Robust CNN-Based Musical Instrument Recognition with Enhanced Feature Learning.

Padmesh Sivalingam*, Aamith Kishore T J*, Sri Krishna P*, Yaswanth Reddy B*, Ragav S*, Lekshmi C. R.*
*Amrita School of Artificial Intelligence, Amrita Vishwa Vidyapeetham, Coimbatore, India
Emails: {cb.sc.u4aie24044, cb.sc.u4aie24001, cb.sc.u4aie24054, cb.sc.u4aie24061, cb.sc.u4aie24041}@cb.students.amrita.edu, cr_lekshmi@cb.amrita.edu

***Abstract*—Musical instrument recognition is a challenging task with applications in music information retrieval, audio processing, and automated transcription. This study presents a Convolutional Neural Network (CNN) model leveraging Mel spectrograms from the IRMAS dataset to classify 11 instrument categories. The model, incorporating convolutional layers, batch normalization, and dropout regularization, achieved a peak validation accuracy of 78.37% over 60 epochs. Comparative analysis with state-of-the-art methods highlights its competitive performance and computational efficiency. Robustness evaluations on varying input lengths and noise levels assess the model's generalization. Performance metrics, including accuracy trends, loss curves, and a confusion matrix, demonstrate strong classification for instruments like piano and violin while revealing challenges in distinguishing spectrally similar instruments. These findings underscore the effectiveness of CNNs for instrument classification and provide insights for enhancing deep learning-based audio recognition models.**

***Index Terms*—Musical Instrument Classification, Convolutional Neural Networks (CNN), Mel Spectrograms, Audio Signal Processing, Deep learning**

I. Introduction

The classification of musical instruments from audio signals is an essential task in music information retrieval (MIR) with applications in automated music transcription, content-based recommendation systems, and digital music analysis. Identifying instruments within an audio recording plays a crucial role in tasks such as music tagging, source separation, and performance analysis, benefiting both academic research and commercial applications.

Despite its significance, instrument classification presents several challenges. The spectral characteristics of different instruments often overlap, making it difficult to distinguish between similar-sounding classes. Variability in recording conditions, background noise, and differences in instrument timbre further complicate the classification process. Additionally, variations in playing techniques introduce intra-class diversity, making it challenging for traditional machine-learning models to achieve high accuracy.

Conventional methods rely on manually extracted audio features, such as spectral centroids and Mel-frequency cepstral coefficients (MFCCs). While these features help characterize audio signals, they often fail to capture intricate patterns necessary for robust classification. Deep learning techniques, particularly Convolutional Neural Networks (CNNs), have emerged as powerful tools for processing audio data. By learning feature representations directly from raw or transformed audio inputs such as spectrograms, CNNs have shown significant success in speech recognition, environmental sound classification, and music genre identification.

This study proposes a CNN-based approach for musical instrument classification using the IRMAS-TrainingData dataset, which contains audio samples from 11 distinct instruments, including guitar, piano, violin, and flute. Our method transforms raw audio signals into Mel spectrograms—a time-frequency representation that effectively captures spectral features—and uses them as input to a CNN model. The architecture incorporates multiple convolutional layers, batch normalization, and dropout techniques to enhance generalization and mitigate overfitting. Additionally, adaptive learning rate adjustments and early stopping mechanisms are implemented to optimize training efficiency.

To assess the effectiveness of our approach, we evaluate its performance using accuracy, loss metrics, and a confusion matrix, providing a comprehensive analysis of its strengths and limitations. Furthermore, we compare our results with state-of-the-art (SOTA) methods and conduct robustness testing by analyzing the model's performance using a confusion matrix and accuracy-loss curves. By leveraging deep learning for instrument classification, this research contributes to the advancement of MIR and audio-based machine learning. The findings highlight the potential of CNNs in musical instrument recognition and offer insights for further improving automated music analysis systems.

II. Realated Work

Early approaches to musical instrument recognition relied on handcrafted feature extraction combined with classical machine-learning techniques. Heittola et al. [6] employed Non-negative Matrix Factorization (NMF) alongside Mel-frequency cepstral coefficients (MFCCs) to analyze and classify instruments in audio recordings.

