is the proportion of correctly predicted positive cases out of all instances predicted as positive.

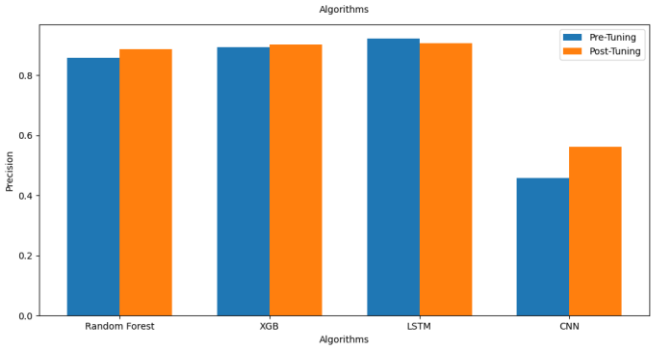

Fig. 8. Precision comparison of different models

LSTM demonstrates the highest precision, followed by XGBoost, Random Forest, and CNN, suggesting that LSTM has the lowest proportion of false positives among the models.

3) Recall is the proportion of correctly predicted positive cases out of all actual positive cases.

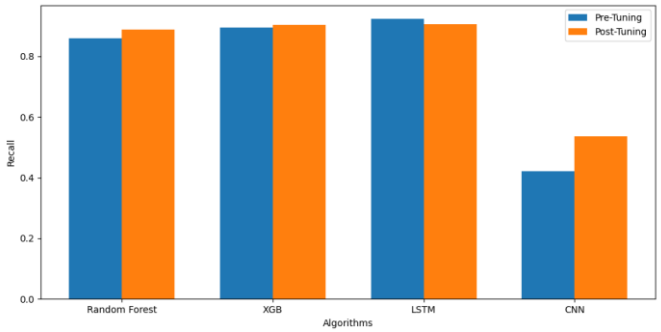

Fig. 9. Recall comparison of different models

LSTM exhibits the highest recall, followed by XGBoost, Random Forest, and CNN, indicating that LSTM has the lowest proportion of false negatives among the models.

4) F1 Score is the harmonic mean of precision and recall, providing a balance between the two metrics.

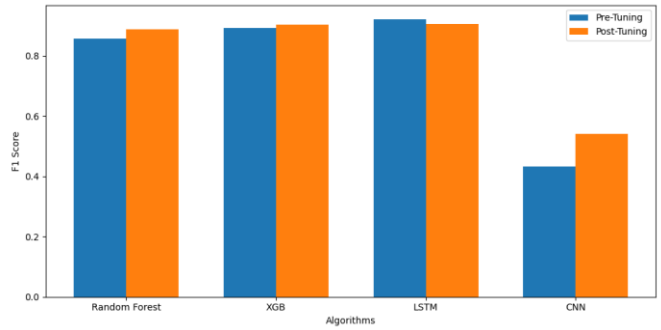

Fig. 10. F1 score comparison of different models

LSTM demonstrates the highest F1 score, followed by XGBoost, Random Forest, and CNN, indicating that LSTM achieves the best balance between precision and recall among the models.

CONCLUSION

Each model showcases different strengths and weaknesses based on the evaluation metrics. Random Forest and XGBoost demonstrate strong accuracy, while LSTM excels in precision, recall, and F1 score. However, CNN, despite its lower accuracy, shows competitive precision and recall, potentially indicating its effectiveness in specific tasks despite the limitations of image data. Therefore, the choice of model depends on the specific requirements and priorities of the music genre classification task.

RESPONSIBILITIES

Siddhartha Kaushik – Training CNN model, Report preparation, Presentation, video, and GitHub repository.
James Zhong – Training LSTM model, report, and video.
Harshit Kamboj – Training Random Forest model and video.
Syed Mohammad Zeeshan Hyder - Training XGBoost model and video.

GITHUB

https://github.com/skaush9/Music-Genre-Classification.git

REFERENCES

[1] **A. Olteanu, "GTZAN Dataset - Music Genre Classification," Kaggle, [Online]. Available: https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification?resource=download. [Accessed: 08-Apr-2024].**

[2] **G. Tessier, "LSTM Implementation_Grad," Kaggle, [Online]. Available: https://www.kaggle.com/code/gtessier/lstm-implementation-grad. [Accessed: 13-Mar-2023].**

[3] **T. Jat, "Music Genre Classification Using CNN," Kaggle, [Online]. Available: https://www.kaggle.com/code/tarushijat/music-genre-classification-using-cnn. [Accessed: 10-Dec-2021].**

[4] **B. Hansenerr, "Music Recommendation, XGBoost over 90% Accuracy," Kaggle, [Online]. Available: https://www.kaggle.com/code/batuhansenerr/music-recommendation-xgboost-over-90-accuracy. [Accessed: 11-Nov-2023]**