Musical Instrument Classification using Audio Features and Convolutional Neural Network

Gst. Ayu Vida Mastrika Giri [1]*, Made Leo Radhitya [2]**
* Informatika, Universitas Udayana
** Teknik Informatika, Institut Bisnis dan Teknologi Indonesia
vida@unud.ac.id [1], leo.radhitya@instiki.ac.id [2]

Article Info

Article history:

Received 2024-06-30
Revised 2024-07-15
Accepted 2024-07-16

Keyword:

Audio Signal Processing, Convolutional Neural Network, Mel-Frequency Cepstral Coefficients, Music Information Retrieval, Musical Instrument Classification.

ABSTRACT

This research classifies acoustic instruments using Convolutional Neural Network (CNN). We utilize a dataset from Kaggle containing audio recordings of piano, violin, drums, and guitar. The training set consists of 700 guitar, percussion, violin, and 528 piano samples. The test set contains 80 samples of each instrument. Features such as Mel spectrograms, MFCCs, and other spectral and non-spectral characteristics are extracted using the Librosa package. Three feature sets—spectral-only, non-spectral-only, and a combined set—are employed to evaluate the efficacy of CNN models. Various CNN configurations are tested by adjusting the number of convolutional filters, learning rates, and epochs. The combined feature set achieves the highest performance, with a validation accuracy of 71.8% and a training accuracy of 76.9%. In comparison, non-spectral features achieve a validation accuracy of 68.4%, and spectral-only features achieve 69.3%. These findings highlight the benefits of using a comprehensive feature set for accurate classification.

I. Introduction

Classifying musical instruments based on audio recordings is a fundamental challenge in Music Information Retrieval (MIR). There are practical applications of the classification of musical instruments in the development of musical aids, the recognition of sound patterns, and educational tools. Precise classification can help to improve interactive experiences in music education, enable automatic music transcription, and strengthen music recommendation systems. Nevertheless, the complex structure of audio signals containing spectral and temporal features poses considerable difficulties. Traditional methods frequently need help capturing these complex subtleties, resulting in below-optimal classification performance.

Previous research has explored various methods for classifying musical instruments, with early methods primarily relying on classical machine-learning algorithms and handcrafted features. Spectral feature-based methods, as utilized in the research by [1], employed the IRMAS dataset for instrument classification. The classification techniques included logistic regression, decision tree classifiers, Support Vector Machine (SVM) classifiers, and unsupervised algorithms like K-means and hierarchical clustering. Among these, SVM outperformed the others, achieving an accuracy of 79%. In another study by [2], Indian musical string instruments, particularly struck and plucked instruments, were classified using audio features, feature selection techniques (MANOVA and Chi-square), and SVM classifiers with different kernels. The highest accuracy of 93.4% was achieved using the top 5 features selected by MANOVA and an SVM with a linear kernel. A different approach was proposed by [3], which used Mel Frequency Cepstral Coefficients (MFCC) as features and Gaussian Mixture Models (GMM) as the classification algorithm, achieving an overall accuracy of 86.98% in classifying ten different musical instruments.

Deep learning methods have shown outstanding results in audio classification tasks in recent years. For instance, the YOLOv7 model used by [4] distinguished similar musical instruments and evaluated performance on the PPMI dataset of 12 musical instrument classes, achieving the highest average accuracy of 86.7%. An RNN model combined with MFCC features, as used by [5], classified the emotions of musical instruments, demonstrating that the combination of MFCC and RNN provided greater accuracy in recognizing instrument emotion, with an accuracy rate of 89.3%.

Machine learning and deep learning models have been examined for monophonic instrument recognition. Research by [6] compared K-NN, SVM, GMM, Artificial Neural Network (ANN), Convolutional Neural Networks (CNN), and RNN. Among these, CNN was the most accurate model for instrument classification, achieving a 96.82% accuracy rate. Additionally, the CNN model outperformed a previous feedforward neural network model in [7], achieving higher F1 scores across classes and an excellent ROC-AUC score of 0.999 for classifying 20 musical instrument classes using MFCC as input features. These experiments demonstrate that CNN can efficiently capture spectral and temporal patterns in audio data, outperforming conventional techniques.

Mel-frequency cepstral Coefficients (MFCCs), chroma features, and spectral descriptors have been extensively used due to their ability to represent instruments' timbral characteristics accurately. For instance, MFCC proved more effective than the power spectrum in identifying Chinese musical instruments using Gaussian mixture models (GMM), as shown by [8]. Using MFCC features and CNN, [9] achieved high precision (93%) and F1 score (0.93) in identifying instruments in musical recordings.

Previous research shows that the combination of MFCC and CNN produces higher accuracy in classifying musical instruments than other machine learning methods, such as SVM, or deep learning methods, such as RNN. However, further investigation into feature combinations and network architectures is needed to improve classification accuracy.

This research aims to create and examine a CNN model capable of classifying acoustic musical instruments based on audio features, thereby significantly contributing to the Music Information Retrieval (MIR) field. At first, it aims to determine the most influential architecture of CNN for classifying musical instruments by conducting experiments with various combinations of spectral and temporal audio data. Furthermore, it analyzes the influence of several feature sets, such as exclusively spectral and exclusively temporal features, and their combination on the effectiveness of a CNN classification model.

This research offers several advantages. The proposed method aims to achieve higher classification accuracy by systematically exploring and choosing feature combinations and CNN configurations. The outcomes of this research are expected to improve the accuracy and applicability of instrument recognition systems significantly.