Further advancements in feature-based classification refined these early approaches. Kitahara et al. [7] introduced a fusion

*Corresponding author: Lekshmi C. R. (Email: cr_lekshmi@cb.amrita.edu)

model that utilized spectral, temporal, and modulation features with principal component analysis (PCA) to enhance classification accuracy. Fuhrmann et al. [3] extended this research by employing Support Vector Machines (SVMs) to classify instruments based on extracted audio features. Bosch et al. [1] incorporated source separation techniques as a preprocessing step, improving feature extraction and overall recognition performance.

With deep learning advancements, Convolutional Neural Networks (CNNs) significantly improved instrument recognition. Han et al. [5] introduced a Mel-spectrogram-CNN approach with aggregation over sliding windows, which Pons et al. [13] optimized for better timbral feature capture. Gururani et al. [15] applied deep neural networks (DNNs) with temporal max-pooling for instrument detection, while Yu et al. [16] explored multitask learning with auxiliary classification to refine category recognition. Gomez et al. [4] investigated source separation and transfer learning, demonstrating improved performance, particularly for smaller datasets. Soraghan et al. [12] employed the Hilbert-Huang Transform (HHT) with CNNs, while Kratimenos et al. [8] trained VGG-like CNN classifiers on augmented datasets for improved recognition.

Building upon these approaches, Lekshmi et al. [9], [11], [14] explored Mel-spectrogram and phase-based modgdgram representations with data augmentation using WaveGAN. They also experimented with different transformer architectures using an ensemble of Mel-spectrogram, tempogram, and modgdgram representations. Zhang and Bai [17] embedded augmentation-based CNN architectures for better generalization across datasets. Zhong et al. [18] investigated pretraining on isolated musical notes for polyphonic instrument classification using transfer learning. Dash et al. [2] integrated Discrete Wavelet Transform (DWT) with deep CNNs to enhance classification in complex musical arrangements. In contrast, Lekshmi and Rajan [10] explored compact convolutional transformers for multi-instrument recognition, setting new benchmarks in classification accuracy.

Despite advancements in instrument recognition, most studies have focused on multiple predominant instruments rather than solo classification. While CNN-based models have improved performance, challenges persist in generalization, dataset biases, and distinguishing instruments with similar timbres. To address these gaps, our work leverages a CNN with Mel-spectrograms for solo instrument classification using the IRMAS dataset. By incorporating regularization to mitigate overfitting and conducting a detailed performance analysis, we contribute to improving the accuracy and robustness of deep learning-based instrument recognition.

The rest of the paper is organized as follows. Section III represents the system description, Section IV represents the results and analysis. Finally, the paper is concluded in Section V.

III. System Description

The proposed system classifies musical instruments from the IRMAS-TrainingData dataset using a Convolutional Neural Network (CNN). The process consists of four key stages: data preprocessing, model design, training with evaluation, and performance comparison.

A. Dataset and Preprocessing

The IRMAS-TrainingData dataset contains 6,705 three-second audio clips in `.wav` format, each corresponding to one of 11 solo instruments: acoustic guitar (gac), clarinet (cla), electric guitar (gel), cello (cel), violin (vio), organ (org), saxophone (sax), trumpet (tru), flute (flu), piano (pia), and voice (voi). Audio files were processed using the Librosa library with a 22,050 Hz sampling rate, ensuring a balance between computational efficiency and audio fidelity.

TABLE I
SUMMARY OF THE IRMAS-TRAININGDATA DATASET

Instrument	Number of Samples
Acoustic Guitar (gac)	637
Clarinet (cla)	505
Electric Guitar (gel)	760
Cello (cel)	388
Violin (vio)	580
Organ (org)	682
Saxophone (sax)	615
Trumpet (tru)	577
Flute (flu)	595
Piano (pia)	721
Voice (voi)	778
Total	**6,705**

B. Mel Spectrogram Computation

To transform raw audio into a suitable format for the CNN, we computed Mel spectrograms, a time-frequency representation that aligns with human auditory perception. The Mel spectrograms were generated with 128 Mel bands, a hop length of 512 samples, and an FFT window size of 2,048, ensuring a balance between frequency resolution and temporal granularity [5]. The resulting power spectrograms were converted to a decibel scale using Librosa's `power_to_db` function to emphasize amplitude variations, which are critical for distinguishing instrument timbres [5].

