Music Genre Classification Using Ensemble Learning

Evan Q. Bonso
BS Computer Science
University of Mindanao
Matina, Davao City, 8000
e.bonso.519162@umindanao.edu.ph

Charleslexcel B. Mendoza
BS Computer Science
University of Mindanao
Matina, Davao City, 8000
c.mendoza.516734@umindanao.edu.ph

George Vincent B. Peña
BS Computer Science
University of Mindanao
Matina, Davao City, 8000
g.pena.519531@umindanao.edu.ph

Fe B. Yara, MSIS
University of Mindanao
Matina, Davao City, 8000
fe_yara@umindanao.edu.ph

ABSTRACT

With the exponential growth of digital music libraries, effective genre classification has become a critical task. This thesis presents an improved method for music genre recognition, leveraging the strengths of Random Forests (RFs), Support Vector Machines (SVMs), and Convolutional Neural Networks (CNNs). A key contribution of this work is the design of a more efficient CNN, which enhances classification performance when combined with the outputs of SVM and RF models. The effectiveness of the proposed strategy is assessed using a selected dataset encompassing ten diverse genres: Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, and Rock. Remarkably, the ensemble model outperforms the baseline models, achieving an accuracy of 84% as opposed to the baseline's 74%. This represents a significant gain in accuracy. Furthermore, the ensemble model demonstrates a substantial reduction in prediction time compared to the baseline models, enhancing its efficiency for real-time applications. The model exhibits robust generalization capabilities, which are crucial for practical deployment. This thesis underscores the potential of the proposed method in advancing music genre classification and opens up opportunities for further refinement and investigation on larger datasets.

Categories and Subject Descriptors

CSS → Information and Retrieval → Artificial Intelligence → Pattern Recognition → Information Interfaces and Presentation

General Terms

Algorithm, Performance, Experimentation, Documentation

Keywords

Convolutional Neural Network; Support Vector Machine; Random Forest; Music Genre Classification; Ensemble Learning

1. INTRODUCTION

1.1 Background of the Study

In recent years, there has been a notable and substantial increase in the widespread adoption and popularity of streaming music services. This has drawn out a range of reactions, including both optimism and apprehension, regarding their effects on revenue generated from recorded music [1]. As a result, there is an increasing need for accurate methods to classify music into genres. The rapid evolution of advanced multimedia technologies has made abundant musical resources accessible online, generating a continuous interest in categorizing diverse music genres [2]. The music industry encompasses various genres and subgenres, making manual classification daunting. The demand for a reliable automated method to classify music is growing increasingly due to the rapid and continuous increase in available music recordings [3]. Thus, automating this process with machine learning techniques can produce more consistent results and save time in classifying.

Recent research on music genre classification has explored various methods to tackle this challenging task. One study employs a Convolutional Neural Network (CNN) algorithm based on spectrogram data but notes the manual selection of feature detectors and a limited number of CNN layers, achieving an accuracy of 72.4% [4]. Although they show promise, CNNs are mostly made to process grid-like data, so they might not be able to accurately represent higher-level musical concepts and for a long-time temporal connection in music [5].

Another study employed musical genre classification using Support Vector Machines (SVMs). They pointed out that they must improve computational efficiency, especially with larger training datasets. While this represents a commendable performance, it also reflects the ongoing challenge of achieving even higher accuracy rates. Moreover, the study emphasizes the necessity to explore additional musical characteristics applicable to the composition of music. Additionally, it underscores the importance of selecting an appropriate kernel function to further enhance the model's effectiveness in music classification.

An examination of a hybrid model that integrates Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) for music categorization suggests that combining these two architectures is beneficial for classifying music based on extractive features. The study achieved an optimal accuracy of 76.40% by combining MFCC features with a blend of Long Short-Term Memory (LSTM) and CNN. Alternatively, when utilizing Mel-spectrogram data, the most effective combination was found to be CNN and Bidirectional Gated Recurrent Unit (Bi-GRU), resulting in an impressive accuracy of 89.30%. This performance surpassed all other hybrid combinations examined in the study [6]. However, they didn't mention how to address the overfitting. Models for deep learning are prone to overfitting, especially when working with limited data [7].

The article presents a fresh method for categorizing musical genres by utilizing the CNN, SVM, and RF. A particular multifaceted strategy is designed to address the challenges highlighted in the studies above to achieve superior accuracy and efficiency in music genre classification. This approach is a promising development in the field. The CNN representation

is commonly used in image retrieval tasks. CNN's are particularly fitted for processing image and audio data due to their aptitude for automatically learning hierarchical characteristics from unprocessed data. Having more than one layer classification system utilizing Support Vector Machines (SVMs) is employed for categorizing musical genres. SVMs are utilized to derive the most effective class boundaries among distinct musical genres by learning from the supplied training facts [8]; SVMs are renowned for their effectiveness in handling high-dimensional data, a crucial aspect when working with spectrogram-based music features. Meanwhile, The Random Forests ensemble predictor has demonstrated adaptability and effectiveness in addressing various prediction tasks.[9]. Random Forests present distinct advantages in music genre classification tasks, offering faster training times compared to Support Vector Machines (SVMs). Their versatility in handling both numerical and categorical data further enhances their suitability for effectively representing features in such classification endeavors. Music genre classification can be enhanced by considering various modalities, including audio features, lyrics, and album art. [10] Addressing these challenges can provide valuable insights into existing methods' limitations and potential improvements. By implementing and comparing the performance of these three prominent machine-learning techniques, researchers can better understand which algorithms are best suited for the music genre classification. This analysis can offer insights into each advantage and disadvantage approach.

Music genre classification poses a fundamental challenge within music information retrieval. At the forefront of standard datasets for this task stands the Kaggle GTZAN dataset, renowned for its significance. One thousand audio tracks in ten different genres are included in this dataset. Every audio track lasts exactly 30 seconds, is sampled at 22,050 Hz, and has been carefully saved in WAV format. Music experts have meticulously curated this dataset, providing ground truth labels as a foundation for evaluation. The primary purpose of the Kaggle GTZAN dataset is to facilitate the development of robust and highly accurate machine-learning models capable of precisely classifying the genre of any given music track.

Over time, it has evolved into a standard benchmark, playing an indispensable role in advancing the field of music information retrieval through the rigorous assessment of diverse music genre classification algorithms.

Researchers are driven to explore CNN optimization for genre classification due to the industry's constant emergence of new music genres. This influx of new genres threatens to overshadow and potentially render older genres obsolete, underscoring the importance of maintaining their relevance through effective classification methods. Our study entitled "Music Genre Classification Using Ensemble Learning" enhances the precision and effectiveness of classification techniques. By employing the GTZAN dataset, researchers can pursue further investigation into innovative methods and strategies for classifying music genres, aiming to enhance the overall quality of music classification.

Table 1. Literature Review of CNN

Reference	Method	Results	Strengths	Weaknesses
Music Genre Classification Using Convolutional Neural Network. [4]	Convolutional Neural Network (CNN)	The accuracy using CNN is 72.4%	Uses a spectrogram that performs better than the MFCC	feature detector is manual-selected
A Hybrid CNN and RNN Variant Model for Music Classification [6]	This method leverages the strengths of both CNNs and LSTMs to capture both local and temporal features in the audio data.	The hybridization of CNN and LSTM using MFCC achieved the greatest accuracy at 76.40%	The achievement of accuracy, specifically 76.40% with CNN and LSTM using MFCC, underscores the effectiveness of the proposed model in the task of music genre classification.	Prone to overfitting.

1.2 Purpose and Description

This research is dedicated to Music Genre Classification Using Ensemble Learning, aiming to substantially enhance the accuracy of music genre identification using machine-learning methods. With the continuous growth of digital music libraries, accurate genre labeling is in pressing demand. This study holds global significance as it contributes to advancing the state-of-the-art in automatic music genre classification within the field of music information retrieval and audio signal processing. Furthermore, it aligns with the United Nations Sustainable Development Goal (SDG) of promoting inclusive and equitable quality education (SDG 4) by facilitating access to organized and accessible digital music libraries, fostering cultural understanding and appreciation through music.

In this research, an innovative approach is suggested for categorizing advanced genres of music, employing Convolutional Neural Networks (CNN) and augmenting their architecture with sophisticated machine-learning techniques like Support Vector Machines (SVM) and Random Forests (RF). The research involves testing various classification approaches on an extended version of the GTZAN dataset, where the original dataset containing 100 audio samples per genre has been augmented with an additional 900 audio samples for each genre like Classical, Hip-hop, Metal, Country, Reggae, Blues, Disco, Jazz, Rock, and Pop. Furthermore, a new label of "Noise" has been introduced with 1000 audio samples. The goal of the researchers is to enhance the accuracy of music genre labeling by optimizing the structure of Convolutional Neural Networks (CNN). We aim to develop a superior system for accurately organizing music by genre through a comparative analysis between the

improved CNN and a conventional one, using this extended dataset. This innovative approach recognizes the potential of CNNs, well-known for their accuracy in image-related tasks, in accurately extracting features from diverse musical patterns. Integrating SVMs in a multi-layer classification system underscores their efficacy in accurately handling high-dimensional data, particularly relevant for spectrogram-based music features. Additionally, we leverage the accuracy and adaptability of Random Forests. Acknowledging challenges such as genre ambiguity and cross-genre influences, the study considers diverse modalities, including audio features, lyrics, and album art, to improve accuracy.

1.3.1 General Objective

This study aims to develop an optimal Convolutional Neural Network (CNN) specifically designed for music genre classification, with the overarching objective of improving accuracy and efficiency in the identification of music genres.

1.3.2 Specific Objective

1.3.2.1 Improve the Convolutional Neural Network (CNN) architecture by integrating the outcomes of Random Forest and Support Vector Machine (SVM) models for the classification of musical genres.

1.3.2.2 To improve the accuracy of music genre classification, train and fine-tune machine learning models such as SVM, Random Forest, and CNN using a well-curated dataset.

1.3.2.3 Analyze the ensemble model's performance in comparison to the baseline CNN model, taking into account information from the RF and SVM models.

1.3.2.4 With an emphasis on the ensemble method with XGBoost, evaluate the outcomes based on accuracy, efficiency, and other pertinent performance metrics to ascertain the efficacy of the improved CNN for music genre classification.

1.4 Scope and Limitation

This study aims to optimize music genre classification by leveraging an ensemble approach that combines Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Random Forests. The research addresses specific challenges in music genre classification, such as genre ambiguity, cross-genre influences, and high-dimensional feature spaces. By integrating these advanced machine learning techniques, the study contributes to the advancement of the field of music information retrieval and audio signal processing, particularly in the domain of automatic music genre classification.

The scope of this study is limited to the classification of ten specific music genres: Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, and Rock. Any other genres beyond these ten will be excluded to minimize potential issues in interpreting the results and ensure a focused analysis. Furthermore, the study will utilize the GTZAN dataset, a widely recognized benchmark in music genre classification research. The GTZAN dataset includes 100 audio files for each of the ten genres, as well as an additional 900 audio files from different songs, not related to the GTZAN dataset. All audio files are 30 seconds in length and in the WAV file format. The study exclusively uses WAV files due to their compatibility with the selected machine learning algorithms and the fact that the GTZAN dataset consists of WAV files. Other audio formats, such as MP3, are not currently supported in this research.