II. Methods

This research used an organized approach, shown in Figure 1, to develop and test a Convolutional Neural Network (CNN) model for classifying musical instruments using audio recordings. The approach begins with dataset collection, followed by a series of preprocessing processes to extract relevant features and handle class imbalances; after preprocessing, the CNNs Model was constructed, the features were selected, and its performance was evaluated.

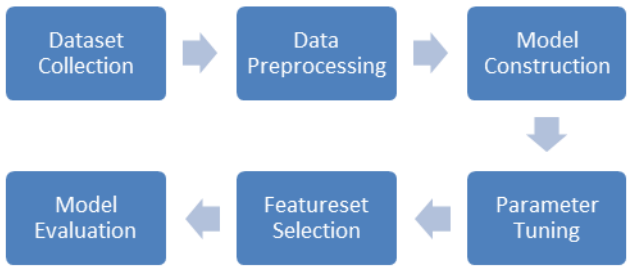


Figure 1. Research Overview

A. Dataset Collection

The dataset utilized in this research was obtained from Kaggle and includes audio recordings in WAV format for four different musical instruments: guitar, drums, violin, and piano [10]. The dataset was compiled from collecting data from the Pixabay site, as well as from a few other sites and open-source datasets. The training set consists of 700 sounds for guitar, 700 for drums, 700 for violin, and 528 for piano, as shown in Figure 2. The test set consists of 80 audio files, with 20 from each class.

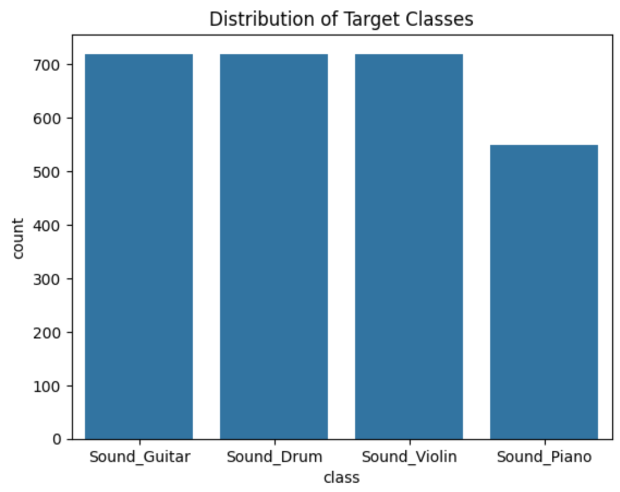


Figure 2. Dataset Label Distribution

The sound files in this dataset have different lengths, sample rates, tracks, and widths. The lengths of these audio files are different. The smallest clip is 1.06 seconds long, and the longest is 128.06 seconds long. The length of the audio files is, on average, 20.86 seconds. The files also vary significantly in sample rate and the number of audio clips sent per second. The slowest sample rate can be taken is

16,000 Hz, and the fastest is 48,000 Hz. The average sample rate for all files is about 42,734.86 Hz, which shows that the dataset has a range of sound quality. Regarding the number of audio channels, most files are mono (single channel), with one channel being the smallest and two being the largest. The files have an average of 1.22 channels, suggesting that most are mono. However, there are also some stereo files. The sample width, which shows the many bits in the audio files, can be anywhere from 2 to 4 bytes. The dataset has an average sample width of 2.13 bytes, which means that most of the files have a bit depth of about 16 bits per sample.

B. Preprocessing

The following steps were undertaken to preprocess the audio data:

1) Feature Extraction

Fourteen features were retrieved from audio sources using the Librosa, Numpy, and Scipy packages, including spectral and non-spectral features. Spectral features can be determined from the frequency domain representation of the acoustic signal. These characteristics offer a glimpse into the sound's timbral characteristics. Non-spectral features either represent the statistical properties of the signal or are derived from the time-domain representation of the audio signal. There are nine spectral features extracted they are Mel Frequency Cepstral Coefficient (MFCC), Chroma Feature, Spectral Centroid, Spectral Bandwidth, Spectral Contrast, Spectral Rolloff, Spectral Flux, Spectral Flatness, and Zero Crossing Rate. Five non-spectral features were extracted: maximum amplitude, minimum amplitude, root mean square (RMS) energy, kurtosis, and skewness.

MFCC is an audio file series of short-term power spectra [11]. MFCC is obtained by applying the `librosa.feature.mfcc` function to the logarithm of the Mel spectrogram. The `feature.melspectrogram` function in librosa computes the Mel spectrogram. It displays audio signals in the Mel scale, which approximates human hearing. Spectrogram settings include Mel band count (64) and frequency range (2000 Hz to half the sampling rate). `librosa.power_to_db` is used to logarithmically scale the Mel spectrogram, standardizing the feature values by limiting the amplitude range. To get the MFCC values, Librosa condenses the spectral envelope of the audio stream, capturing the timbral texture. In this research, 13 Mel-frequency cepstral coefficients (MFCCs) were calculated, and their average, standard deviation, and variance were determined across a series of time frames.

Chroma features represent the 12 distinct pitch classes included in the audio [11]. They are calculated using the `librosa.feature.chroma_stft` function.

The center of mass of the spectrum is determined by calculating the spectral centroid [11]. It is calculated using the `librosa.feature.spectral_centroid` function. The weighted mean of the frequencies in the signal determines the central point of the frequency spectrum. The average value over periods is utilized.

The spectral bandwidth measures the breadth of the spectrum [11] and may be determined using the function librosa.feature.spectral_bandwidth. It offers a concept of the spectrum of frequencies present. The average value is obtained from the time frames.