To ensure uniformity across all spectrograms, we addressed shape inconsistencies by defining a target shape of 128×128 pixels. Spectrograms smaller than this size were padded with zeros, while larger ones were truncated using a custom `pad_or_truncate` function. This step was crucial for batch processing in the CNN, which requires fixed-size inputs. The spectrograms were then normalized using min-max scaling to map their values to the range [0, 1], reducing the impact of amplitude variations across samples. Finally, a channel dimension was added to each spectrogram, resulting in an input shape of (128, 128, 1), making the data compatible with the 2D convolutional layers of the CNN.

The Mel spectrogram is computed by first applying the Short-Time Fourier Transform (STFT) to the audio signal $x(t)$, followed by a Mel filterbank transformation. The STFT is defined as:

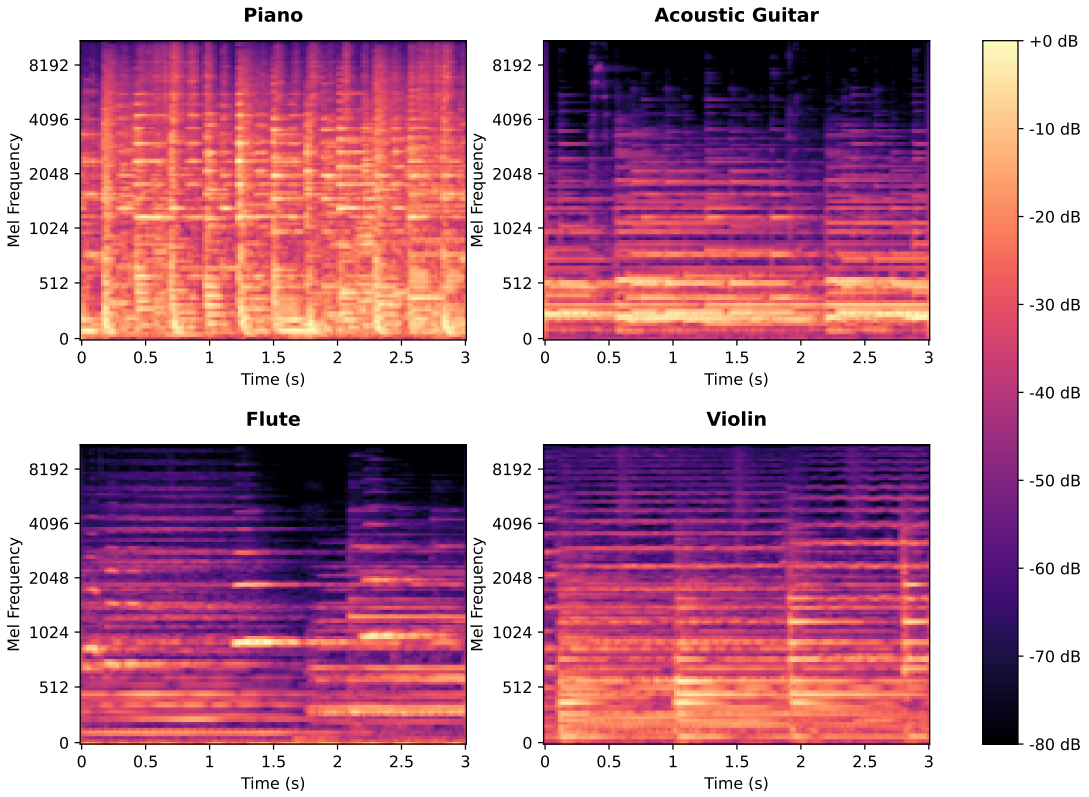

Fig. 1. Mel spectrograms of selected instruments: Piano, Acoustic Guitar, Flute, and Violin. Each spectrogram represents the time-frequency distribution of the corresponding instrument.

$$S(t,f) = \sum_{n=0}^{N-1} x(n)w(n-t)e^{-j2\pi fn/N}, \tag{1}$$

where $w(n)$ is the window function, N is the FFT window size (2,048), and f is the frequency bin.