2. Methodology

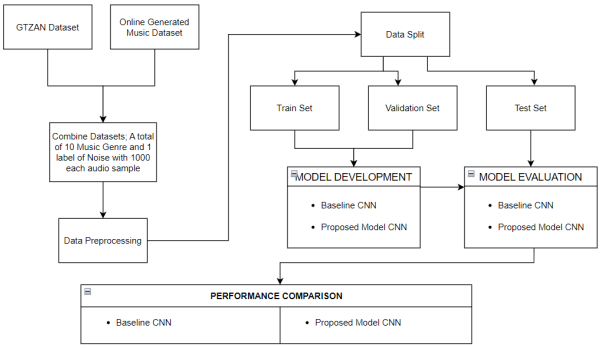

Figure 1. Methodology Flow

Figure 1 illustrates the comprehensive workflow of the project methodology. It begins with the integration of two primary datasets, GTZan and an Online Generated Music dataset, resulting in a diverse dataset encompassing multiple music genres and noise samples. This combined dataset undergoes preprocessing before being divided into Train, Validation, and Test Sets.

The Train and Validation Sets are utilized for model development, where both Baseline and Proposed CNN models are constructed and evaluated. Simultaneously, the Test Set is used for independent evaluation of the models. The results are then compared to determine the performance of each model variant. This structured approach ensures a systematic evaluation of the models' effectiveness in classifying music genres and noise, thereby providing valuable insights for the project's objectives.

2.1 Dataset Collection

This stage involves acquiring or curating a diverse music genre dataset. To ensure thorough model evaluation, the dataset is meticulously partitioned into training and testing sets. This partitioning is crucial to accurately evaluate the performance of machine learning models. [11]

One of the widely adopted datasets for evaluating machine hearing studies, particularly in the classification of music genres by the public, is the GTZAN dataset. Assembled by Tzanetakis et al. in 2001, the GTZAN dataset has since become one of the most-used public datasets, facilitating the evaluation of numerous machine listening research endeavors. Each genre within the GTZAN dataset originally comprised 100 audio samples, representative of songs fitting the genre. To enhance model performance and robustness in real-world scenarios, researchers have expanded each genre by adding 900 additional audio samples from different songs, resulting in a total of 1000 audio samples per genre and additional 1000 audio for noise.

By incorporating this expanded dataset with diverse samples and including noise samples, the model is expected to exhibit increased robustness, accuracy, and efficacy in handling real-world scenarios, ultimately improving the effectiveness and reliability of music genre classification models in music genre classification research.

Table 2. Music Genre Dataset

Genre	GTZAN	Generated	Total
Blues	100	900	1000
Classic	100	900	1000

Country	100	900	1000
Disco	100	900	1000
Hip-hop	100	900	1000
Jazz	100	900	1000
Metal	100	900	1000
Pop	100	900	1000
Reggae	100	900	1000
Rock	100	900	1000

Researchers ensure to obtain 9 basic types of noise datasets that will be used to address potential future scenarios in our study, allowing us to determine if a given sample is noise in case noise is used.

Table 3. Noise Dataset

Type of Noise	Audio Count
Cats	120
Crying	120
Dogs	121
Electricity	120
Fire and Fireplace	120
Ocean Waves	121
Rain and Thunder	119
Stadium Crowd	120
Street Crowd	39
TOTAL:	**1000**

The inclusion of a noise category is essential for our study for two main reasons. Firstly, it enables our model to distinguish between music genres and background noise in audio recordings, improving its reliability in real-world scenarios. Secondly, training our model with noise samples helps it generalize better to diverse audio environments by effectively handling various levels and types of background noise. Overall, incorporating noise samples enhances the practical relevance of our dataset and ensures more accurate music genre classification results in real-world applications.

2.2 Data Preprocessing

The Python library named librosa.load function was used to load the audio file. This function automatically resamples the audio to the given sampling rate (sr=22050 Hz), converts the audio to mono, and loads the audio from the WAV file format.

2.2.1 Loading Audio Files: Raw audio files from the dataset are stored in WAV formats. The ***librosa.load()*** function was used to load audio files into memory. This function returns the audio waveform as a one-dimensional NumPy array and the sampling rate of the audio.

```
def extract_features(file_path):
    audio, _ = librosa.load(file_path)
    mfccs = librosa.feature.mfcc(y=audio, sr=22050, n_mfcc=13)
    chroma = librosa.feature.chroma_stft(y=audio, sr=22050)
    spectral_contrast = librosa.feature.spectral_contrast(y=audio, sr=22050)
    tonnetz = librosa.feature.tonnetz(y=audio, sr=22050)
    features = np.vstack([mfccs, chroma, spectral_contrast, tonnetz])
    return np.mean(features.T, axis=0)
```

Figure 2. Feature Extraction Code

2.3 Feature Extraction

After the preprocessing stage, the function extracts a variety of features from the audio. These features include:

2.3.1 Mel-frequency cepstral coefficients (MFCCs)

In feature extraction their is mfccs function which is ***librosa.feature.mfcc*** ,where it computes the MFCCs of the input audio signal audio and the parameter specified to 13 sets of mfcc ***(n_mfcc=13)*** to extract 13 coefficients. These coefficients capture aspects of the audio signal related to the perceived timbral texture, and they are commonly used as features in music genre classification and also, keep its sampling rate in 22050 ***(sr=22050 Hz)*** to have balance between capturing high-frequency information and computational efficiency.

The 13 MFCCs represent the short-term power spectrum of a sound. The values represent the extracted MFCC features for different audio samples. MFCCs capture the timbral texture of the sound. In music genre classification, they can help distinguish between different instruments and vocal characteristics.

Starting with Figure 3, this is a visual representation of the extracted feature of Mel-frequency cepstral coefficients, it is mfcc1 up until mfcc7. Each row corresponds to one audio sample, and each column (mfcc1, mfcc2, ..., mfcc7) represents a MFCC coefficient.

mfcc1	mfcc2	mfcc3	mfcc4	mfcc5	mfcc6	mfcc7
-113.599	121.571	-19.1623	42.3639	-6.36227	18.6219	-13.6997
-207.524	123.985	8.94702	35.8671	2.90959	21.5195	-8.55651
-90.7572	140.441	-29.0845	31.6867	-13.9765	25.7538	-13.665
-199.575	150.086	5.6634	26.8553	1.77007	14.2326	-4.82785
-160.354	126.209	-35.5814	22.1393	-32.4736	10.8507	-23.3501

Figure 3. Extracted Features

Next is Figure 4, a continuation of Figure 3. The extracted feature of mfcc8 continues to mfcc13.

mfcc8	mfcc9	mfcc10	mfcc11	mfcc12	mfcc13
15.3398	-12.2743	10.9709	-8.32606	8.80209	-3.66994
23.3707	-10.1036	11.8992	-5.55882	5.37788	-2.23449
11.6344	-11.7783	9.71476	-13.1253	5.79125	-8.90197
9.28685	-0.75612	8.13443	-3.20003	6.07808	-2.47845
0.49325	-11.7965	1.20352	-13.085	-2.8105	-6.93447

Figure 4. Extracted Features

The spectral view of MFCC feature vectors for each music genre is as shown below (Figure 5). These visualizations demonstrate the transformation of numerical audio features into spectrograms, allowing us to observe genre-specific patterns and variations.

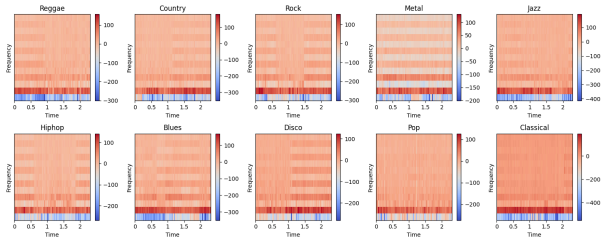

Figure 5. Spectral visualization of MFCC feature vector of each genre

2.3.2 Chroma Features

Chroma features represent the energy distribution of pitch classes in the audio signal, summarizing the presence of different musical notes (pitch classes) in the audio. They are useful for distinguishing between different music genres. Chroma features were also included in feature extraction. The ***librosa.feature.chroma_stft*** function was used to compute these features also the sample rate was set to 22050 ***(sr=22050 Hz),*** the sampling rate is necessary for accurate computation of features like chroma.

The properties of chroma that have been extracted are shown in Figure 6, which includes chromas 1 through 6. In the table, each row is a unique observation or sample. The values represent the energy or intensity of each chroma bin for different audio frames or segments. Higher values indicate a stronger presence of that particular pitch class or semitone in the corresponding frame.

chroma1	chroma2	chroma3	chroma4	chroma5	chroma6
0.36224	0.36782	0.43983	0.24469	0.24786	0.33562
0.46004	0.35018	0.32081	0.21147	0.20079	0.33079
0.2729	0.19595	0.31517	0.40799	0.54682	0.3217
0.34137	0.40862	0.52432	0.54701	0.65708	0.51152
0.1926	0.31727	0.39202	0.25583	0.19266	0.53022

Figure 6. Extracted Features

On the other hand, Figure 7 shows the continuation of Figure 7 which includes chromas 8 through 12.

chroma6	chroma7	chroma8	chroma9	chroma10	chroma11	chroma12
0.33562	0.36462	0.43568	0.29599	0.31508	0.40701	0.3851
0.33079	0.39738	0.56036	0.38418	0.25533	0.28478	0.33409
0.3217	0.30466	0.2884	0.33415	0.40182	0.38439	0.58851
0.51152	0.35555	0.25161	0.24627	0.31511	0.31658	0.3832
0.53022	0.25378	0.13912	0.27642	0.32433	0.55512	0.27293

Figure 7. Extracted Features

The spectral view of chroma feature vectors for each music genre is as depicted below (Figure 8). These visualizations illustrate the conversion of numerical audio features into chroma spectrograms, enabling the observation of genre-specific patterns and variations.

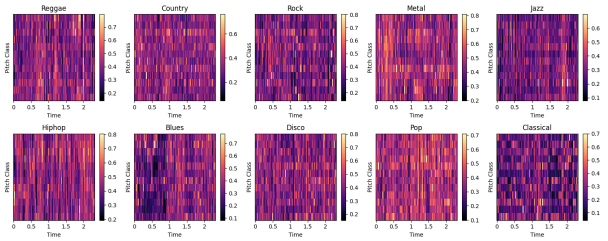

Figure 8. Spectral visualization of Chroma feature vector of each genre

2.3.3 Spectral Contrast

Different music genres often exhibit distinct spectral characteristics, such as differences in tonal color, brightness, and overall tonal complexity. The function from the Librosa library ***librosa.feature.spectral_contrast*** was included during feature extraction to compute the spectral contrast of an audio signal in a quantitative manner while the sample rate of it was set at 22050 ***(sr=22050 Hz)*** to ensures consistency, accuracy, and compatibility of the computed features.