The calculation of spectral contrast is performed using the `librosa.feature.spectral_contrast` function. It quantifies the variation in amplitude between the highest and lowest points in a sound spectrum, aiding in the differentiation of various sound qualities.

Spectral rolloff refers to the frequency at which a certain percentage of the overall spectral energy is located[11]. The calculation is performed using the `librosa feature.spectral_rolloff function, which offers information about the spectrum's asymmetry`. The average value over periods is utilized.

The Zero Crossing Rate measures the frequency at which the signal changes polarity, showing its rate of oscillation [12]. The computation is performed using the librosa `feature.zero_crossing_rate` function and the resulting mean value is extracted across time frames.

Spectral flatness measures the level of evenly dispersed frequencies in a power spectrum. It is calculated by taking the ratio of the geometric mean to the arithmetic mean of the subbands [13]. The computation is performed using the librosa.feature.spectral_flatness function and the resulting mean value is used.

The RMS energy, which quantifies a signal's loudness, can be calculated using the librosa feature.rms function. The value is the square root of the average of the squared amplitude values [12]. The average root mean square (RMS) value is used across time frames.

The maximum and minimum amplitude features represent the highest and lowest values of amplitude in the signal, which indicate the audio's dynamic range. The maximum and minimum values are calculated using the max and min functions from the Numpy library applied to the audio signal.

Spectral flux measures the speed at which the power spectrum varies, reflecting the magnitude of abrupt variations in the sound. The computation is performed using the `librosa.onset.onset_strength` function, and the resulting mean value is utilized.

Kurtosis quantifies the degree of deviation from a normal distribution in a signal's amplitude distribution. It is computed using the `scipy.stats.kurtosis` function.

Skewness quantifies the degree of asymmetry in a signal's amplitude distribution. It is computed using the `scipy.stats.skew` function.

Features such as MFCCs, chroma, spectral centroid, spectral bandwidth, spectral contrast, spectral roll-off, zero crossing rate, RMS energy, and spectral flux are calculated by taking the average value across different time frames. Normalization decreases variability and accurately depicts the overall characteristics of the audio signal.

1. Class Encoding

The instrument classes were converted into numerical values using a label encoder. This phase guarantees that the categorical labels are transformed into an appropriate format for model training.

2) *Under Sampling*

Class imbalance in a dataset happens when one or more classes have more instances than others (s). A significant difference in the number of majority and minority class instances, such as in a substantial class imbalance, may bias the results of classification tasks [12]. Random Under Sampling (RUS) was used to compensate for the class imbalance in the dataset. Under-sampling aims to balance the class populations by eliminating majority class examples. It consists of randomly picking and eliminating some examples from the majority class until the number of cases in each class is about equal. After resampling, each class had 549 samples, resulting in a balanced training dataset.

3) *Scaling*

Differences in the data range for each feature can cause the data to be poorly distributed and affect the classification results. Data normalization is performed to equalize all feature data ranges obtained. A Min-Max scaler was used to scale the features, bringing the data into the range of 0 to 1, using Equation 1 [13]. This scaling step is required to ensure that all features contribute equally to the model training process.

$$norm\,(x) = \frac{x - minValue}{maxValue - \min value} \quad (1)$$

4) *Data Splitting*

The dataset was divided into training and testing sets in an 80:20 ratio. The training set had 439 samples per class, while the test set featured 110 samples for each class. This split ensures that the model may be efficiently trained while evaluated on a different, unknown test set.

C. *CNNs Model Construction*

The architecture shown in Figure 2 consisted of two convolutional layers, each followed by a max-pooling layer to reduce the spatial dimensions and regulate overfitting. The initial convolutional layer employed a filter size equivalent to the number of evaluated units. Subsequently, the second convolutional layer doubled the number of filters. The ReLU activation function was used for non-linear transformations, and both layers employed a kernel size of 2.

After convolution and pooling, the model was flattened and passed through a dense layer with 50 neurons and a ReLU activation function. A softmax activation function was employed to generate class probabilities in the final layer, composed of four neurons, each corresponding to one of the four instrument classes.

The Adam optimizer was employed to compile the model, adjusting the learning rate by the specified configurations. Sparse categorical cross-entropy was selected as the loss function for our multi-class classification problem. The models were trained on the training dataset with a batch size of 32 and subsequently evaluated on the validation dataset after each epoch to monitor performance and overfitting.

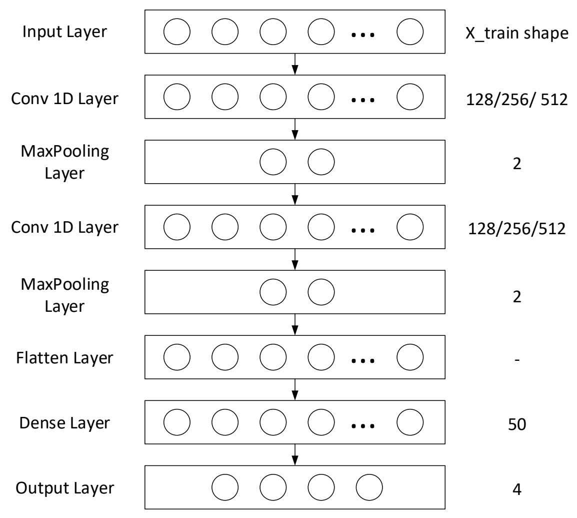


Figure 3. CNNs Model Architecture

D. *Parameter Tuning*

Adjusting the number of convolutional filters, learning rates, and epochs allows for various CNN configurations. To be more precise, we conducted experiments with filter units of 128, 256, and 512, learning rates of 0.0001, 0.001, and 0.01, and training epochs of 50, 75, and 100. By systematically testing these combinations, we identified the optimal parameters for our classification task.

