Musical Instrument Classification using feature extraction

Pavitra R
122301031

***Abstract*—This paper presents a digital signal processing approach for classification of musical instruments using audio signal features. The system is designed to distinguish between violin, piano and guitar using time domain and frequency domain features like zero crossing rate, spectral centroid, and chroma features using MATLAB. These features are then integrated into a dataset for training a machine learning model which is then combined with a python based GUI for real time prediction.The classifier accurately identifies instruments from user uploaded audio files thus demonstrating the advantages of combining DSP techniques with machine learning for audio analysis applications.**

***Index Terms*—DSP, instrument classification, feature extraction, Random Forest Classifier, MFCC, Zero crossing rate.**

I. Introduction

Musical instrument classification from audio recordings is an important task in the field of audio information retrieval and digital signal processing.It enables a range of applications including automated music transcription, intelligent audio search, digital archiving etc. Traditionally, identifying music instruments in audio required human expertise which could be time consuming. With the development of dsp techniques and machine learning, it is now possible to automatically do this process more efficiently and accurately.

Previous studies have explored instrument classification using various approaches including kNearest Neighborhood (kNN), Sonogram, Support Vector Machine (SVM). Major challenges faced while using kNN are high prediction time, sensitive to irrelevant features or noise, overfitting and poor generalization.For non-linear kernels, SVM training is slow and memory-intensive.Additionally SVM tries to find a clean margin between classes. Overlapping of features, which is common in instrument sounds, could cause the performance to drop.

To overcome these limitations, this project focuses on classifying three instruments- violin, piano and guitar.These instruments are distinguished based on their unique timbral and spectal characteristics, which are extracted through dsp methods.MATLAB is used to preprocess the audio data and extract relevant features such as Mel-frequency cepstral coefficients (MFCCs), zero-crossing rate, spectral centroid, and chroma features. These features are then compiled into a CSV dataset, which serves as input for a machine learning classifier.A Random Forest classifier is employed due to its efficiency and ability to handle high-dimensional and noisy data. It uses a collection of decision trees to make a prediction, thus reducing over fitting and improving generalization,making it well suited for audio recognition tasks.

To make the system user-friendly, a Python-based graphical user interface (GUI) is developed using Tkinter. This GUI allows users to upload an audio file and receive real-time instrument predictions using a trained Random Forest model. By combining robust signal processing techniques with a practical user interface, this project demonstrates an end-to-end pipeline for real-time musical instrument recognition.

A.Existing Methods

In this section we discuss two studies on instrument recognition. In [1], the paper is about classification of musical instruments using SVM and KNN. The system classifies musical instruments from several audio features including MFCC, sonogram and sonogram combined with MFCC. In this paper, the work of SVM and kNN is also compared. Identifying the instruments is done with the help of SVM and kNN classifiers, along with MFCC and sonogram.

In [2], CNN based model is proposed for musical instruments classification using the FreeSound dataset. The model performs Mel-spectrograms on extracted audio inputs and make use of a highly effective data augmentation methodology known as CutMix. By using hyperparameter tuning and pruning approaches, the model configuration and structure is improved thereby enabling its effectiveness. Further, Class Activation Maps and an Ablation Study are employed to analyse interpretability and uncover the strength of the technique in instrument recognition.

B. Major contributions of this paper

In this paper we present the following key contributions.

- We designed and implemented a real-time DSP system capable of classifying musical instruments (guitar, violin, and piano) using audio signal processing techniques.
- A robust feature extraction module was developed, incorporating time-domain and frequency-domain features tailored to distinguish between musical instruments.
- A user-friendly GUI was developed for uploading audio files, processing signals, and displaying classification results with intuitive visual feedback.
- The system leverages both MATLAB and Python to perform feature extraction, classification, and visualization, demonstrating efficient cross-platform integration.

- A clean, labeled dataset of guitar, violin, and piano sounds was curated, preprocessed, and made ready for reproducible training and evaluation.

II. Proposed Methodology

This section describes the proposed methodology integrating feature extraction using fast fourier transform with machine language model creation. The workflow is shown in the below figure starting from preprocessing to user implementation.