The spectral contrast features that were retrieved are shown in Figure 9, ranging from spectral_contrast1 to spectral_contrast7. The values in the table represent the spectral contrast feature values extracted from audios. Higher values indicate greater spectral contrast, while lower values suggest more spectral flatness.

spectral_contrast1	spectral_contrast2	spectral_contrast3	spectral_contrast4	spectral_contrast5	spectral_contrast6	spectral_contrast7
15.98611558	15.09820328	18.45414937	18.36224256	18.91743082	17.18981618	39.65506589
15.98493822	17.05962995	19.13378648	19.18801307	18.32691537	17.50523891	37.50887831
26.34550276	14.19188167	17.70192409	19.41960509	20.34266469	18.31105909	39.02640652
29.78403828	14.76604254	17.23854978	18.63406551	18.490613	16.78712292	34.26491572
17.05524929	15.69360078	19.13249646	20.5222761	20.78478754	19.71064006	37.31827597

Figure 9. Extracted Features

The spectral view of spectral contrast for each music genre is shown below (Figure 10). These visualizations exemplify the transformation of numerical audio features into spectrograms, facilitating the observation of genre-specific patterns and variations

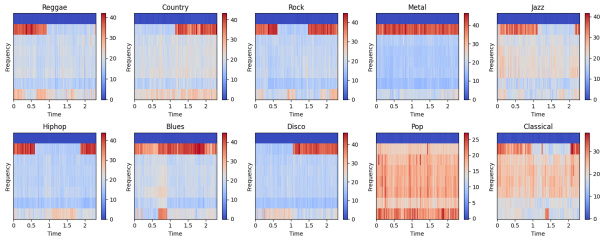

Figure 10. Spectral visualization of Spectral Contrast of each genre

2.3.4 Tonnetz

The tonnetz feature can be used alongside other features such as MFCCs, chroma features, and spectral contrast to provide a comprehensive representation of the audio signal. The librosa.feature.tonnetz function computes the tonnetz feature of an audio signal, and the sampling rate remains consistent with other features, set at 22050 Hz, equivalent to 22,050 samples per second. By incorporating tonnetz, the classification can better capture the tonal and harmonic aspects that are often distinctive of different music genres.

The features of Tonnetz that have been extracted are shown in Figure 11 and range from Tonnetz 1 to Tonnetz 6. Tonnetz 1

to Tonnetz 6 represent different components of the Tonnetz feature vector. Each value quantifies the relative energy or importance of a particular tonal or harmonic relationship within the audio signal. Positive values indicate the presence or dominance of certain tonal relationships, while negative values may suggest their absence or suppression. The magnitudes of the values can indicate the strength or prevalence of those tonal characteristics.

tonnetz1	tonnetz2	tonnetz3	tonnetz4	tonnetz5	tonnetz6
0.02655	0.01822	0.00486	-0.01663	0.00978	-0.00421
0.0356	0.08445	0.04146	0.00797	0.02063	-0.01383
0.1201	-0.06507	0.06609	0.04373	0.00017	-0.01968
0.10203	-0.06181	0.08682	0.07474	-0.005	-0.0368
-0.00525	0.06067	-0.15858	-0.01076	0.01717	-0.02894

Figure 11. Extracted Features

The spectral view of tonnetz feature vectors for each music genre is shown below (Figure 12). These visualizations exemplify the transformation of numerical audio features into tonnetz representations, enabling the observation of genre-specific patterns and variations.

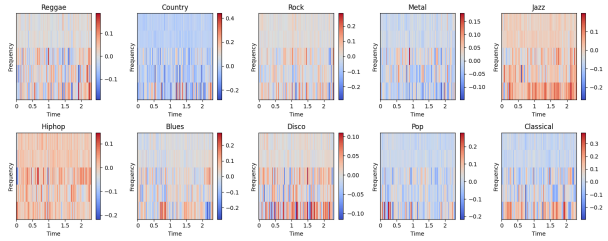

Figure 12. Tonnetz Spectrogram

2.3.5 Feature Representation: Extracted features are typically represented as numerical vectors. These vectors capture various aspects of the audio signal, such as its frequency content, timbral characteristics, and tonal properties.

Table 4 lists the different features together with the relevant data types that are used. Mel-frequency cepstral coefficients (MFCCs) ranging from mfcc1 to mfcc13 are among the features. The distribution of pitch classes is represented by the chroma characteristics, which are designated as chroma1 through chroma12. Furthermore, the properties of spectral contrast (spectral_contrast1 through spectral_contrast7). Lastly, the features of Tonnetz (tonnetz1 through Tonnetz6). The data type that was utilized for each feature was float64.

Table 4. Features and Data Type

Features Name	Data Type
mfcc1	float64
mfcc2	float64
mfcc3	float64
mfcc4	float64
mfcc5	float64
mfcc6	float64
mfcc7	float64
mfcc8	float64
mfcc9	float64
mfcc10	float64
mfcc11	float64
mfcc12	float64
mfcc13	float64
chroma1	float64
chroma2	float64
chroma3	float64
chroma4	float64
chroma5	float64
chroma6	float64
chroma7	float64
chroma8	float64
chroma9	float64
chroma10	float64
chroma11	float64
chroma12	float64
spectral_contrast1	float64
spectral_contrast2	float64
spectral_contrast3	float64
spectral_contrast4	float64
spectral_contrast5	float64
spectral_contrast6	float64
spectral_contrast7	float64
tonnetz1	float64
tonnetz2	float64
tonnetz3	float64
tonnetz4	float64
tonnetz5	float64
tonnetz6	float64

2.4 Data Split (train, validation, test)

It's worth noting that dataset consistency was maintained for both the baseline Convolutional Neural Network (CNN) and the study of Music Genre Classification Using Ensemble Learning across all split ratios. This consistency facilitates the assessment of any disparities in training performance between the baseline and the model. By ensuring consistency in dataset application, we can effectively determine whether notable

differences exist in the training performance of the baseline and models [12].

Table 5 provides a detailed breakdown of music genre sample distribution across the training, testing, and validation sets. The training set comprises 70% of the data, while the testing set includes 20%, and the validation set contains 10%. This distribution allows for a thorough examination of dataset balance across various genres, which is crucial for mitigating biases during model training. Maintaining a balanced dataset, where genres are evenly represented, is critical for achieving unbiased model performance.

Table 5. Dataset Split result

Genre	Training	Testing	Validation
Blues	725	206	69
Classical	720	202	78
Country	734	184	82
Disco	696	214	90
Hiphop	729	197	74
Jazz	701	204	95
Metal	712	210	78
Pop	718	198	84
Reggae	732	194	74
Rock	705	213	82
Noise	748	178	74

2.5 Training Environment

The researcher's training environment has 12 physical cores (and 16 total cores) and approximately 8.28 GB of RAM, which is capable of handling a variety of data processing and model training tasks.

2.5.1 Hardware Specifications

Table 6. Hardware Specification

Physical Cores	**12 cores**
Total Cores	**16 cores**
Total Memory	**8280072192 bytes**

The hardware specifications of the researcher training environment are presented in Table 6, which includes important information necessary to comprehend its capabilities. There are 16 cores in total in the environment, 12 of which are physical cores. Effective multitasking and parallel processing are made possible by this architecture, which is essential for managing a variety of data processing and model training tasks. Together with its strong number of cores, the system has about 8.28 gigabytes of RAM, which is more than enough memory to support these jobs efficiently. These specs highlight the environment's overall aptitude for demanding computational workloads, which makes it a good fit for a range of data science and machine learning research and development projects.

2.5.2 Software Specifications

Table 7. Software Specifications

Operating System	Programming language
Windows 11	Python version. 3.10.6

The software specifications of the researcher's training environment are presented in Table 7, which includes crucial information about the operating system and programming language. Python is the main programming language used in the system; version 3.10.6 is listed. Python is a popular choice for activities related to data analysis, machine learning, and scientific computing because of its huge libraries and versatility. Furthermore, the environment runs on the Windows 11 operating system, which is renowned for its intuitive interface and broad program compatibility. The combination of Python and Windows 11 highlights how well-suited the environment is for research and development in domains like machine learning and data science, giving researchers the instruments they need to examine and evaluate large, intricate datasets efficiently.

2.5.3 Imports/library

A selection of import lines and library installations necessary for data analysis and machine learning operations are shown in Figure 10. Many tools and frameworks are included in these imports. Some of them are TensorFlow, Keras, scikit-learn, XGBoost, and librosa. Every library has a distinct function, such as evaluation, model construction, and data preprocessing. Specifically, deep learning, ensemble learning, and conventional machine learning algorithms can be used for various tasks thanks to these imports. Creating informative plots and graphs for data exploration and presentation is further made easier by including visualization packages like Matplotlib and Seaborn. All things considered, these imports set the stage for thorough workflows for data analysis and model creation, enabling practitioners and academics to tackle challenging issues in data science and machine learning.

```
import json
import joblib
import matplotlib.pyplot as plt
import numpy as np
import os
import seaborn as sns
import tensorflow as tf
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout, BatchNormalization
from keras.models import Sequential
from keras.utils import to_categorical
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from tabulate import tabulate
from xgboost import XGBClassifier
import librosa
import joblib
import pandas as pd
import platform
import psutil
import platform
import pkg_resources
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import time
```

Figure 10. Imports/Library

2.6 Model Development

Figure 14 depicts the architecture of a Baseline Convolutional Neural Network (CNN) for music genre classification. It starts with input data. Following Data Preprocessing and Feature

Extraction, the processed data is fed into the CNN. The input audio data is used by CNN to learn hierarchical representations of features, and then finally, the result.

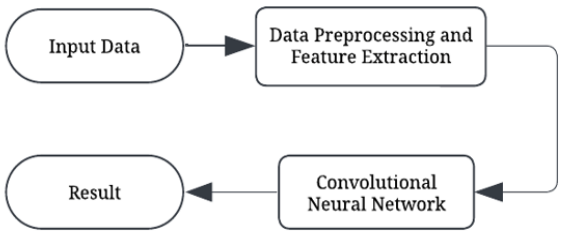

Figure 14. Baseline Convolutional Neural Network

Figure 15 presents a comprehensive workflow for music genre classification employing ensemble learning methodologies. It begins with the input data, comprising audio files from the dataset that the researchers gathered. These data undergo data preprocessing and feature extraction stages. After processing these features, the Convolutional Neural Network (CNN) learns hierarchical representations from the input data. Following this, the processed features are also inputted into Support Vector Machine (SVM) and Random Forest classifiers, each employing distinct strategies for classification. The outputs from these classifiers, alongside the CNN, are combined in an ensemble using XGBoost to improve the model's performance. The ensemble model generates the final result.