The Hann window function, used to reduce spectral leakage, is given by:

$$w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right). \tag{2}$$

The power spectrogram $|S(t,f)|^2$ is then passed through a Mel filterbank with 128 bands to obtain the Mel spectrogram:

$$S_{\text{Mel}}(t,m) = \sum_{f} H(m,f) \cdot |S(t,f)|^2, \tag{3}$$

where $H(m,f)$ represents the Mel filter weights.

To convert from frequency (Hz) to the Mel scale, we use:

$$f_{\text{mel}} = 2595 \log_{10}\left(1 + \frac{f}{700}\right). \tag{4}$$

The inverse Mel scale transformation, used when necessary, is given by:

$$f = 700\left(10^{\frac{f_{\text{mel}}}{2595}} - 1\right). \tag{5}$$

The computed Mel spectrogram is converted to a decibel scale as:

$$S_{\text{dB}}(t,m) = 10 \log_{10}(S_{\text{Mel}}(t,m)). \tag{6}$$

Finally, min-max normalization is applied to ensure consistency across spectrograms:

$$S_{\text{norm}}(t,m) = \frac{S_{\text{dB}}(t,m) - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}}, \tag{7}$$

where S_{min} and S_{max} are the minimum and maximum values of the decibel-scaled spectrogram. Figure 1 shows sample Mel-spectrograms from the dataset.

These steps ensure that the extracted spectrogram features are standardized and well-suited for CNN-based classification.

C. Model Architecture

The CNN model was implemented using TensorFlow and Keras, designed to extract hierarchical features from Mel spectrograms for instrument classification. The architecture consists of three convolutional blocks, followed by global average pooling and dense layers. The first block includes a Conv2D layer with 64 filters (3×3 kernel), ReLU activation, and L2 regularization (coefficient of 0.0001) to prevent overfitting by penalizing large weights. This is followed by batch normalization to stabilize training by normalizing the activations, and a 2×2 max-pooling layer to reduce spatial

TABLE II
CNN MODEL ARCHITECTURE SUMMARY

Layer	Type	Output Shape	Params
Input	Input Layer	(128, 128, 1)	0
Conv2D	64 filters (3×3), ReLU	(126, 126, 64)	640
BatchNorm	Batch Norm	(126, 126, 64)	256
MaxPool	2×2 Pooling	(63, 63, 64)	0
Conv2D	128 filters (3×3), ReLU	(61, 61, 128)	73,856
BatchNorm	Batch Norm	(61, 61, 128)	512
MaxPool	2×2 Pooling	(30, 30, 128)	0
Conv2D	256 filters (3×3), ReLU	(28, 28, 256)	295,168
BatchNorm	Batch Norm	(28, 28, 256)	1,024
MaxPool	2×2 Pooling	(14, 14, 256)	0
GlobalAvgPool	Pooling Layer	(256)	0
Dense	Fully Connected (256), ReLU	(256)	65,792
Dropout	Dropout (50%)	(256)	0
Dense	Fully Connected (11), Softmax	(11)	2,827
Total Params			**440,075**

dimensions while retaining important features. The second and third blocks follow a similar structure but increase the number of filters to 128 and 256, respectively, allowing the model to learn more complex patterns as the depth increases. The progressive increase in filters reflects the need to capture both low-level features (e.g., edges in the spectrogram) and high-level features (e.g., instrument-specific spectral patterns).