A sequential CNN model was developed for each configuration. Each combination of these parameters was tested separately, doing a full grid search to observe the CNN's performance across many configurations.

E. *Features Selection*

Our research implemented three distinct experiments to assess the influence of diverse audio feature sets on the functionality of our CNN-based musical instrument classification system. The Librosa package was employed to extract the features, and each experiment concentrated on a distinct combination of these features.

The first experiment was exclusively concerned with spectral features. These features are particularly effective in capturing audio signals' frequency content and temporal characteristics, which are essential for distinguishing between various musical instruments. The CNN model was trained and tested using these spectral features to evaluate its standalone effectiveness in the classification assignment.

The spectral features were left out of the second experiment, and instead, other pertinent audio features were employed. These nonspectral features offer supplementary information regarding the audio signal's complexity, dynamics, and distribution. This investigation aimed to ascertain the classification performance when these non-spectral features were exclusively employed.

The third experiment integrated spectral and non-spectral features to establish an exhaustive feature set. This method's objective was to capitalize on the advantages of both types of features, thereby delivering a more comprehensive representation of the audio signals. This research aimed to optimize the CNN model's utilization of information by integrating all available features, which could enhance its classification accuracy and robustness.

The same CNN architecture, shown in Figure 2, was employed in each of these experiments, and the parameters were varied by the previously specified protocols for training and testing. Subsequently, the results were contrasted to ascertain which feature set demonstrated the most effective performance.

F. Model Evaluation

The accuracy and loss were documented during the training procedure for both the training and validation sets. These metrics were plotted to identify potential overfitting or underfitting issues and visualize the learning curves.

The performance of a classifier model can be evaluated by comparing the predicted labels with the actual data labels. This information can be summarized in a table called the confusion matrix. The matrices emphasized the occurrences of correct and incorrect classifications, which facilitated our comprehension of prevalent misclassification patterns and the further refinement of the model. The confusion matrix summarizes the amount of data that was correctly or incorrectly predicted by the classifier using the True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN).

While the confusion matrix already provides the information needed to determine how well a classification model performs, summarizing this information into a single number in the form of an accuracy value makes it easier to compare the relative performance of different models. The accuracy value calculates the overall ratio of the amount of data that correctly predicted the class to the total number of predictions. Equation 2 is used to calculate the accuracy value[14].

$$Accuracy = \frac{Number\ of\ correct\ prediction}{Total\ number\ of\ prediction} \qquad (2)$$

Precision measures how many positive predictions are true positives. The precision value is the ratio between true positives and the amount of data predicted to be positive. Equation 3 can be used to calculate the value of precision [14].

$$Precision = \frac{TP}{TP+FP} \qquad (3)$$

The recall value measures how many positive cases the classifier correctly predicted overall positive cases in the data. It can be calculated by comparing the actual positive value with the amount of positive data. Equation 4 can be used to calculate the recall value [14].

$$Recall = \frac{TP}{TP+FN} \qquad (4)$$

F1-Score or F1-Measure is a measure that combines precision and recall. F1-Score can describe the harmonic mean of precision and recall. A high F1-Score value represents high precision and recall values as well. Equation 5 is used to calculate the F1 Score [14].

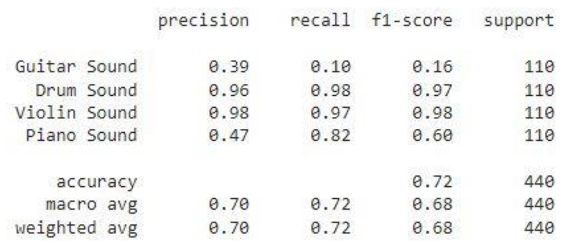
$$F1\ Score = \frac{2 \times TP}{2 \times TP+FP+FN} \qquad (5)$$

The macro average scores were calculated by averaging these metrics across all courses, which allowed for a fair assessment of the model's performance on various instruments.

III. Results and Discussions

This research evaluated the performance of CNN models for musical instrument classification using three different feature sets: all features combined, spectral features only, and non-spectral features only. The results were assessed using a variety of critical metrics, such as macro average precision, recall, F1-score, validation loss, and accuracy.

CNN models demonstrated a remarkable overall performance when employing the complete feature set of spectral and non-spectral features. The optimal configuration for this feature set was attained by utilizing 512 units, a learning rate of 0.01, and 75 epochs, resulting in a training accuracy of 76.9% and a validation accuracy of 71.8%. The macro average precision, recall, and F1-score for this configuration were 70.3%, 71.8%, and 67.7%, respectively, as shown in Figure 4, emphasizing a balanced performance across all classes.



```
              precision    recall  f1-score   support

Guitar Sound       0.39      0.10      0.16       110
  Drum Sound       0.96      0.98      0.97       110
Violin Sound       0.98      0.97      0.98       110
 Piano Sound       0.47      0.82      0.60       110

    accuracy                           0.72       440
   macro avg       0.70      0.72      0.68       440
weighted avg       0.70      0.72      0.68       440
```

Figure 4. Classification report on CNNs model, which uses complete feature sets.