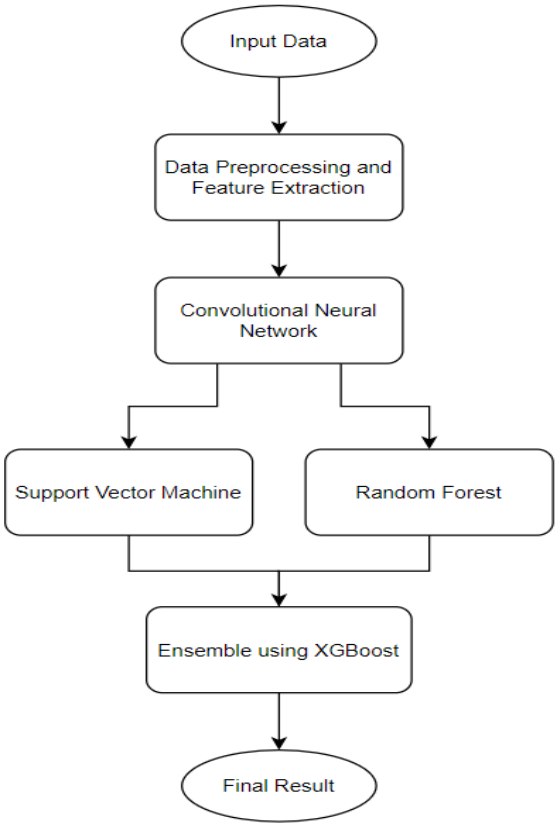

Figure 15. Music Genre Classification Using Ensemble Learning

2.6.1 Build CNN

pre-processed audio files are fed to the CNN model. The pre-processing involves converting the raw audio signals into a suitable format (such as spectrograms or Mel-frequency cepstral coefficients (MFCCs)) that can be input to the CNN. Researchers created a feature_extractor_model for extraction that includes all layers up to the last dense layer, which excludes the classification layer. The training of the model is done using the Adam optimizer, with a learning rate of 0.0001. The loss function is categorical cross-entropy, which is suitable for multi-class classification problems. Researchers also set up early stopping and learning rate reduction callbacks to improve the training process.

The results after CNN-feature extraction would be a set of 256-dimensional feature vectors, one for each audio file. These feature vectors capture the essential characteristics of the audio files, as learned by the CNN, and can be used as input for other models or for further analysis.

2.6.1.1 Model Architecture

The CNN model consists of three convolutional layers, each followed by batch normalization and max pooling. After the convolutional layers, the model has a flattened layer, a dense layer with 256 units, a dropout layer, and a feature_extractor_model for feature extraction to feed into another model (SVM and Random Forest), which it excludes from the classification layer.

Researchers used the EarlyStopping callback from Keras. This callback will monitor the validation loss (val_loss) during the training. If the validation loss does not decrease for a specified number of epochs (patience=10), the training will be stopped.

The restore_best_weights parameter is set to True, which means that the model weights from the epoch with the best value of the monitored quantity (validation loss) will be restored. This means that even if the model performance on the validation set starts to degrade after a certain point due to overfitting, it will still end up with the weights that gave the best validation performance.

This technique allows researchers to automatically find a good compromise between the need to train the model long enough to allow it to converge to a good solution and the need to stop training early enough to prevent overfitting. It's a simple yet effective method to prevent overfitting and save computational resources.

Table 8. Model Architecture

Layer	Type	Output Shape	Param #
1	conv1d_3 (Conv1D)	38, 64	384
2	batch_normalization_3 (BatchNormalization)	38, 64	256
3	max_pooling1d_3 (MaxPooling1D)	19, 64	0
4	conv1d_4 (Conv1D)	19, 128	41088
5	batch_normalization_4 (BatchNormalization)	19, 128	512
6	max_pooling1d_4 (MaxPooling1D)	9, 128	0

7	conv1d_5 (Conv1D)	9, 256	164096
8	batch_normalization_5 (BatchNormalization)	9, 256	1024
Trainable params:		471691	
Non-trainable params:		896	

2.6.1.2 Hyper-parameter Tuning for CNN

Hyper-parameter tuning plays a critical role in optimizing the performance of the CNN model, particularly in the realm of feature extraction from spectrogram data. Various hyper-parameters were systematically tuned to enhance the quality of the extracted features and improve the overall performance of the model.

2.6.1.2. Filters

The number of filters in each convolutional layer determines the model's capacity to extract diverse and discriminative features from the input spectrogram.

2.6.1.1.3 Kernel Size

The size of the convolutional kernel influences the receptive field of the filters and the granularity of feature extraction.

2.6.1.1.4 Dropout Rate

The dropout rate, regulates the extent of regularization applied during training by specifying the fraction of neurons to be dropped at each training iteration.

2.6.1.1.5 L2 Regularization

L2 regularization, incorporated via kernel regularization in convolutional layers, penalizes large weight values, thereby preventing overfitting and promoting smoother decision boundaries.

These hyper-parameters were meticulously tuned through empirical experimentation and cross-validation to strike a balance between model complexity and generalization performance. The resulting CNN model demonstrated superior capability in extracting meaningful representations, thereby facilitating accurate and reliable music genre classification.

2.6.2 Build SVM and RF Classifier

In addition to the CNN model, support vector machine (SVM) and random forest (RF) classifiers were employed to complement feature extraction. Leveraging the extracted features from the CNN model, SVM and RF classifiers were trained to exploit the discriminative information embedded within the learned representations.

SVM classifiers were trained to discern boundaries between different music genres by leveraging the high-dimensional feature space generated by the CNN model. Through grid search and cross-validation, optimal hyper-parameters for the SVM model, including regularization parameter (C), kernel type, and gamma parameter, were meticulously tuned to maximize classification accuracy.

RF classifiers were employed to harness the ensemble decision-making capabilities of multiple decision trees. By leveraging the extracted features as input, RF classifiers were trained to exploit the diversity of decision trees to enhance classification performance. Hyper-parameters such as the number of estimators, maximum depth of trees, and minimum samples per leaf were optimized through grid search and cross-validation.

The SVM and RF classifiers, coupled with the feature extraction capabilities of the CNN model, collectively constitute a robust framework for music genre classification. Through meticulous hyper-parameter tuning and ensemble learning, these classifiers offer complementary approaches to exploit the discriminative information embedded within the spectrogram representations, thereby facilitating accurate and reliable genre classification.

2.6.2.1 SVM

Support Vector Machine was utilized as a classification algorithm. The implementation involved creating an SVM model with default hyperparameter values. Hyperparameter tuning is performed using grid search, a method for exhaustively searching over a predefined parameter grid to identify the best combination of hyperparameters. Subsequently, hyperparameter tuning was performed to optimize the SVM model's performance. This tuning process involved systematically exploring different combinations of hyperparameters to identify the configuration that maximized the model's effectiveness.

To evaluate the computational efficiency of the SVM model, the execution time was measured. This involved recording the time taken for key operations such as hyperparameter tuning and making predictions on the test dataset. Analyzing the execution time provided insights into the model's computational demands and scalability, which are essential considerations for real-world applications and large-scale datasets.

2.6.2.2 RF Classifier

The Random Forest (RF) Classifier was implemented using a widely used ensemble learning technique. Initially, a Random Forest model was created with default settings, and training was conducted on the provided training dataset. After training, the RF Classifier's performance was evaluated on the test dataset to assess its classification effectiveness. Furthermore, to understand the computational efficiency of the RF Classifier, the execution time was measured. This involved recording the time taken for both model training and making predictions on the test dataset. These time measurements provided insights into the RF Classifier's computational demands and scalability, which are essential considerations for real-world applications and large-scale datasets. Overall, the accuracy and execution time measurement of the RF Classifier contributed to a thorough assessment of its suitability and effectiveness for the thesis project's objectives.

2.6.3 Ensemble Method

The ensemble method used is a form of stacking, and research using the XGBoost model serves as the meta-learner. In stacking, multiple base models are trained to make predictions, and then a meta-learner (in this case, XGBoost) is trained to make a final prediction based on the predictions of the base models. The base models are typically different machine learning algorithms, and they can be trained on different subsets of the data. The underlying idea is that the meta-learner can learn how to optimally combine the strengths of the base models to enhance the overall prediction, thereby making the prediction process much faster. This approach often leads to better performance than any single base model could achieve on its own, effectively addressing the objectives of the study.

3. Results and Discussion

3.1 Performance of Baseline and Ensemble Model

In the conducted experiments, several models were evaluated for their accuracy and prediction time, including Support Vector Machines (SVM), Random Forest, an ensemble model using XGBoost, and a traditional baseline approach. The XGBoost ensemble model demonstrated superior performance with the highest accuracy of 0.8368% and the fastest prediction time of 0.0448 seconds, making it the most effective model in terms of both accuracy and efficiency. The SVM model, while achieving a commendable accuracy of 0.8214, exhibited a longer prediction time of 1.4705 seconds. The Random Forest model, on the other hand, had a lower accuracy of 0.7786 but boasted a faster prediction time of 0.2580 seconds compared to the SVM model. The baseline model showed the lowest accuracy of 73.86% among the models tested. However, its prediction time of 0.3672 seconds was competitive, being faster than the SVM model.

Furthermore, the ensemble model using XGBoost emerged as the most effective model in this study, thereby successfully attaining Objective 1.3.2.4 of our study. It outperformed the base models (SVM and Random Forest) and the traditional baseline approach in terms of accuracy, while also demonstrating superior efficiency in prediction time. This suggests that the ensemble model is a promising approach for music genre classification tasks.

Table 9. Accuracy and Speed

	Accuracy	Speed
Ensemble	0.8368%	0.0448 seconds
SVM	0.8014%	1.4705 seconds
RF	0.7886%	0.2580 seconds
Baseline (CNN)	74.1228%	0.3672 seconds

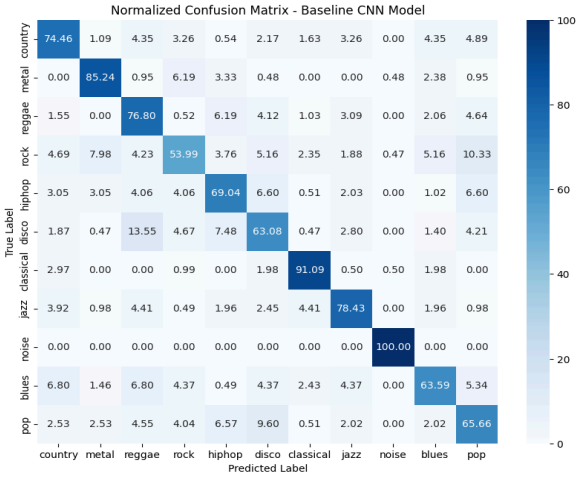

Figure 16. Baseline Model Confusion Matrix

From the matrix of the ensemble model (Figure 17) and the baseline model (Figure 16), we can observe that the ensemble model performs exceptionally well on certain genres like metal (90.5% accuracy), jazz (90.7% accuracy), noise (99.4% accuracy), and classical (95% accuracy). We can see that the baseline model appears to have more difficulty separating "pop" from "rock" compared to the ensemble model. The ensemble model demonstrates a more stable performance across genres, with fewer instances of very low accuracies (below 50%), unlike the baseline model which shows some instances of low accuracies, e.g., "disco" being misclassified as "reggae" (13.55%). Furthermore, the ensemble model's performance is more consistent, stable, and has better accuracy across genres, with fewer instances of significant confusion compared to the baseline model.