After the convolutional blocks, a global average pooling layer reduces the feature maps to a 1D vector, eliminating the need for a flattening operation and reducing the number of parameters, which helps mitigate overfitting. The resulting vector is fed into a dense layer with 256 units and ReLU activation, also with L2 regularization, to learn non-linear combinations of the extracted features. A dropout layer with a 50% rate is applied to randomly disable half of the neurons during training, further preventing overfitting by promoting robustness in the learned features. The final layer is a dense layer with 11 units (one per instrument class) and a softmax activation, producing a probability distribution over the classes. The model was compiled with the Adam optimizer (initial learning rate of 0.0005) for adaptive gradient updates, categorical cross-entropy loss for multi-class classification, and accuracy as the evaluation metric.

The model's architecture and parameter distribution are summarized in Table II.

The softmax activation in the final layer computes the probability distribution over the 11 instrument classes as:

$$p(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{11} e^{z_j}}, \tag{8}$$

where z_i is the input to the softmax layer for class i, and $p(y_i)$ is the predicted probability of class i.

This architecture effectively balances complexity and generalization, ensuring robust feature extraction while mitigating overfitting through regularization and dropout.

D. Training and Evaluation

The dataset was split into training (80%) and test (20%) sets using stratified sampling to ensure that the class distribution was preserved, addressing the potential for class imbalance in the IRMAS dataset. The labels, originally integers representing the 11 instrument classes, were converted to one-hot encoded format using Keras' `to_categorical` function, enabling the model to perform multi-class classification. The model was trained for 60 epochs with a batch size of 32, a choice that balances computational efficiency with gradient stability. The Adam optimizer was used along with categorical cross-entropy as the loss function to enhance learning efficiency and handle the multi-class classification task effectively.

To improve training dynamics and prevent overfitting, two callbacks were employed. The `ReduceLROnPlateau` callback monitored the validation loss and reduced the learning rate by a factor of 0.5 (down to a minimum of 1e-6) if the loss did not improve for 3 consecutive epochs, allowing the model to make finer adjustments as it approached convergence. The `EarlyStopping` callback halted training if no improvement was observed for 10 epochs and restored the model weights from the epoch with the best validation performance to ensure optimal generalization. Table III summarizes the key hyperparameters used in the training process.

TABLE III
KEY HYPERPARAMETERS FOR TRAINING THE CNN MODEL

Hyperparameter	Value
Initial Learning Rate	0.0005
Batch Size	32
Number of Epochs	60
L2 Regularization Coefficient	0.0001
Dropout Rate	50%
Learning Rate Reduction Factor	0.5
Minimum Learning Rate	1e-6
Early Stopping Patience	10 epochs
ReduceLROnPlateau Patience	3 epochs
Optimizer	Adam
Loss Function	Categorical Cross-Entropy

During training, the model's performance was tracked using accuracy and loss metrics for both the training and validation

sets. For evaluation, a confusion matrix was generated to analyze the model's classification performance across the 11 instrument classes. The matrix was computed by predicting the classes of the test set, converting the softmax outputs to class labels using `argmax`, and comparing them to the true labels. Additionally, accuracy and loss curves were plotted over the 60 epochs to visualize the training dynamics and assess the model's generalization ability. These metrics provide a comprehensive view of the system's strengths and limitations, as discussed in the results section.

E. Performance Comparison

The performance of the proposed model is compared with state-of-the-art methods, including the Han model [5], Pons model [13], and a deep neural network (DNN) trained on handcrafted music texture features. Han [5] and Pons [13] employed deep CNN architectures for musical instrument recognition, incorporating a sliding window analysis and aggregation strategy. In their approaches, multiple sigmoid outputs were aggregated and thresholded to form the final prediction. Both models were evaluated using 1-second audio slices. In contrast, our method does not rely on sliding window analysis or aggregation strategies.

Additionally, we compare the proposed method with a model utilizing handcrafted music features such as spectral centroid, chroma short-time Fourier transform (chroma-STFT), MFCCs (13 coefficients), spectral roll-off, zero-crossing rate, and root mean square energy (RMSE), which were fed into a deep neural network (DNN). This DNN consists of seven layers, with the number of filters increasing from 8 to 512. The Adam optimizer was used with categorical cross-entropy loss for training.

