Analyzing and Classifying Digital Time Signals Using Convolutional Neural Networks for Source Identification

Bidhan Paul
bidhan.paul@stud.fra-uas.de

Fahim Talukdar
fahim.talukdar@stud.fra-uas.de

***Abstract*— This paper presents a method for classifying labeled digital time signals emitted from two distinct sources using a Convolutional Neural Network (CNN). The challenge of accurately distinguishing between the two signal sources is heightened by the presence of noise within the dataset. To address this, we apply Daubechies wavelet-based denoising, specifically the db4 wavelet, to filter out noise and enhance the signal quality. The filtered signals are then used as input for a CNN designed to identify the source of each signal. In addition, to evaluate the robustness of the CNN, we perform signal classification using both denoised and raw, unfiltered signals.**

The CNN is trained on both pre-processed (filtered) and unprocessed (raw) signals to assess the impact of signal filtering on classification performance. Results demonstrate that denoising significantly improves classification accuracy, with the wavelet-filtered signals providing clearer input for feature extraction. However, the CNN also exhibits robust performance when classifying raw signals, indicating its ability to capture essential features even in the presence of noise.

The trained CNN model is stored for future use, enabling real-time classification of digital time signals in various applications. This research highlights the importance of combining signal processing techniques with deep learning models for enhanced performance and underscores the potential of CNNs in classifying noisy, real-world signals.

Keywords: Convolutional Neural Networks (CNN), Signal Classification, Daubechies Wavelet (db4), Signal Denoising, Source Identification.

I. Introduction

Signal classification is a fundamental task in modern engineering, with widespread applications across various domains such as telecommunications, radar systems, biomedical signal analysis, and industrial monitoring. Digital time signals, especially those emitted in noisy environments, often carry critical information that can be challenging to decode due to the interference introduced by noise. The ability to accurately classify such signals not only ensures the integrity of information transmission but also enhances decision-making processes in numerous real-world applications. In the context of this work, we focus on classifying digital time signals originating from two distinct sources, using deep learning approaches to differentiate between the signal sources despite the presence of noise.

Machine learning, and more specifically, Convolutional Neural Networks (CNNs), has proven to be highly effective for tasks involving pattern recognition and classification. CNNs have demonstrated superior performance in image classification and speech recognition, and their use has expanded to time-series data such as signals. The strength of CNNs lies in their ability to automatically extract hierarchical features from raw data, making them ideal for handling complex and noisy signals. In this study, we aim to leverage the capabilities of CNNs to address the problem of signal source classification, where the challenge lies in distinguishing signals affected by noise from two different origins.

A key component of our approach involves signal preprocessing, where we utilize wavelet-based denoising techniques. In signal processing, wavelet transforms are extensively used due to their multi-resolution analysis capabilities. By decomposing a signal into different frequency components, wavelets allow for both time and frequency localization, making them highly effective for denoising tasks. The **Daubechies wavelet** family, particularly **db4**, is known for its smoothness and compact support, making it a popular choice for signal denoising. The db4 wavelet is well-suited for signals that require both precision and smooth reconstruction after denoising. In our study, the db4 wavelet is employed to remove noise from the raw digital signals, preserving their underlying structure for more accurate classification.

The process of wavelet denoising involves several steps, including the decomposition of the signal into multiple levels, thresholding the wavelet coefficients to eliminate noise, and reconstructing the denoised signal. By applying this technique, we aim to improve the quality of the input data fed into the CNN. The impact of denoising is crucial because the presence of noise can obscure the distinctive features of the signal, making classification more difficult. However, signal preprocessing itself can introduce a new layer of complexity. Therefore, in addition to denoising the signals, we also explore the performance of the CNN when presented with raw, unfiltered signals. This comparison helps us evaluate how signal filtering affects the model's ability to learn and classify.

In signal classification tasks, the balance between preprocessing and model complexity plays a pivotal role. While signal filtering can aid the classifier by providing cleaner data, it also raises questions about the model's generalization capabilities when working with raw signals. CNNs, due to their inherent ability to learn features across different levels, may still be capable of capturing the essential characteristics of unprocessed signals. Thus, by training the model on both filtered and unfiltered signals, we aim to assess the robustness of the CNN under different conditions, providing insights into its adaptability and accuracy.