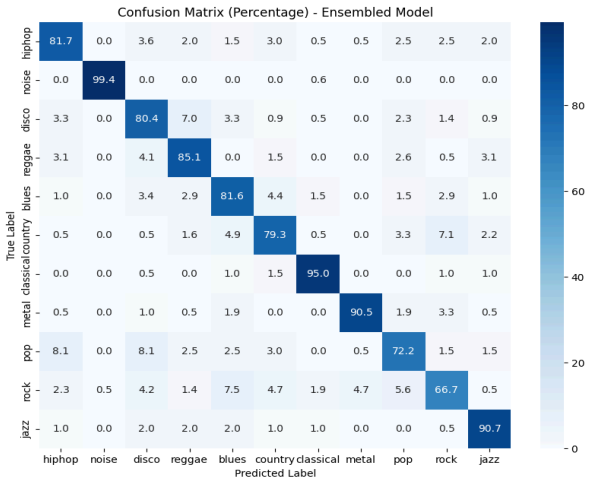

Figure 17. Ensemble model confusion matrix

3.1.1 Evaluation metrics

Researchers used F1 score, recall, precision, and support for the evaluation metrics commonly to assess the performance of the ensemble model and baseline CNN model, since it involves classification tasks. They are useful for understanding how well a model is performing.

3.1.1.1 Accuracy

It is a crucial evaluation statistic that quantifies the overall correctness of the classification model's predictions is a fundamental aspect of assessment. It represents the ratio of correctly predicted cases to the total number of instances. In music genre classification, accuracy provides insights into how effectively the model identifies genre labels.

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ positive + False\ Negative}$$

The table highlights the improved performance of the ensemble model over the baseline model, with the ensemble model achieving an accuracy of 0.84 compared to the baseline of 0.74, demonstrating the potential benefits of combining multiple models for enhanced accuracy.

Table 10. Accuracy Comparison between the Baseline and the Ensemble Model

Accuracy	
Baseline	Ensemble Model
0.74	**0.84**

3.1.1.2 Precision

Precision, also referred to as A good value for prediction, gauges the model's effectiveness in accurately predicting positive outcomes. In the context of this study, precision denotes the ratio of correctly predicted positive instances (i.e., accurately classified genre labels) to the total instances

predicted as positive (i.e., genre labels predicted by the model).

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Precision is an assessment score that evaluates the accurate positive predictions made by a machine learning model. Calculated by taking the true positives and dividing them by the total instances predicted as positive, including both true and false positives [29].

The ensemble model outperforms the baseline model in terms of precision. The genres where the ensemble model shows a significant improvement over the baseline are Classical (0.94 vs. 0.87), Metal (0.94 vs. 0.83), Hip-hop (0.80 vs. 0.69), Jazz (0.88 vs. 0.80), Reggae (0.80 vs. 0.63), and Disco (0.76 vs. 0.65).

This shows that, the results suggest that the ensemble model is generally more precise than the baseline model in classifying music genres,

The results indicate that the ensemble model, which combines multiple models, generally outperforms the baseline model in terms of precision across most genres.

Table 11. Precision Result Comparison between the Baseline and the Ensemble Model

	Precision	
	Baseline	Ensemble Model
Blues	0.74	**0.77**
Classical	0.87	**0.94**
Country	0.71	**0.78**
Disco	0.65	**0.76**
Hip-hop	0.69	**0.80**
Jazz	0.80	**0.88**
Metal	0.83	**0.94**
Pop	0.63	**0.78**
Reggae	0.63	**0.80**
Rock	0.66	**0.78**
Noise	0.98	**0.99**

3.1.1.3 Recall

It is also referred to as sensibility or true positive rate, assesses the model's abilities to correctly detect all relevant events. It gauges the portion of true positive genre labels successfully identified by the model in music genre classification relative to all positive instances.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Recall, as an evaluation metric, quantifies the ability of a machine learning model's capability to accurately identify positive instances, particularly the true positives, within all the genuine positive samples present in the dataset. To compute recall, the true positives (correctly identified cases) are divided by the total number of positive instances, including both the true positives and the instances that were missed, termed false negatives. [29].

The ensemble model demonstrates superior recall performance across most categories, suggesting it is better at identifying true positive instances compared to the baseline model. The only exception is the "Noise" category, where the baseline model has a marginal edge in the recall.

Table 12. Recall Result Comparison between the Baseline and the Ensemble Model

	Recall	
	Baseline	Ensemble Model
Blues	0.64	**0.82**
Classical	0.91	**0.95**
Country	0.74	**0.79**
Disco	0.63	**0.80**
Hip-hop	0.69	**0.82**
Jazz	0.78	**0.91**
Metal	0.85	**0.90**
Pop	0.66	**0.72**
Reggae	0.77	**0.85**
Rock	0.54	**0.67**
Noise	**1.00**	0.99

3.1.1.4 F1 score

It provides an equal evaluation of the model's performance. as it represents the harmonic mean of precision and recall. It is especially useful when the dataset is unbalanced, as in many real-world music genre classification scenarios. The F1 score considers both precision and recall, offering a single metric that encapsulates the trade-off between these two crucial aspects of classification accuracy.

The mathematical formula for the F1 score is as follows:

$$F1 = 2\,x\,\frac{Precision * Recall}{Precision + Recall}$$

The F1 score is calculated as the harmonic mean of precision and recall, providing a fair evaluation that considers both precision and recall together [30].

The ensemble model demonstrates superior F1 scores across most categories compared to the baseline model, indicating its better ability to balance precision and recall. The only exception is the "Noise" category, where both models perform equally well in terms of F1 score.

Table 13. F-1 Score Result Comparison between the Baseline and the Ensemble Model

	F-1 Score	
	Baseline	Ensemble Model
Blues	0.69	**0.79**

Classical	0.89	**0.94**
Country	0.73	**0.79**
Disco	0.64	**0.78**
Hip-hop	0.69	**0.81**
Jazz	0.79	**0.89**
Metal	0.84	**0.92**
Pop	0.64	**0.75**
Reggae	0.69	**0.82**
Rock	0.60	**0.72**
Noise	0.99	0.99

The support is a measure of how many actual positives there are in the dataset. In the context of music genre classification, it would represent the number of songs in each genre in your dataset. The support results for the Baseline and Ensemble models are almost identical across all genres, it suggests that both models have similar performance in terms of recognizing the different genres in the dataset.

Table 14. Support Result Comparison between the Baseline and the Ensemble Model

	Support	
	Baseline	Ensemble Model
Blues	206	206
Classical	202	202
Country	184	184
Disco	214	214
Hip-hop	197	197
Jazz	204	204
Metal	210	210
Pop	198	198
Reggae	194	194
Rock	213	213
Noise	178	178
Total	2200	2200

3.1.2 Training vs Validation

To understand how well the CNN is performing on its own before feeding its results into the SVM and Random Forest models, researchers test its performance on the training and validation sets. This is done to understand the effectiveness of the model and to ensure that it contributes positively to the final ensemble model. It's useful to understand its performance since this can give researchers an insight into the quality of the features it's extracting. If the CNN is performing poorly, the features it extracts might not be very useful for the SVM and RF models.

Moreover, evaluating the model on the training and validation sets also helps in checking if the model is overfitting or underfitting. Overfitting occurs when the model performs well on the training data but poorly on the validation data, which can indicate that the model is memorizing the training data rather than learning to generalize from it. On the other hand, underfitting occurs when the model performs poorly on both the training and validation data, which can suggest that the model is too simple to capture the underlying patterns in the data. By monitoring the training and validation loss and accuracy, researchers can identify if the model is overfitting or underfitting, and adjust the model's complexity or the training process accordingly to improve its performance.

3.1.2.1 Performance of Baseline Model in 30 Epochs

In epoch 30, the models' performance achieved the highest training accuracy of 73.93% at epoch 30, accompanied by a validation accuracy of 70.91%. The minimum training loss occurred at epoch 30, with a value of 0.7725, while the validation loss was 0.8219.

After testing, the restore_best_weights parameter will be executed to restore the epoch with better performance, which is the one that has the closest match in terms of loss and accuracy between training and validation, occurring in epoch 21. At this point, the baseline model demonstrates balanced performance between the training and validation datasets, with a training loss of 0.9102 and a validation loss of 0.9032. Additionally, the validation accuracy slightly surpasses the training accuracy, with values of 69.20% and 69.05%, respectively. The minimal difference in loss values indicates good generalization and suggests that the model is not prone to overfitting.

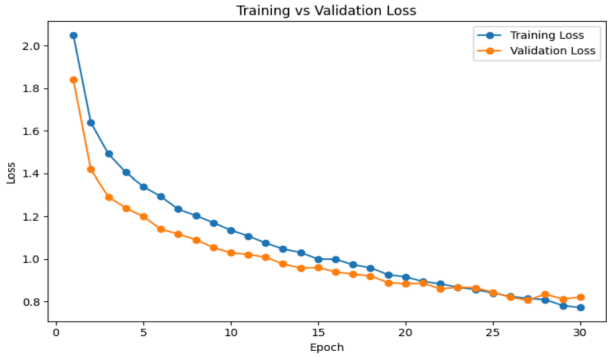

Figure 18. Baseline Training VS Validation Loss in 30 Epochs

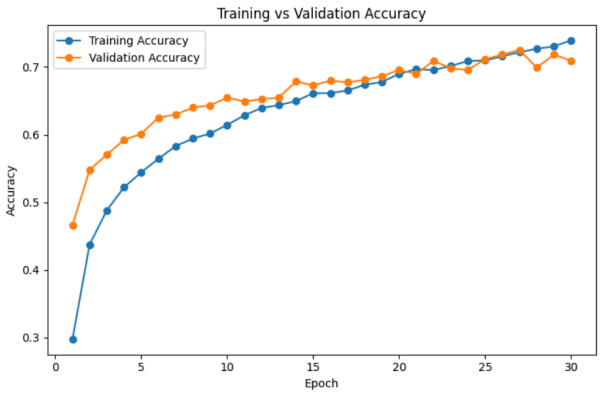

Figure 19. Baseline Training VS Validation Accuracy in 30 Epochs

3.1.2.2 Performances Result of the Model in 30 Epochs

At epoch 30, the model exhibited impressive performance metrics: a training accuracy of 78.83% and a validation accuracy of 77.73%, indicating robust performance on both training and validation datasets. The minimum training loss was 1.1316, with a validation loss of 1.1831, demonstrating effective model training and generalization. Suggests that the model's performance extends well to unseen data.

The restore_best_weights parameter restores the model's weights at epoch 14, as it demonstrates balanced performance between the training and validation datasets. At this epoch, the training loss is 1.4739 and the validation loss is 1.4340, with a minimal difference between them, indicating good generalization. Additionally, the validation accuracy of 69.66% slightly surpasses the training accuracy of 68.51%, suggesting effective learning without overfitting.