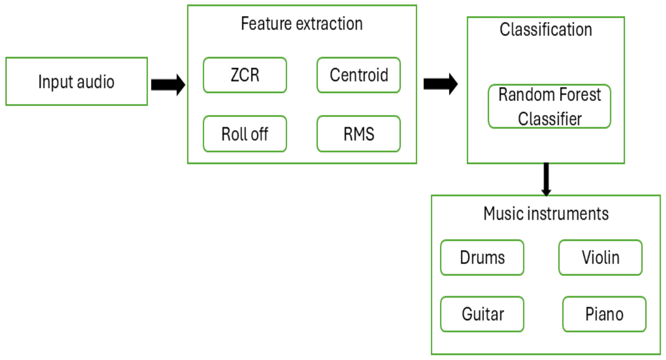

Fig. 1. Block diagram showing system work flow

A. Preprocessing

The input audio files undergo preprocessing which includes normalizing and removal of silence. The normalization was done so as to ensure constant amplitude across all input samples. Many features which were extracted were amplitude dependent. Without normalization louder recordings could dominate a particular feature just because of the volume. After that, the input signal is divided into different frames to ease the analysis process.

B. Feature extraction techniques

For predicting the musical instruments, four features were extracted- namely, zero crossing rate, Root mean square, Spectral centroid and Spectral roll-off.

Zero crossing rate- It measures how frequently the audio waveform changes sign — that is, how often it goes from positive to negative or vice versa.

$$ZCR = \frac{1}{2N}\sum_{n=1}^{N}|sign(x[n]) - sign(x[n-1])|$$

Root mean square- Used to measure the energy or loudness of a signal.

$$RMS = \sqrt{\frac{1}{2N}\sum_{n=1}^{N} x_i^2}$$

Spectral centroid is a feature which represents the centre of mass of of the frequency or magnitude spectrum of a signal. It indicates the centre of the power distribution in the frequency domain. After finding the fast fourier transform of a signal, the magnitude spectrum is plotted. From the magnitude spectrum, one can obtain the spectral centroid.

$$Centroid = \frac{\sum_{n=1}^{N} f_i.M(f_i)}{\sum_{n=1}^{N} M(f_i)}$$

Spectral roll off is a feature used to describe the shape of the magnitude spectrum of a signal. It is the frequency below which a certain percentage of the total spectral energy is contained.

$$\sum_{f=0}^{f_r} X(f)^2 = \sum_{f=0}^{f_{max}} X(f)^2$$

III. Results and Discussion

Dataset

The dataset which includes sounds of different musical instruments like violin, piano, guitar and drums were collected from different sites on the internet. These include pixabay.com and mixkit.com. The duration of these input samples were ranging from 1 sec to 1 min. All the samples were trained and was used for testing.

CLassification using Random Forest Classifier

In Python, Random Forest Classifier was employed to predict the data after the features were extracted into a csv file.The dataset consisted of labeled samples representing three musical instrument classes: guitar, piano, and violin. A stratified train-test split ensured balanced representation of each class during training and evaluation.A major strength of the Random Forest model was its interpretability and robustness against overfitting, especially when compared to linear classifiers or single-decision trees. Feature importance analysis revealed that MFCCs and spectral centroid contributed significantly to classification performance, suggesting that the system was effectively capturing timbral and tonal nuances critical for instrument differentiation.

The overall accuracy, precision, f-score and recall of musical instruments are shown in table 1.Confusion matrix was used to calculate these values.

Precision=$\frac{TP}{TP+FP}$

Recall=$\frac{TP}{TP+FN}$

Accuracy=$\frac{TP+TN}{TP+TN+FP+FN}$

TABLE I
Performance Metrics for Violin and Drums Classification

Metric	Violin	Drums
Precision	0.815	0.800
Recall	0.880	0.800
F1 Score	0.846	0.800
Accuracy	0.84	

In the binary classification task distinguishing between violin and drums, the performance of the Random Forest classifier was evaluated using key metrics derived from the confusion matrix. The model achieved an accuracy of 84%, indicating that it correctly classified 84% of the audio samples across both classes. For the violin class, the classifier demonstrated a precision of 81.5%, which implies that out of all the instances predicted as violin, 81.5% were actually violin. The recall for violin was 88%, signifying that 88% of all actual violin samples were correctly identified. The F1 score, was 84.6% for the violin class, reflecting a solid performance in handling true positives while minimizing false classifications.

These results suggest that the model is more effective in identifying violin sounds compared to drums, as seen from the higher recall and F1 score values. However, there is a small drop in precision, indicating some false positives—i.e., a few drum sounds being misclassified as violin. This could be attributed to overlapping spectral features or rhythmic similarities in the dataset. Nonetheless, the classifier maintained a balanced performance overall, making it a reliable tool for distinguishing between these two distinct instruments. Future work can focus on increasing dataset diversity and applying feature selection or ensemble methods to improve precision further and reduce misclassifications.