The combination of spectral and non-spectral data yielded superior results in classifying musical instruments using Convolutional Neural Networks (CNNs). Several things can be attributed to this. Spectral features capture the frequency-domain information essential for understanding a musical instrument's timbral qualities. The mentioned features include Mel Frequency Cepstral Coefficients (MFCCs), spectral centroid, spectral bandwidth, spectral contrast, spectral rolloff, and chroma features, among other characteristics. Conversely, non-spectral features offer additional information about audio signals' time-domain and statistical characteristics, including RMS energy, zero-crossing rate, kurtosis, and skewness. By incorporating these characteristics, a thorough depiction of the audio signals is achieved, covering both the spectral composition and the range of sound intensity.

Combining several features results in a more equitable performance across diverse instrument categories. The results indicate that the macro average precision, recall, and F1-score are more significant when using the combined feature set than when employing spectral or non-spectral features individually. This suggests the model consistently performs well classifying various instrument types, minimizing the chances of biased or distorted categorization.

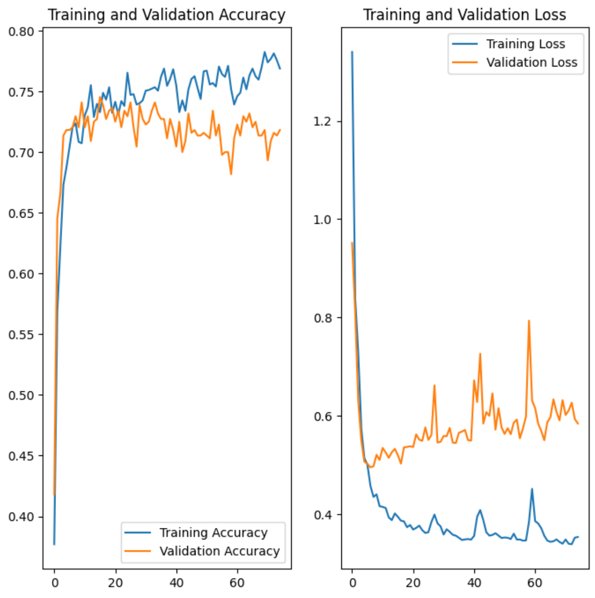
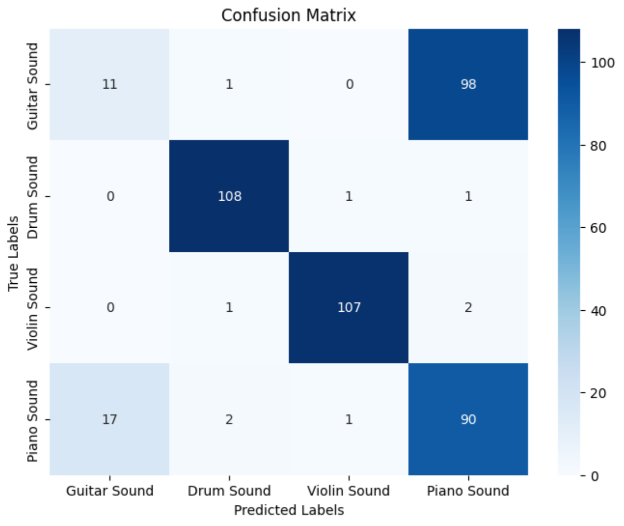


Figure 5. Training and validation performance of CNNs model, which uses complete feature sets.

The training and validation performance visualizations in Figure 5 show the model's learning process, which spans 75 epochs. The validation accuracy stabilized at approximately 72%, while the training accuracy progressively increased, reaching approximately 77%. The training loss consistently decreased, suggesting that learning was practical. However, some fluctuations in the validation loss indicate the possibility of overfitting in specific epochs.



	Guitar Sound	Drum Sound	Violin Sound	Piano Sound
Guitar Sound	11	1	0	98
Drum Sound	0	108	1	1
Violin Sound	0	1	107	2
Piano Sound	17	2	1	90

Figure 6. The confusion matrix of the CNN model uses complete feature sets.

The confusion matrix shown in Figure 6 revealed that the model correctly classified 11 instances of Guitar Sound but misclassified 98 instances as Piano Sound. It accurately identified 108 instances of Drum Sound, with minimal misclassifications. Violin Sound had 107 correct classifications with only a few errors. Piano Sound showed many correct classifications (90) but also some confusion with Guitar Sound and other classes.

Models trained exclusively with spectral features, including the Mel spectrogram, MFCCs, chroma, and other spectral parameters, also demonstrated satisfactory performance; however, they could not duplicate the levels attained by the comprehensive feature set. The most effective configuration in this category achieved a training accuracy of 79.8% and a validation accuracy of 69.3%. The details of validation performance can be seen in Figure 6. The training accuracy showed an increasing pattern, reaching almost 80%, whereas the validation accuracy displayed fluctuations before stabilizing at around 69.3%. The disparity between the model's accuracy on the training and validation data indicates the presence of overfitting. Overfitting occurs when the model performs well on the training data but needs to generalize better to unknown validation data. The training loss exhibited a continuous decline, whereas the validation loss increased at a certain point, providing further evidence of overfitting. This configuration consisted of 128 units with a learning rate 0.001 and 75 epochs. The spectral feature set still offered substantial information for classification despite marginally higher loss values than the all-features model.