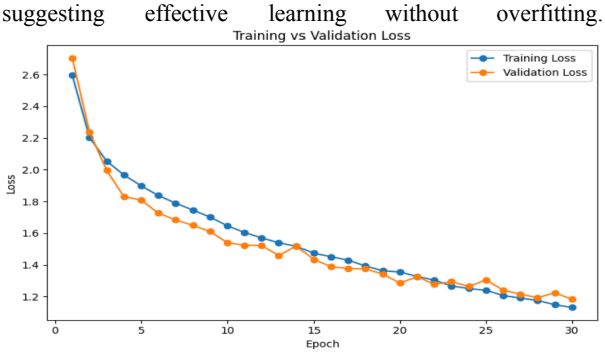

Figure 21. Model Training VS Validation Loss in 30 Epochs

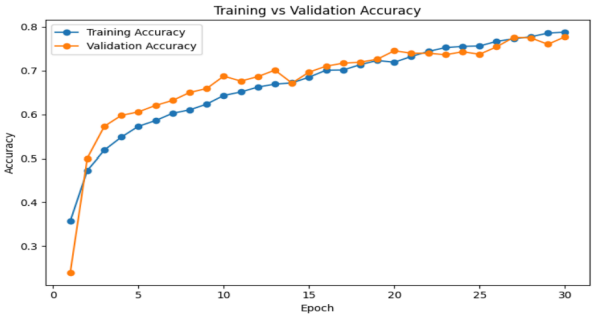

Figure 22. Model Training VS Validation Accuracy in 30 Epochs

3.1.2.3 Performance result of Baseline Model in 40 Epochs

The performance of the baseline model after 40 epochs of training attained its highest training accuracy of 76.44% at epoch 40, accompanied by a validation accuracy of 73.43%. At epoch 40, the minimum training loss was 0.6839, and the validation loss was 0.7276.

After testing the performance result of the baseline model in 40 epochs the restore_best_weights parameter restores the result in 23 epochs. At this epoch, the training loss is 0.9334 and the validation loss is 1.9125, with a minimal difference between them, indicating good generalization. Additionally, the validation accuracy of 69.21% is closer to the training accuracy of 69.87%, suggesting effective learning without overfitting.

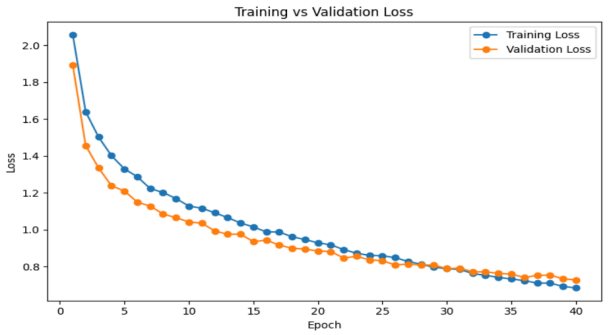

Figure 23. Baseline Model Training VS Validation Loss in 40 Epochs

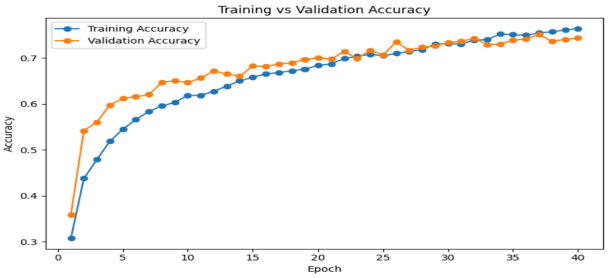

Figure 24. Baseline Model Training VS Validation Accuracy in 40 Epochs

3.1.2.4 Performances Result of the Model in 40 Epochs

The Performance result of the model in 40 epochs, training accuracy achieved was 82.77%, with a validation accuracy of 78.80%. The minimum training loss occurred at Epoch 40, with a value of 0.9733, and the validation loss was 1.1005. The performance comparison between the model and the baseline model after 40 epochs of training reveals significant improvements in accuracy and loss metrics. Specifically, the model achieved a notably higher training accuracy of 82.77% compared to the baseline model which is 76.44%. Additionally, the validation accuracy of the model was substantially higher at 79.80%, outperforming the baseline model's validation accuracy of 73.43%.

The restore_best_weights parameter restores the result of epoch 18 as its result since it represents the best performance of the model in 40 epochs. The training loss was 1.5014, while the validation loss was 1.5112. This is the point where the validation loss becomes slightly lower than the training loss. Additionally, the validation accuracy (71.24%) surpasses the training accuracy (71.09%), suggesting that the model is

generalizing well to the validation dataset, indicating a balanced performance.

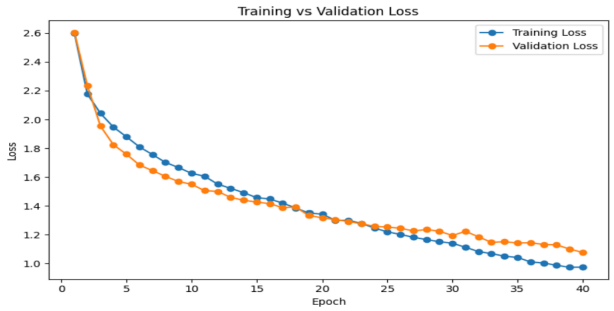

Figure 25. Model Training VS Validation Loss in 40 epochs

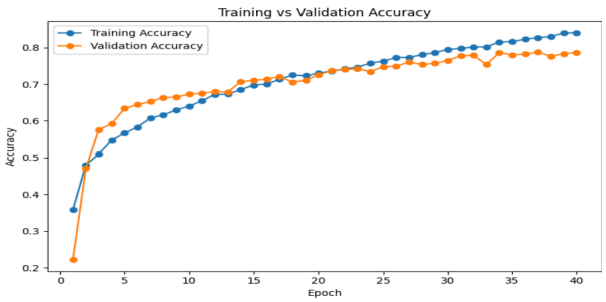

Figure 26. Model Training VS Validation Accuracy in 40 Epochs

3.1.2.5 Performance result of Baseline Model in 50 Epochs

In the 50th epoch, the baseline model achieved the highest training accuracy of 77.78% and a corresponding validation accuracy of 74.27%. The minimum training loss recorded at epoch 50 was 0.6382, with a validation loss of 0.7108. These results indicate moderate performance in classifying music genres, with the potential for further optimization to improve accuracy for practical applications.

At epoch 25, significant observations are made. The training loss was 0.9446, while the validation loss was 0.9465. This is the point where the validation loss becomes slightly lower than the training loss for the first time. Additionally, the validation accuracy (69.73%) surpasses the training accuracy (69.29%) for the first time as well. This epoch marks a critical transition point in the model's training process, as the training and validation loss curves start to intersect. It suggests that the model is generalizing well to the validation dataset, indicating a balanced performance.

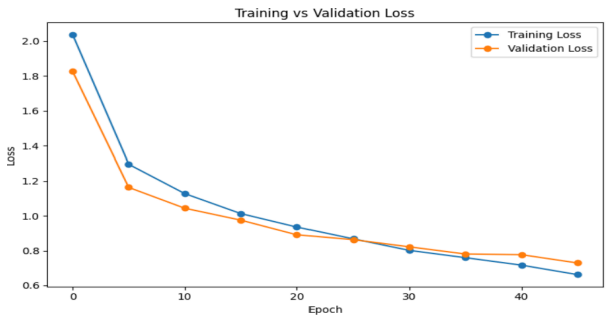

Figure 27 Baseline Training VS Validation Loss in 50 Epochs

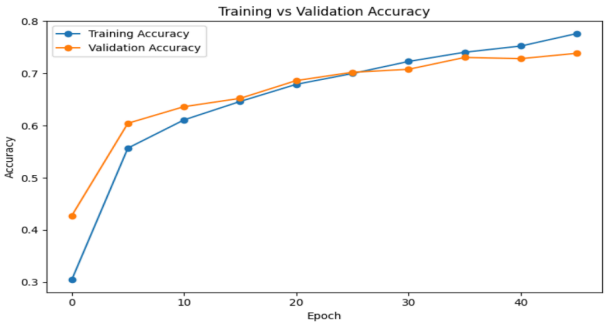

Figure 28. Baseline Training VS Validation Accuracy in 50 Epochs

3.1.2.6 Performances Result of the Model in 50 Epochs

In the 50th epoch, the model achieved its highest training accuracy of 85.22%, accompanied by a validation accuracy of 79.89%. The model also recorded its lowest training loss of 0.8172 and a validation loss of 0.6964, both occurring at Epoch 50.

After the testing of 50 epochs, the restore_best_weights parameter restores epoch 20, since its training loss was 1.3487, while the validation loss was 1.3248. This is the point where the validation loss becomes slightly lower than the training loss for the first time. Additionally, the validation accuracy (73.73%) surpasses the training accuracy (73.29%) for the first time as well. It suggests that the model is generalizing well, indicating a balanced performance and not prone to overfit.

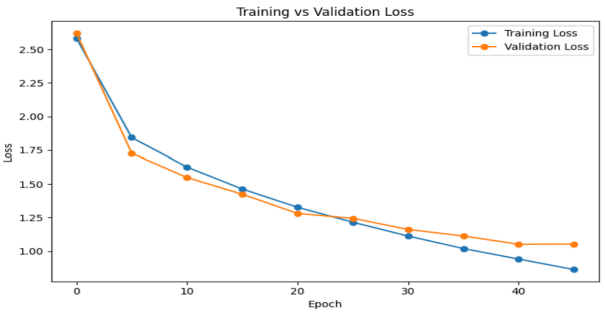

Figure 29. Model Training VS Validation Loss in 50 Epochs

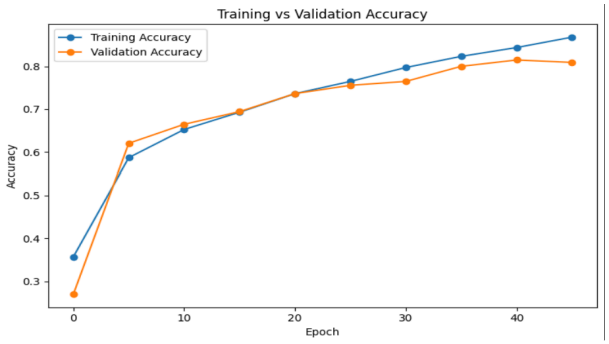

Figure 30. Model Training VS Validation Accuracy in 50 Epochs

In summary, the model achieved its highest training accuracy of 86.22% and a validation accuracy of 79.39%.Despite the impressive performance at epoch 50, it's noteworthy that the restore_best_weights parameter chose to revert to the model's state at epoch 20. This decision was based on specific criteria: at epoch 20, the model exhibited a training loss of 1.3487 and

a validation loss of 1.3248. Importantly, this epoch marked the first instance where the validation loss became slightly lower than the training loss. Furthermore, epoch 20 also saw the validation accuracy (73.73%) surpassing the training accuracy (73.29%) for the first time, indicating that the model was effectively learning from the validation data and not just memorizing the training set, the decision to restore weights from epoch 20 underscores the importance of prioritizing balanced performance and generalization over solely maximizing accuracy. By reverting to epoch 20, the model can maintain its ability to generalize well to unseen data, ensuring robust performance in music genre classification tasks.

The baseline model achieved a training accuracy of 77.78% and a validation accuracy of 74.27%. The restore_best_weights parameter chose to revert to the baseline model state at epoch 25, where significant observations are made. The training loss was 0.9446, while the validation loss was 0.9465. This is the point where the validation loss becomes slightly lower than the training loss for the first time. Additionally, the validation accuracy (69.73%) surpasses the training accuracy (69.29%) for the first time as well. This epoch marks a critical transition point in the model's training process.