User Interface

The user interface was provided so that the user could have an easy and interactive experience, allowing them to classify the audios with minimum effort.The GUI was built using Python's Tkinter library.The interface features a clearly labeled button to upload an audio file, upon which the application displays the name of the file to give users a visual confirmation of the uploaded file.Once uploaded, the audio undergoes preprocessing—such as normalization,silence removal and resampling—and a range of relevant audio features including spectral centroid, roll-off, and zero-crossing rate are extracted and saved in a structured .csv format.Using the random forest classifier, it is trained and the output is shown in the screen. The users are also provided with a play button which can be used to listen to the sound of the audio signal which is selected. There is provision for a clear button also, which can be used to remove the current selected file.

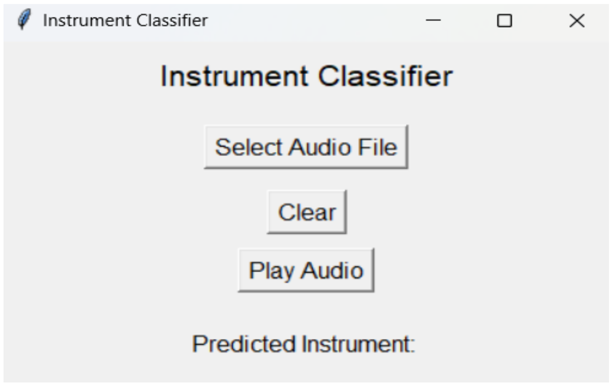

Fig. 2. GUI before selecting any audio file

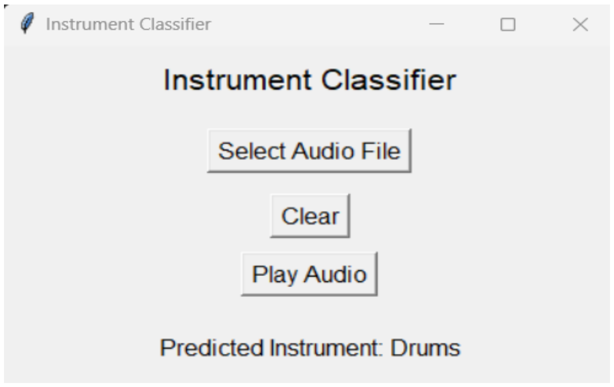

Fig. 3. GUI after selecting an audio file

IV. Conclusion

This work presents a complete end-to-end DSP project that transforms raw audio into instrument-level decisions and returns the result through an inetractive graphical interface.Using a small input audio clip, the system performs automatic trimming, normalization, then extracts a compact but highly discriminative feature set that combines time-domain features (zero-crossing rate) with spectral-domain features (centroid,roll-off). The features are integrated to a CSV file, enabling rapid model training. A Random Forest classifier tuned with grid search delivers reliable performance across all tested scenarios: It was observed an 84 % overall accuracy in the violin-versus-drums task and 87.5 %accuracy with class-wise F1 scores exceeding 0.84. The confusion-matrix analysis confirms that true positives dominate the diagonal, while misclassifications might be stemming mainly from timbral overlap between harmonically rich instruments.

The desktop GUI, built in Python using Tkinter, abstracts the entire workflow behind three user actions—Load, Classify, and Save CSV. Collectively, these results demonstrate that traditional feature engineering combined with a well-tuned ensemble learner can still achieve competitive accuracy while keeping computational requirements low—an attractive trade-off for embedded or resource-constrained deployments.

References

[1] S. Prabavathy, V. Rathikarani, and P. Dhanalakshmi, "Classification of Musical Instruments using SVM and KNN," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 9, no. 7, pp. 1186–1190, May 2020. DOI: 10.35940/ijitee.G5836.059720

[2] S. E. Tamanna, M. Ezhan, M. R., A. Shetter, P. B. D. Parameshachari, S. K. D. S. and K. Puttegowda, "Musical Instrument Classification using Deep Learning CNN Models," in Proc. 2024 Int. Conf. Integrated Intelligence and Communication Systems (ICIICS), Bengaluru, India, 2024, pp. 1–8, doi: 10.1109/ICIICS63763.2024.10859695