IV. Results and Analysis

The proposed CNN model was trained and evaluated on the IRMAS dataset, achieving a peak validation accuracy of 78.37% at epoch 51 during the 60-epoch training process. The training logs indicate a steady improvement in both training and validation accuracy, with the learning rate being adaptively reduced from 0.0005 to 1e-6 over the course of training. Below, we present a detailed analysis of the model's performance using the confusion matrix, accuracy trends, and loss curves. Also, we compared our proposed CNN with state-of-the-art architectures.

A. Confusion Matrix Analysis

Figure 3 illustrates the confusion matrix for the test set, where the rows represent true instrument labels, and the columns represent predicted labels for the 11 instrument classes (gac: guitar acoustic, cla: clarinet, gel: guitar electric, cel: cello, vio: violin, org: organ, sax: saxophone, tru: trumpet, flu: flute, pia: piano, voi: voice).

The diagonal values indicate correct classifications, with high accuracy for piano (133), violin (142), and guitar acoustic (115). This suggests that the model performs well in recognizing these instruments. However, some misclassifications occur due to timbral similarities:

- Clarinet (cla) and Flute (flu): 10 clarinet samples were misclassified as flute, possibly due to overlapping spectral characteristics. - Guitar Electric (gel) and Guitar Acoustic (gac): 9 samples of guitar electric were classified as guitar acoustic, likely due to similar harmonic structures. - Saxophone (sax) and Trumpet (tru): There are a few cases of saxophone being confused with the trumpet, likely because both belong to the brass family and share similar frequency ranges.

These results indicate that while the CNN model captures distinguishing features for most instruments, fine-grained spectral distinctions remain challenging. Additional training data, augmentation techniques, or contrastive learning approaches could help mitigate these misclassifications.

B. Model Accuracy and Loss Trends

Figure 2 presents the training and validation accuracy and loss over 60 epochs. The training accuracy steadily increases, reaching 90.96% by the final epoch. The validation accuracy peaks at 78.37% at epoch 51, before slightly declining to 78.00%, indicating a potential overfitting issue despite regularization.

The training loss decreases consistently to 0.39, suggesting effective learning. The validation loss stabilizes at 0.73 after an initial spike, indicating that while the model generalizes reasonably well, further optimization could improve performance. The gap between training and validation accuracy suggests overfitting, which could be addressed by using data augmentation techniques such as pitch shifting, time stretching, and noise injection. Adding more dropout layers or stronger L2 regularization may also help reduce overfitting.

Additionally, the confusion matrix highlights misclassification issues for spectrally similar instruments. Potential solutions include incorporating self-supervised pretraining on larger unlabeled instrument datasets and using attention mechanisms to focus on discriminative spectral features. From a computational standpoint, the CNN model achieved reasonably fast convergence. Further tuning with a learning rate scheduler and adaptive batch sizes could enhance training efficiency and reduce unnecessary parameter updates.

C. Comparison with State-of-the-Art

To assess the performance of the proposed CNN model, we compare it with state-of-the-art (SOTA) methods, including the Han model [5], the Pons model [13], and a deep neural network (DNN) trained on handcrafted features such as MFCCs and music texture features (MTF). Table IV summarizes the comparison results, demonstrating that our approach achieves the highest accuracy while maintaining a significantly lower parameter count compared to the Han and Pons models.

The results indicate that the proposed CNN model achieves the highest accuracy while maintaining a significantly lower parameter count compared to other models. Despite having fewer parameters than the Han and Pons models, our approach surpasses them in classification performance, highlighting its efficiency. Also, we are not using sliding window analysis