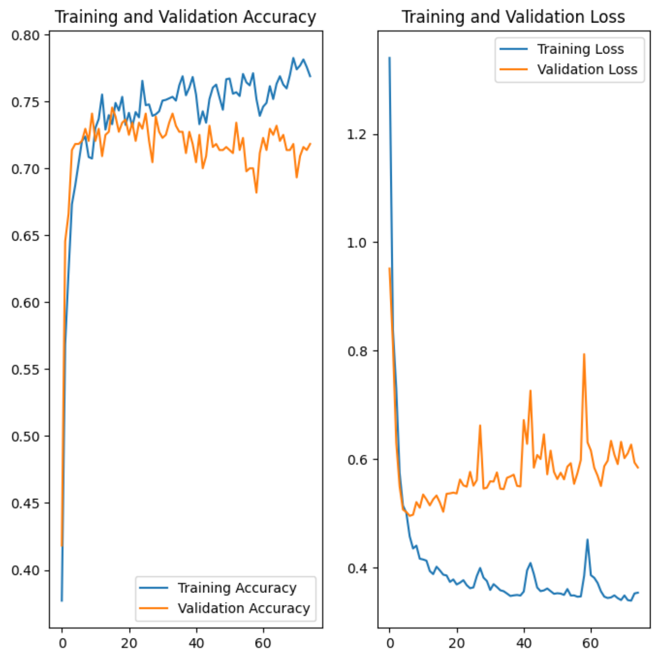
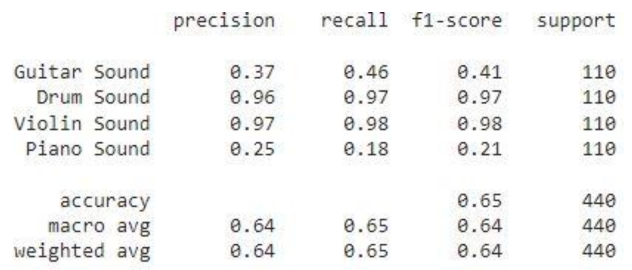


Figure 7. The training and validation performance of the CNNs model uses only spectral feature sets.

The macro average precision, recall, and F1-score were 63.9%, 65.0%, and 64.2%, as shown in Figure 8. The precision, recall, and F1 scores of each class—Guitar Sound, Drum Sound, Violin Sound, and Piano Sound—are displayed in the classification report. With a precision of 0.97, recall of 0.98, and an F1-score of 0.98, the Violin Sound class achieved the highest scores, suggesting that the model can accurately identify violin notes. The Drum Sound class also demonstrated exceptional performance, with scores ranging from 0.96 to 0.97. Nevertheless, the Piano Sound and Guitar Sound classes had substantially lower scores, with the Piano Sound class having a precision of 0.25, a recall of 0.18, and an F1 score of 0.21. This implies that the model encountered difficulty in differentiating piano notes from those of other instruments.

The confusion matrix shown in Figure 9 offers more information regarding the model's performance. The model accurately classified 51 instances of Guitar Sounds but misclassified 58 as Piano Sounds. The notable misclassification of Guitar and Piano Sounds underscores a potential opportunity for enhancement. The identification of the Drum and Violin Sounds was accurate in most cases, with only a few misclassifications. Nevertheless, the Piano Sounds were frequently misidentified as Guitar Sounds (86 occurrences), highlighting the model's challenge in differentiating between these categories.


```
              precision    recall  f1-score   support

Guitar Sound       0.37      0.46      0.41       110
  Drum Sound       0.96      0.97      0.97       110
Violin Sound       0.97      0.98      0.98       110
 Piano Sound       0.25      0.18      0.21       110

    accuracy                           0.65       440
   macro avg       0.64      0.65      0.64       440
weighted avg       0.64      0.65      0.64       440
```

Figure 8. A classification report on the CNN model uses only spectral feature sets.

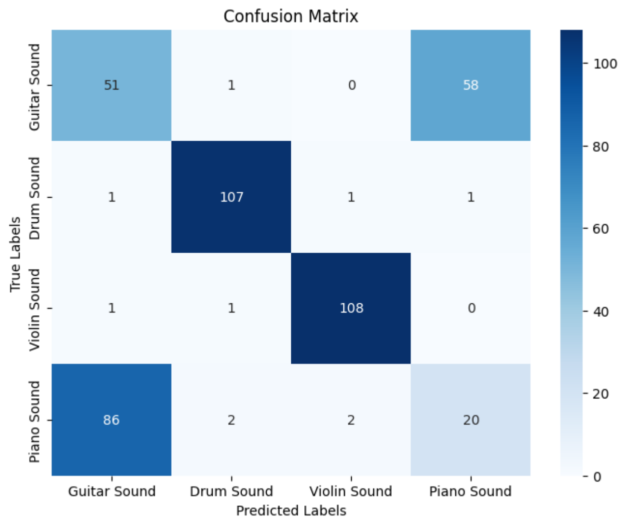


Figure 9. The confusion matrix of the CNN model only uses spectral feature sets.

Among the three feature sets, the CNN models that only employed non-spectral features, such as statistical measures of MFCCs, amplitude-based features, and various measures of the audio signal's dynamics and complexity, exhibited the lowest performance. The configuration that demonstrated the highest performance, which consisted of 128 units, a learning rate of 0.001, and 75 epochs, attained a validation accuracy of 68.4% and a training accuracy of 75.5%. The loss values for these models were higher, suggesting that the learning process was less efficient. The training and validation performance can be seen in Figure 9. The training accuracy steadily improves, reaching around 75.5%, while the validation accuracy peaks at approximately 68.4%. The validation accuracy fluctuates more than the training accuracy, indicating potential overfitting where the model performs well on training data but less consistently on unseen validation data. The loss curves highlight this further, with training loss decreasing continuously, whereas validation loss decreases initially but then shows variability, reflecting the challenges in generalizing the learning.