In the context of this research, the dataset comprises labeled digital time signals that stem from two distinct sources, and the objective is to develop a CNN model that can classify the signal's origin. The CNN architecture is designed to learn the underlying patterns that differentiate signals from source #1 and source #2, regardless of the noise level. To further enhance the applicability of our approach, the trained CNN model is stored for future use, allowing it to be deployed in real-time classification tasks or as a foundation for further research involving more complex signal datasets.

The novelty of our research lies in the dual approach of classifying signals with and without preprocessing. By using the db4 wavelet for filtering and comparing the results with classifications based on raw signals, we contribute to the understanding of how signal processing techniques influence the performance of deep learning models. Additionally, the use of CNNs in this context highlights their potential for handling noisy and complex signal classification tasks, pushing the boundaries of what can be achieved with minimal preprocessing.

In summary, this paper presents a comprehensive investigation into the classification of digital time signals using CNNs, with a focus on the effects of signal denoising through wavelet transforms. By comparing the performance of the model on both filtered and unfiltered signals, we provide valuable insights into the role of signal preprocessing in improving classification accuracy. This research has potential applications in fields where real-time signal analysis is critical, and accurate source identification is essential for effective decision-making.

II. Literature Review

A study proposes a decision fusion model for identifying unknown radar waveforms with limited samples and low signal-to-noise ratio using transfer learning and deep features from convolutional neural networks (CNN). The CNN is initially trained on known radar signals, and transfer learning is applied to extract features from time-frequency images (TFI). The decision-making process combines random forest classifiers applied to original TFI and short-time autocorrelation feature images (SAFI), using a linear weighting method to improve recognition. The approach achieves a recognition rate of over 80.31% for unknown signals and 99.15% accuracy for known radar waveforms. [1]

This study addresses dynamic spectrum access (DSA) by proposing a deep learning-based signal classification approach for detecting and classifying interference sources, such as in-network users, out-network users, and jammers, in a wireless network. The solution handles various challenges, including dynamic signal types, unknown signals, spoofed signals, and interference from concurrent transmissions. For evolving signal types, the model uses continual learning with a CNN and Elastic Weight Consolidation (EWC) loss. Unknown signals are detected through outlier detection using Minimum Covariance Determinant (MCD) and k-means clustering. Spoofed signals are identified by capturing phase shifts from radio hardware effects, and Independent Component Analysis (ICA) is used for blind source separation to deal with interference. This classification informs a distributed scheduling protocol, improving in-network user throughput and out-network user success rates compared to traditional TDMA-based schemes. [2]

This paper introduces an IA-optimal CNN to improve signal classification accuracy and stability by addressing limitations in the existing A-optimal-based CNN. The proposed method uses the trace of the covariance matrix of the fully connected layer's weights as the optimization objective function without simplifications, enhancing classification precision. To handle challenges related to calculating the partial derivative of the inverse matrix with respect to network parameters, a novel dual function is applied, transforming the problem into a binary function optimization. Parameters are updated using an alternate iterative optimization method, with accurate weight updates derived in detail. Testing across five signal datasets demonstrates the IA-optimal CNN's universality and improved performance compared to the A-optimal-based approach. The study also theoretically proves that while the A-optimal-based CNN reduces the trace of the covariance matrix toward a convergence value, it cannot reach the A-optimal state. [3]

This study presents a supervised learning method combining a convolutional neural network (CNN) with a sliding-window threshold to classify data with completely unknown categories. The sliding-window threshold locally smooths the continuous CNN predictions, and data labels are updated when a change point is detected based on simple rules. The method was tested using partial discharge (PD) data from accelerated deterioration tests on nine underground cable straight joints. Two initial change points, set at 30% and 70%, converged closely. Additionally, an unsupervised learning method was used to determine change points similar to those identified by the proposed method, confirming its validity through multiple verification approaches. [4]