These results underscore the robust performance of the model in music genre classification, highlighting its effectiveness in achieving high accuracy rates across training, validation, and test datasets which uses a CNN for feature extraction, consistently outperforming the baseline model across all epochs. This suggests that the features extracted by the CNN were highly effective for music genre classification. The use of a CNN for feature extraction appears to be a significant factor in the superior performance of the model compared to the baseline model. The training and validation accuracies of the model are relatively close, suggesting that the model is neither overfitting nor underfitting, but rather, it is generalizing well to unseen data, which is a good indication of a well-performing model.
The process of feature extraction in an ensemble model can greatly enhance accuracy by converting the input data into a more beneficial format, which is subsequently utilized by the ensemble models for making predictions. This procedure can assist in decreasing the dimensionality of the data, thereby simplifying the learning and processing tasks for the model. Furthermore, ensemble models amalgamate the predictions from multiple models, which can frequently outperform any individual model. This amalgamation is achieved by aggregating the output from each model with dual objectives: minimizing the model error and preserving its generalization.

Hyperparameters play a crucial role in influencing a model's performance as they govern the model's complexity and the optimization process during model training.

CNN Hyperparameters: The model, researchers used a specific number of filters, kernel size, dropout rate, and L2 regularization. These are all hyperparameters that can significantly affect the performance of the model. For example, the number of filters and kernel size can affect the complexity of the learned features, the dropout rate can help prevent overfitting, and L2 regularization can help keep the model weights small.

Training Hyperparameters: Researchers used a specific learning rate, batch size, and number of epochs for training the model. These hyperparameters control the learning process. For example, the learning rate controls how much the model changes in response to the estimated error each time the model weights are updated, the batch size affects the speed and stability of the learning process, and the number of epochs determines how long to train the model.

Ensemble Method: Researchers used an ensemble of multiple models combined with an XGBoost model. This ensemble method can improve accuracy by leveraging the strengths of each individual model and making a final prediction that is more robust and less prone to errors.

SVM and Random Forest Hyperparameters: In the SVM and Random Forest models, researchers used specific hyperparameters like 'C', 'kernel', and 'gamma' for SVM and 'n_estimators', 'max_depth', 'min_samples_split', 'min_samples_leaf' for Random Forest. These hyperparameters can significantly affect the performance of these models.

By understanding and tuning these hyperparameters, researchers are able to improve the performance of the model and achieve higher accuracy compared to the baseline model.

3.2 Model Deployment/Testing

A song titled "One Day," known for its reggae genre, was uploaded for classification with a length of 3.20 minutes (see Figure 31). The results indicate that both models correctly identified the genre as reggae. However, there are noticeable differences in the predicted probabilities assigned to other genres. For the baseline CNN model, while reggae was correctly identified as the predicted genre, there are notable percentages assigned to other genres as well, such as blues (11.01%), rock (7.10%), and country (10.06%) and prediction time of 0.4623 seconds. This suggests that the baseline model may have had some difficulty in confidently distinguishing reggae from other genres, leading to higher probabilities assigned to multiple categories. In contrast, the ensemble model exhibits a more focused prediction, with reggae being assigned a significantly higher probability of 80.04%. The percentages assigned to other genres are comparatively lower and much faster compared to the baseline with a speed of 0.000031 seconds, indicating a stronger confidence in the prediction of reggae as the genre of the song "One Day." This suggests that the ensemble model has enhanced its ability to accurately also much faster in identifying and classifying music.

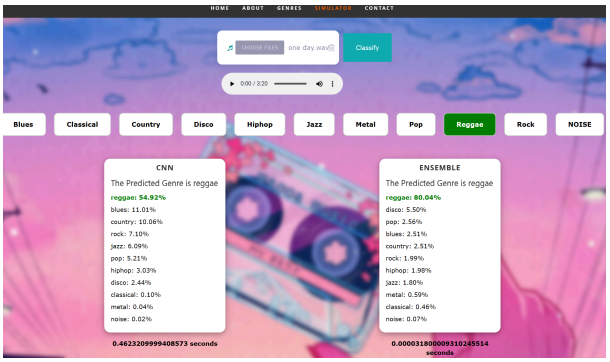

Figure 31. Predict Genre (Reggae)

Researchers test another Audio to classify its genre with a length of 4.29 minutes (see Figure 32). the genre classification results of two different models - a Baseline CNN model and an ensemble model - when provided with a classical music input file. For the Baseline CNN model, the predicted genre is classical with a probability of 28.62%. However, it also shows relatively high probabilities for other genres like "blues" (26.40%), "pop" (13.76%), and "rock" (3.24%), indicating some uncertainty or confusion in the classification and the speed of its prediction is 0.5779 seconds. On the other hand,

the ensemble model predicts the genre as "classical" with a much higher probability of 95.88%. This model seems to have correctly identified the classical nature of the input file with significantly higher confidence compared to the regular CNN model with a speed of prediction of 0.00004 seconds. This suggests that the ensemble model has learned to better discriminate and distinguish the classical genre from other genres and is much faster in prediction.

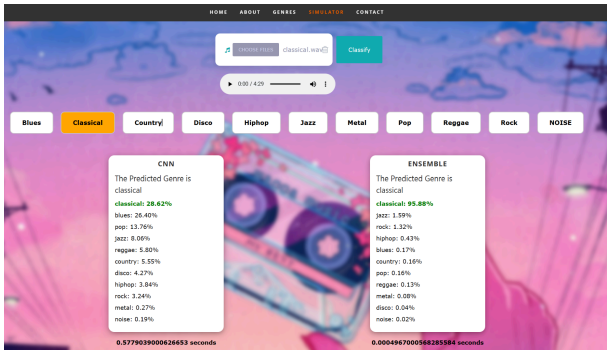

Figure 32. Predict Genre (Classical)

When researchers test an audio file containing the bark of a dog (see Figure 33), it would ideally expect the models to classify it as "noise" or some other non-music category. However, the ensemble model, having learned more robust features to distinguish noise from music, predicts the genre as "noise" with a much higher probability of 95%. It assigns very low probabilities to actual music genres, demonstrating its improved ability to accurately identify non-music audio inputs like dog barks. Conversely, the Baseline model, predicts the genre as "disco" with a probability of 50.07%. This shows relatively high probabilities and incorrect genre prediction, indicating confusion or a lack of clear separation between noise and music genres. This scenario further highlights the superiority of the optimized model in accurately classifying both music genres and non-music audio inputs. By accurately handling non-music inputs like dog barks and confidently classifying them as noise, the ensemble model showcases its improved generalization capabilities and robustness, making it a more reliable tool for real-world audio classification tasks.

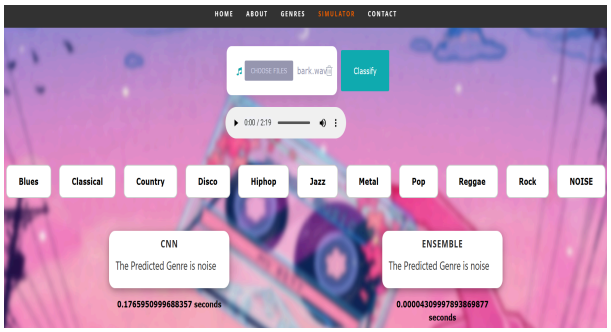

Figure 33. Predict Genre (Noise/Non-genre)

In an experiment, researchers tested an audio file that was a blend of classical music with crowd noise with a length of 4.52 minutes (see Figure 34). The ensemble model successfully identified the primary genre as classical with a confidence of 75.88% and also detected the presence of noise with a confidence of 20.02% with a time of prediction of 0.00002 seconds. On the other hand, the baseline Convolutional Neural Network (CNN) model was also able to recognize the classical genre, but with a slightly lower confidence of 68.33%. Interestingly, it misclassified the noise as jazz with a confidence of 16.11%, indicating that it struggled to accurately detect the noise in the audio. In fact, the baseline model only recognized the noise with a minimal confidence of 0.25% with a time of prediction of 0.1146 seconds.

These results highlight the superior performance of the ensemble model and demonstrated robustness by accurately identifying the classical genre and detecting the presence of noise in the audio file and the speed of its prediction in handling complex audio scenarios compared to the baseline CNN model.

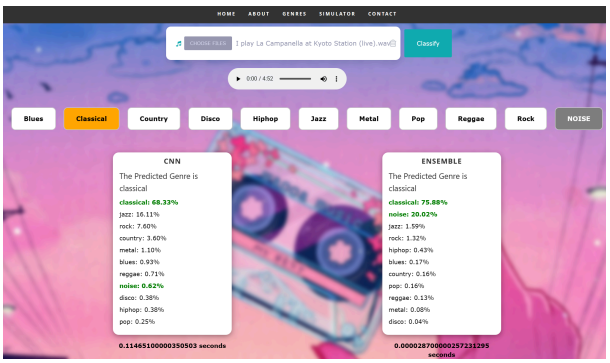

Figure 34. Predict Genre (Classical Genre with Crowd Noise)

The results presented in Figures 31, 32, 33, and 34 indeed demonstrate the significant performance improvements achieved by the ensemble model over the Baseline CNN model in classifying music genres and distinguishing noise from music. This superior performance holds promising implications for real-world applications, particularly in scenarios where accurate genre classification and noise detection are crucial. If commercialized, this model could potentially revolutionize the field of audio classification.

In Figure 33, the ensemble model's robustness and superior feature learning capabilities are further highlighted. When tested with an audio file containing a dog's bark, the ensemble model accurately classified it as "noise" with a high confidence of 95%. This demonstrates the model's ability to distinguish non-music audio inputs effectively. In contrast, the Baseline model incorrectly predicted the genre as "disco" with a probability of 50.07%, indicating a lack of clear separation between noise and music genres in its learning.

Figure 34 presents another interesting scenario where an audio file blending classical music with crowd noise was tested. The ensemble model successfully identified the primary genre as classical with a confidence of 75.88% and also detected the presence of noise with a confidence of 20.02%. The Baseline CNN model, while able to recognize the classical genre, did so with a lower confidence of 68.33% and misclassified the noise as jazz with a confidence of 16.11%. This further underscores the ensemble model's superior performance and robustness in handling complex audio scenarios.

These results demonstrate the ensemble model's enhanced generalization capabilities, making it a more reliable tool for real-world audio classification tasks. Its ability to accurately handle music genres and non-music audio inputs, such as dog barks and crowd noise, showcases its robustness and potential for commercial application. The ensemble model's superior performance over the Baseline CNN model in these tests suggests that it could provide significant advancements in the field of music genre classification.

In real-world scenarios, such as music streaming platforms or audio content recommendation systems, accurate genre classification plays a crucial role in enhancing user

experience. By deploying the ensemble model, these platforms can offer more precise genre-based recommendations tailored to individual user preferences. Moreover, the model's ability to accurately identify non-music inputs as noise can improve the quality of content filtering, ensuring that irrelevant or undesirable audio content, such as background noise or non-music sounds, is appropriately categorized and filtered out.

Commercialization of the music genre classification using ensemble learning can also extend to industries beyond entertainment. For instance, accurate classification of non-music audio inputs, such as dog barks or other environmental noises, can enhance security and automation functionalities in surveillance systems or smart home devices. By reliably distinguishing between music genres and non-music audio inputs, the model can contribute to more efficient and effective decision-making processes in various real-world applications.