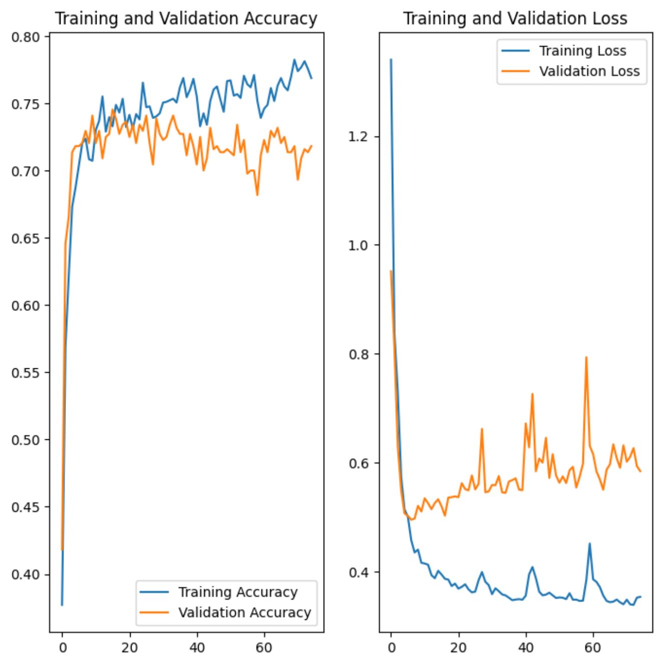
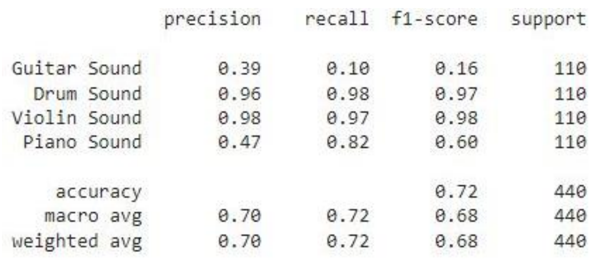


Figure 10. The training and validation performance of the CNNs model uses only nonspectral feature sets.

Precision, recall, and F1-score were 66.5%, 66.4%, and 66.2%, respectively, for the macro average, as shown in Figure 10. The classification report indicates that the model attained a 66% overall accuracy across the four categories: Guitar Sound, Drum Sound, Violin Sound, and Piano Sound. The precision, recall, and F1-score metrics for each class provide insights into the model's capacity to accurately classify different types of sounds. The model performs exceptionally accurately in classifying drum sounds, achieving a precision of 0.95, recall of 0.90, and F1-score of 0.93. These metrics indicate a high level of dependability and consistency in this particular category. On the other hand, the model has difficulties reliably detecting Guitar Sounds, as noted in a precision of 0.42, recall of 0.35, and F1-score of 0.38. The performance metrics of Violin Sounds and Piano Sounds are both at an intermediate level, with Violin Sounds exhibiting comparatively high scores and Piano Sounds displaying moderate scores.

The confusion matrix in Figure 12 shows a substantial misclassification rate for guitar sounds, frequently predicted as piano sounds. Conversely, the model's robustness in this category is illustrated by the fact that Drum Sounds are predominantly classified accurately. The classification of violin sounds is also highly accurate, although there are occasional misclassifications as drum sounds. Piano Sounds demonstrate a balanced distribution of correct and incorrect classifications, with errors dispersed across all other categories. Non-spectral features provide vital information regarding the distribution and dynamics of the audio signals; however, they do not capture the detailed frequency content that spectral features do, which is crucial for high-accuracy classification.


```
              precision    recall  f1-score   support

Guitar Sound       0.39      0.10      0.16       110
  Drum Sound       0.96      0.98      0.97       110
Violin Sound       0.98      0.97      0.98       110
 Piano Sound       0.47      0.82      0.60       110

    accuracy                           0.72       440
   macro avg       0.70      0.72      0.68       440
weighted avg       0.70      0.72      0.68       440
```

Figure 10. A classification report on the CNN model uses only spectral feature sets.

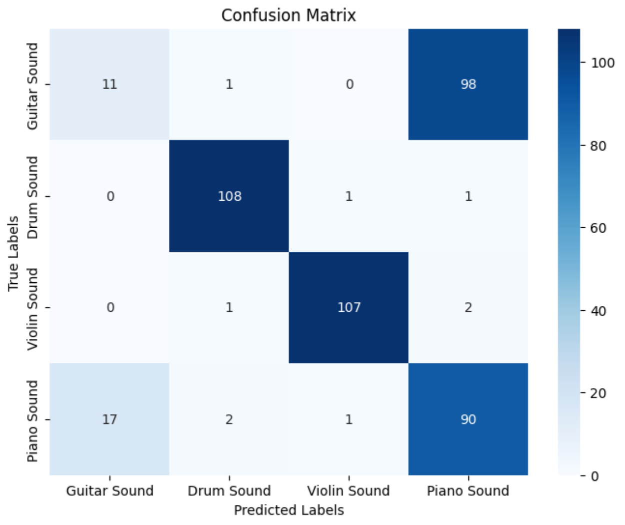


Figure 12. The confusion matrix of the CNN model only uses spectral feature sets.

Multiple configurations of the CNN model were evaluated by modifying factors such as the number of convolutional filters, learning rates, and epochs. Based on the performance of different configurations, an evaluation was conducted to determine the most effective arrangement for the classification task. The quantity of filters also increases the model's ability to acquire more intricate patterns in the data. The peak performance was attained using 512 filters, which offered ample capacity to capture detailed features in the audio signals without experiencing overfitting.