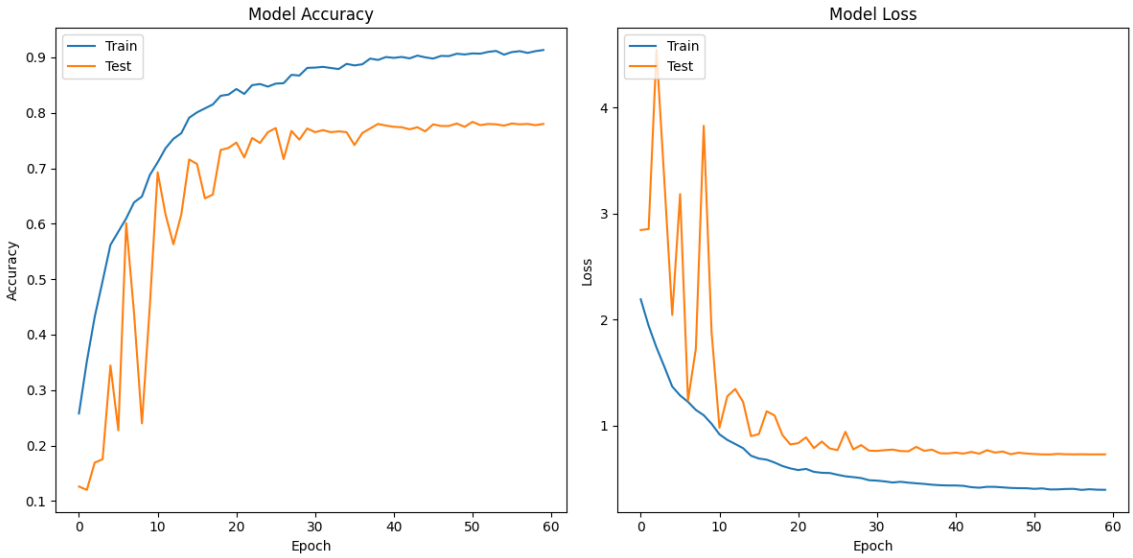

Fig. 2. Training and validation accuracy and loss over 60 epochs.

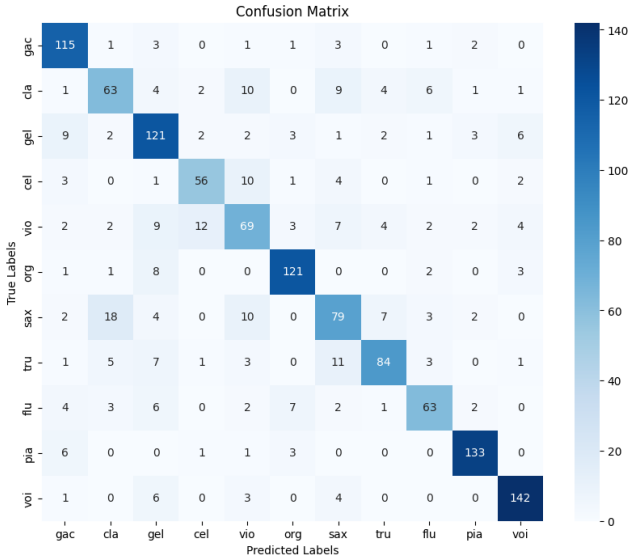

Fig. 3. Confusion Matrix for the test set, showing the classification performance across the 11 instrument classes.

TABLE IV
COMPARISON OF THE PROPOSED METHOD WITH STATE-OF-THE-ART MODELS

Method	Model Parameters	Test Accuracy (%)
Han et al. [5]	1442k	76.5
Pons et al. [13]	743k	75.2
MTF-DNN	NA	70.8
Proposed CNN Model	**440k**	**78.37**

and aggregation strategy for our method. The MTF + DNN approach, which relies on handcrafted features, achieves the lowest accuracy, suggesting that deep feature extraction from Mel spectrograms is more effective for instrument classification. These findings reinforce the advantage of using a well-optimized CNN architecture for musical instrument recognition.

V. CONCLUSION

This study proposed a CNN-based approach for musical instrument classification using Mel spectrograms, achieving a test accuracy of 78.47% on the IRMAS dataset. The model effectively learned discriminative features, accurately classifying predominant instruments despite challenges like overlapping spectral characteristics and background noise. Error analysis revealed confusion between similar timbral instruments, such as clarinet and flute, suggesting the need for additional refinements.