4. Conclusion and Recommendation

4.1 Conclusion

In conclusion, this thesis has presented the development and ensembled model tailored for music genre classification. The overarching objective of enhancing accuracy and efficiency in genre identification has been successfully realized through a systematic exploration of various methodologies and optimizations.

The ensemble model, informed by insights from Random Forest and Support Vector Machine models, has yielded impressive results. Achieving the highest accuracy of 84% with a substantial improvement of nearly 10% over the baseline model has an accuracy of 74%. It validates the effectiveness of researchers' approach in accurately discerning music genres. The robustness and generalization capabilities of the ensemble model are further evidenced by its ability to identify a genre and a non-genre.

Researchers tested several audios with different lengths of speed of prediction time of baseline and ensemble model, where the result outperformed the baseline in all tests and its accuracy where it showcases that ensemble is much faster than baseline cnn model in prediction time.

By proficiently learning and extracting relevant features from audio data, complemented by its deep learning capabilities, the ensembled model emerges as a promising solution for real-world applications in music recommendation systems, content tagging, and beyond. Moreover, this research contributes to the broader field of audio analysis by highlighting the significance of advanced machine learning techniques in addressing complex tasks like music genre classification. Through methodical experimentation and analysis, researchers enhance existing methodologies and provide valuable insights for future research endeavors.

However, it is imperative to acknowledge the limitations encountered during this study, such as dataset size constraints, computational resource limitations, and the complexity of the model. These limitations present avenues for further investigation and underscore the need for ongoing research in this domain.

Overall, the success of ensemble models in music genre classification signifies the potential of machine-learning approaches to revolutionize the music industry. This thesis catalyzes further innovation and exploration in this exciting field of study, ultimately contributing to the advancement of audio analysis and its practical applications.

4.2 Recommendation

In addition to exploring different audio formats, consideration should be given to incorporating domain-specific features and metadata. Features such as artist information, song length, tempo, and key signature can enrich the model's understanding of music genres by capturing additional contextual information beyond raw audio signals. The model can potentially achieve greater discriminative power and robustness in genre classification tasks by augmenting the feature space with domain-specific attributes.

Furthermore, leveraging ensemble techniques such as combining multiple models with XGBoost can offer significant performance gains. XGBoost, known for its effectiveness in handling structured data and categorical features, can complement the deep learning capabilities of CNNs by incorporating ensemble diversity and capturing complex relationships within music data. By harnessing the strengths of CNN, SVM, RF, and XGBoost, the ensemble model can achieve superior performance and resilience to variability in audio characteristics.

By incorporating these recommendations into future iterations of the ensemble model, researchers can continue to push the boundaries of music genre classification and explore new frontiers in applications such as music recommendation systems, content tagging, and beyond.

5. REFERENCES

[1] L. Aguiar and J. Waldfogel, "As streaming reaches flood stage, does it stimulate or depress music sales?" International Journal of Industrial Organization, vol. 57, pp. 278–307, Mar. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167718717301753. [Accessed: 2023].

[2] S. K. Prabhakar and S.-W. Lee, "Holistic Approaches to Music Genre Classification using Efficient Transfer and Deep Learning Techniques," Expert Systems with Applications, vol. 211, p. 118636, Jan. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422016815. [Accessed: 2023].

[3] E. Motamedi, D. K. Kholgh, S. Saghari, M. Elahi, F. Barile, M. Tkalcic, "Predicting movies' eudaimonic and hedonic scores: A machine learning approach using metadata, audio and visual features," Information Processing & Management, vol. 61, no. 2, p. 103610, 2024, ISSN 0306-4573. [Online]. Available: https://doi.org/10.1016/j.ipm.2023.103610. [Accessed: 2023].

[4] Q. Kong, X. Feng, and Y. Li, "Music Genre Classification Using Convolutional Neural Network," in Proceedings of the 2023 4th International Conference on Computing and Communication Systems (I3CS), Shillong, India, 2023, pp. 1-5. [Online]. Available: https://ieeexplore.ieee.org/document/10127554 [Accessed: 2023].

[5] J. Lee and J. H. Lee, "Simultaneous extraction of intra- and inter-cycle features for predicting lithium-ion battery's knees using convolutional and recurrent neural networks," Applied Energy, vol. 356, p. 122399, 2024, ISSN 0306-2619. [Online]. Available: https://doi.org/10.1016/j.apenergy.2023.122399.

[6] M. Ashraf, F. Abid, and I.U. Din, "A Hybrid CNN and RNN Variant Model for Music Classification," Appl.

Sci., 2023, 13(3), 1476 [Online]. Available: https://doi.org/10.3390/app13031476.

[7] X. Ying, "An Overview of Overfitting and its Solutions," Journal of Physics: Conference Series, vol. 1168, no. 2, p. 022022, June 2019. [Online]. Available: 10.1088/1742-6596/1168/2/022022.

[8] Y. Lett and R. Kyaw, "Music Genre Classification Using Support Vector Machine Techniques," 2023 International Conference on Information Management and Technology (ICIMTech), Malang, Indonesia, 2023, pp. 511-516. [Online]. Available: https://ieeexplore.ieee.org/document/10277842.

[9] H. Linus son, "Multi-Output Random Forest Regression," in 2021 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, USA, 2021, pp. 1-6. [Online]. Available: https://ieeexplore.ieee.org/document/9483749.

[10] R. Mayer and A. Rauber, "MUSICAL GENRE CLASSIFICATION BY ENSEMBLES OF AUDIO AND LYRICS FEATURES," 12th International Society for Music Information Retrieval Conference (ISMIR 2011). Available: https:/api.semanticscholar.org/CorpusID:7987239.

[11] A. Acharya, "Fair train-test split in machine learning," Journal of Petroleum Science and Engineering, vol. 209, p. 109885, 2022. [Online]. Available: https://doi.org/10.1016/j.petrol.2021.109885.

[12] J. Foleis and T. Tavares, "Texture selection for automatic music genre classification," Applied Soft Computing, vol. 89, p. 10-27, 2020. [Online]. Available: https://doi.org/10.1016/j.asoc.2020.106127.

[13] H. Chu, “A CNN Sound Classification Mechanism Using Data Augmentation,” 2023, 23(15), 6972. [Online]. Available: https://doi.org/10.3390/s23156972.

[14] T. de Oliveira Nogueira and G. B. A. Palacio, "Imbalance classification in a scaled-down wind turbine using radial basis function kernel and support vector machines," Energy, vol. 238, Part C, p. 122064, 2022, ISSN 0360-5442. [Online]. Available: https://doi.org/10.1016/j.energy.2021.122064.

[15] J. Park, Y. Choi, and J. Byun, "Efficient differentially private kernel support vector classifier for multi-class classification," Information Sciences, vol. 619, pp. 889-907, 2023, ISSN 0020-0255. [Online]. Available: https://doi.org/10.1016/j.ins.2022.10.075.

[16] I. Jamaleddin, R. El Hayashi, and M. Biniz, "An improved approach to Arabic news classification based on hyperparameter tuning of machine learning algorithms," Journal of Engineering Research, vol. 11, no. 2, pp. 100061, 2023, ISSN 2307-1877. [Online]. Available: https://doi.org/10.1016/j.jer.2023.100061.

[17] H. Wang, S. S. Jamali, Z. Chen, Q. Shan, and J. Ren, "An intelligent music genre analysis using feature extraction and classification using deep learning techniques," Computers and Electrical Engineering, vol. 100, pp. 107978, 2022, ISSN 0045-7906. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2022.107978.

[18] S. Zheng, F. Lan, and M. Castellani, "A competitive learning scheme for deep neural network pattern classifier training," Applied Soft Computing, vol. 146, p. 110662, 2023, ISSN 1568-4946. [Online]. Available: https://doi.org/10.1016/j.asoc.2023.110662.

[19] H. Liu, Z. Zhong, N. Sebe, and S. Satoh, "Mitigating robust overfitting via self-residual-calibration regularization," Artificial Intelligence, vol. 317, p. 103877, 2023, ISSN 0004-3702. [Online]. Available: https://doi.org/10.1016/j.artint.2023.103877.

[20] Xiuyan Liu, Chen Chen, and Yongjun He, "Temporal feature extraction based on CNN-BLSTM and temporal pooling for language identification," Applied Acoustics, Volume 195, 2022, 108854, ISSN 0003-682X. [Online]. Available: https://doi.org/10.1016/j.apacoust.2022.108854.

[21] Md. S. Chowdhury, "Comparison of accuracy and reliability of random forest, support vector machine, and artificial neural network in classification of urban setting," Environmental Challenges, vol. 14, p. 100800, 2024, ISSN 2667-0100. [Online]. Available: https://doi.org/10.1016/j.envc.2023.100800.

[22] Z. Sun, G. Wang, P. Li, H. Wang, M. Zhang, and X. Liang, "An improved random forest based on the classification accuracy and correlation measurement of decision trees," Expert Systems with Applications, vol. 237, Part B, p. 121549, 2024, ISSN 0957-4174. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.121549.

[23] Y. Singh and A. Biswas, "Robustness of musical features on deep learning models for music genre classification," Expert Systems with Applications, vol. 199, p. 116879, 2022, ISSN 0957-4174. [Online]. Available: https://doi.org/10.1016/j.eswa.2022.116879.

[24] R. Pérez and J. Bajorath, "Evolution of Support Vector Machine and Regression Modeling in Chemoinformatics and Drug Discovery," J Comput Aided Mol Des, 36, 355–362 (2022). https://doi.org/10.1007/s10822-022-00442-9.

[25] Z. Rustam, E. Sudarsono, and D. Sarwinda, "Random-Forest (R.F.) and Support Vector Machine (SVM) Implementation for Analysis of Gene Expression Data in Chronic Kidney Disease (CKD)," Volume 546, Issue 5, 2019. https://iopscience.iop.org/article/10.1088/1757-899X/546/5/052066.

[26] L. Hong, Z. Chen, Y. Wang, M. Shahidehpour, and M. Wu, "A novel SVM-based decision framework considering feature distribution for Power Transformer Fault Diagnosis," Energy Reports, vol. 8, pp. 9392-9401, 2022, ISSN 2352-4847. [Online]. Available: https://doi.org/10.1016/j.egyr.2022.07.062.

[27] V. Ziyatdinov and M. Tereshonok, "Noise Immunity and Robustness Study of Image Recognition Using a Convolutional Neural Network," Feb 6 22. [Online]. Available: 10.3390/s22031241.

[28] S. Mazumder, "5 Techniques to Handle Imbalanced Data for a Classification Problem," Published online 2022 Feb 6. [Online]. Available: 10.3390/s22031241.

[29] S. A.S., J. B. Mala, and R. Rajan, "Automatic music mood classification using multi-modal attention framework," Engineering Applications of Artificial Intelligence, vol. 128, p. 107355, 2024, ISSN 0952-1976. [Online]. Available: https://doi.org/10.1016/j.engappai.2023.107355.

[30] T. Sadeh and M. Moscovitch, "Retrieval of temporal structure at recall can occur automatically," Cognition, vol. 242, p. 105647, 2024, ISSN 0010-0277. [Online]. Available: https://doi.org/10.1016/j.cognition.2023.105647.