The learning rate controls the speed at which the model updates its weights during training. A learning rate of 0.01 was optimal, balancing convergence speed and stability. This learning rate allowed the model to learn effectively without causing large oscillations in the loss function.

The model achieved optimal results after being trained for 75 epochs. The time was adequate for the model to acquire the underlying patterns in the data without succumbing to overfitting. The training and validation performance curves show that the validation accuracy reached a plateau at approximately 71.8%, indicating that further epochs did not result in substantial enhancements and may have caused overfitting.

IV. Conclusion

In this study, we investigated the efficacy of Convolutional Neural Networks (CNNs) in classifying acoustic instruments by utilizing a variety of audio feature sets. Our objective was to determine the most effective CNN configurations for this classification task by methodically adjusting the number of convolutional filters, learning rates, and epochs.

Combining spectral and non-spectral features significantly improves the performance of CNNs for classifying musical instruments, as shown by the results of this research. Spectral features, which record the frequency-domain features of audio data, give us essential information about how instruments sound. On the other hand, non-spectral features add important information from the time domain and statistical traits, which helps the model understand how dynamics and amplitude change. This combination leads to better accuracy in classifying sounds and more even performance across all instrument types. This shows the importance of using an integrated feature selection method when making deep learning models that work well for complex audio tasks.

The study of different CNN configurations shows that carefully tuning model parameters is the key to getting the best results in classifying musical instruments. We got the best results using 512 convolutional filters, a learning rate 0.01, and 75 epochs. This model obtained a validation accuracy of 71.8% and a training accuracy of 76.9%. The performance of this configuration was balanced and robust, as evidenced by the macro average precision, recall, and F1-score of 70.3%, 71.8%, and 67.7%, separately. The number of chosen filters lets the model pick out fine details, and the learning rate selected makes convergence steady and reliable. The training period was long enough to keep the model's performance stable, which led to solid classification accuracy. These results show how important it is to fine-tune CNN settings to get the most out of the model's ability to tell the difference between things and generally do well in complex classification tasks.

References

[1] K. Racharla, V. Kumar, C. B. Jayant, A. Khairkar, and P. Harish, “Predominant musical instrument classification based on spectral features,” in *2020 7th International Conference on Signal Processing and Integrated Networks, SPIN 2020*, 2020. doi: 10.1109/SPIN48934.2020.9071125.

[2] S. R. Chaudhary, S. N. Kakarwal, and J. V. Bagade, “Feature selection and classification of indian musical string instruments using svm,” *Indian Journal of Computer Science and Engineering*, vol. 12, no. 4, 2021, doi: 10.21817/indjcse/2021/v12i4/211204142.

[3] P. K. Aurchana, “Musical Instruments Sound Classification using GMM,” *London Journal of Social Sciences*, 2021, doi: 10.31039/ljss.2021.1.37.

[4] C. Dewi, A. P. S. Chen, and H. J. Christanto, “Recognizing Similar Musical Instruments with YOLO Models,” *Big Data and Cognitive Computing*, vol. 7, no. 2, 2023, doi: 10.3390/bdcc7020094.

[5] S. Rajesh and N. J. Nalini, “Musical instrument emotion recognition using deep recurrent neural network,” in *Procedia Computer Science*, 2020. doi: 10.1016/j.procs.2020.03.178.

[6] Y. Su, “Instrument Classification Using Different Machine Learning and Deep Learning Methods,” *Highlights in Science, Engineering and Technology*, vol. 34, 2023, doi: 10.54097/hset.v34i.5435.

[7] S. K. Mahanta, N. J. Basisth, E. Halder, A. F. U. R. Khilji, and P. Pakray, “Exploiting cepstral coefficients and CNN for efficient musical instrument classification,” *Evolving Systems*, vol. 15, no. 3, 2024, doi: 10.1007/s12530-023-09540-x.

[8] C.-W. Weng, C.-Y. Lin, and J.-S. R. Jang, “Music Instrument Identification Using MFCC: Erhu as an Example,” *Chinese Music Dept, Tainan National …*, 2004.

[9] M. Blaszke and B. Kostek, “Musical Instrument Identification Using Deep Learning Approach,” *Sensors*, vol. 22, no. 8, 2022, doi: 10.3390/s22083033.

[10] SOUMENDRA PRASAD MOHANTY, “Musical Instrument’s Sound Dataset.” Accessed: Jun. 01, 2024. [Online]. Available: https://www.kaggle.com/datasets/soumendraprasad/musical-instruments-sound-dataset/

[11] D. S. Lau and R. Ajoodha, “Music Genre Classification: A Comparative Study Between Deep Learning and Traditional Machine Learning Approaches,” in *Lecture Notes in Networks and Systems*, 2022. doi: 10.1007/978-981-16-2102-4_22.

[12] J. L. Leevy, J. M. Johnson, J. Hancock, and T. M. Khoshgoftaar, “Threshold optimization and random undersampling for imbalanced credit card data,” *J Big Data*, vol. 10, no. 1, 2023, doi: 10.1186/s40537-023-00738-z.

[13] R. A. Nawasta, N. H. Cahyana, and H. Heriyanto, “Implementation of Mel-Frequency Cepstral Coefficient as Feature Extraction using K-Nearest Neighbor for Emotion Detection Based on Voice Intonation,” *Telematika*, vol. 20, no. 1, 2023, doi: 10.31315/telematika.v20i1.9518.

[14] P.-N. Tan, M. Steinbach, A. Karpatne, and V. Kumar, “Introduction to data mining Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar.,” *Introduction to data mining*, 2019.