Future improvements could include incorporating advanced architectures like CRNNs or attention mechanisms to enhance feature extraction. Expanding the dataset, applying more data augmentation techniques, and exploring alternative spectrogram representations could further improve classification performance. These findings contribute to the field of music information retrieval, with potential applications in automatic transcription, music search, and genre classification.

REFERENCES

[1] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera. Acomparison of sound segregation techniques for predominant instrument recognition in musical audio signals. *in Proc. of 13th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2012.

[2] Sukanta Kumar Dash, SS Solanki, and Soubhik Chakraborty. Deep convolutional neural networks for predominant instrument recognition in polyphonic music using discrete wavelet transform. *Circuits, Systems, and Signal Processing*, 43(7):4239–4271, 2024.

[3] F. Fuhrmann and P. Herrera. Polyphonic instrument recognition for exploring semantic similarities in music. *in Proc. of 13th Int. Conf. on Digital Audio Effects DAFx10, Graz, Austria*, 14(1):1–8, 2010.

[4] Juan S Gómez, Jakob Abeßer, and Estefanía Cano. Jazz solo instrument classification with convolutional neural networks, source separation, and transfer learning. In *ISMIR*, pages 577–584, 2018.

[5] Y. Han, J. Kim, and K. Lee. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(1):208–221, 2017.

[6] T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. *in Proc. of International Society of Music Information Retrieval Conference*, pages 327–332, 2009.

[7] T Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Instrument identification in polyphonic music: Feature weighting to minimize influence of sound overlaps. *EURASIP Journal of Applied Signal Processing*, pages 155–175, 2007.

[8] A. Kratimenos, K Avramidis, C. Garoufis, Athanasia Zlatintsi, and Petros M. Augmentation methods on monophonic audio for instrument classification in polyphonic music. *in Proc.of 28th European Signal Processing Conference (EUSIPCO)*, pages 156–160, 2021.

[9] CR Lekshmi and Rajeev Rajan. Predominant instrument recognition in polyphonic music using convolutional recurrent neural networks. In *International Symposium on Computer Music Multidisciplinary Research*, pages 214–227. Springer, 2021.

[10] CR Lekshmi and Rajeev Rajan. Compact convolutional transformers for multiple predominant instrument recognition in polyphonic music. In *2024 9th International Conference on Communication and Electronics Systems (ICCES)*, pages 01–06. IEEE, 2024.

[11] CR Lekshmi and Rajan Rajeev. Multiple predominant instruments recognition in polyphonic music using spectro/modgd-gram fusion. *Circuits, Systems, and Signal Processing*, 42(6):3464–3484, 2023.

[12] X. Li, K. Wang, J. Soraghan, and J. Ren. Fusion of hilbert-huang transform and deep convolutional neural network for predominant musical instruments recognition. *in Proc. of 9th Int. Conf. on Artificial Intelligence in Music, Sound, Art and Design*, 2020.

[13] J. Pons, O. Slizovskaia, R. Gong, Emilia Gómez, and X. Serra. Timbre analysis of music audio signals with convolutional neural networks. *in Proc. of 2017 25th European Signal Processing Conference (EUSIPCO)*, pages 2744–2748, 2017.

[14] Lekshmi Chandrika Reghunath and Rajeev Rajan. Transformer-based ensemble method for multiple predominant instruments recognition in polyphonic music. *EURASIP Journal on Audio, Speech, and Music Processing*, 2022(1):11, 2022.

[15] G. Siddharth, C. Summers, and A. Lerch. Instrument activity detection in polyphonic music using deep neural networks. *in Proc. of Int. Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[16] Dongyan Yu, Huiping Duan, Jun Fang, and Bing Zeng. Predominant instrument recognition based on deep neural network with auxiliary classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:852–861, 2020.

[17] Jian Zhang and Na Bai. Augmentation embedded deep convolutional neural network for predominant instrument recognition. *Applied Sciences*, 13(18):10189, 2023.

[18] Lifan Zhong, Erica Cooper, Junichi Yamagishi, and Nobuaki Minematsu. Exploring isolated musical notes as pre-training data for predominant instrument recognition in polyphonic music. In *2023 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 2312–2319. IEEE, 2023.