This study focuses on improving audio event classification, which involves detecting and classifying non-verbal sounds like dog barks and horn honks. While CNN-based models using spectrograms (2D images of audio signals) have shown superior performance, they struggle when test data is recorded on unknown devices due to differences in frequency emphasis and spectrogram shapes. To address this, the study applies a CNN using a log mel-spectrogram separation technique to enhance classification performance, particularly for unknown devices. Testing across 16 audio signal types, the model, trained on known devices, improved accuracy on test data from unknown devices. Results showed a relative improvement of up to 37.33%, increasing accuracy from 63.63% to 73.33% on Google Pixel, and from 47.42% to 65.12% on LG V50 compared to the baseline. [5]

This paper addresses the growing need for efficient, wavelet-based real-time signal processing in portable medical devices, emphasizing reduced hardware size, cost, and power consumption. It proposes an improved Reconfigurable Multiplier Block (ReMB) architecture for an 8-tap Daubechies wavelet filter, implemented in a tree-structured filter bank targeting recent Field-Programmable Gate Array (FPGA) technologies. The ReMB replaces power-hungry multipliers and coefficient memories used in time-multiplexed finite impulse response filters. Implemented on a Kintex-7 FPGA, the architecture demonstrates a 30% reduction in hardware utilization and a 44% improvement in power consumption compared to general-purpose multiplier-based architectures. This makes it suitable for low-cost, low-power embedded platforms in portable medical devices. [6]

This paper presents an ECG feature extraction algorithm based on the Daubechies Wavelet Transform (DB4) for analyzing the electrical activity of the heart. The DB4 wavelet is chosen due to its scaling function's similarity to the ECG signal shape. The core of the algorithm is R peak detection, with other primary peaks extracted relative to the R peaks using windows based on normal time intervals. The algorithm was tested on the MIT-BIH Arrhythmia Database, demonstrating successful detection and extraction of key ECG features with a deviation error of less than 10%. This method aids in diagnosing heart conditions like Tachycardia, Bradycardia, and Heart Rate Variation. [7]

This article proposes a novel method for automatic ECG identification and classification using a dense heart rhythm network that combines a 24-layer Deep Convolutional Neural Network (DCNN) and Bidirectional Long Short-Term Memory (BiLSTM). This approach captures both hierarchical and time-sensitive features of ECG data. Three different convolution kernel sizes (32, 64, and 128) are utilized to extract detailed features, and noise is reduced through a combination of wavelet transform and median filtering. Additionally, a new loss function is introduced to stabilize loss fluctuation during training, employing a tan function-based convergence mapping (0–1 range) to improve model optimization. [8]

This study addresses the challenges in motor imagery (MI) electroencephalogram (EEG) signal classification, a key area in brain-computer interface (BCI) research. Deep learning has shown promise in automatically extracting and classifying raw MI EEG signal features, but it faces two main challenges: the need for large amounts of labeled data, which is often unavailable, and the time and computational expense of training models from scratch. To overcome these issues, the paper proposes a deep transfer learning framework based on the VGG-16 convolutional neural network (CNN). The framework transfers parameters from a VGG-16 model pre-trained on ImageNet to a target CNN model for MI EEG classification, freezing the front-layer parameters and fine-tuning the later layers using time-frequency spectrum images of EEG signals. Tested on the BCI competition IV dataset 2b, the proposed framework improves classification accuracy and efficiency compared to traditional methods like support vector machines (SVM), artificial neural networks (ANN), and standard CNNs. [9]

This paper presents a new approach for classifying underwater acoustic signals, such as those from fish, using a convolutional neural network (CNN) combined with discrete wavelet transform (DWT), referred to as CNN_DWT. The method reduces the dimensionality of signal processing, lowers computational costs, and improves target detection and classification performance. The proposed CNN processes scalogram images of underwater acoustic signals converted to the wavelet domain for greater accuracy. Experimental comparisons with classical methods, including spatial domain CNN and back-propagation neural networks (BPNNs), demonstrate that CNN_DWT significantly enhances classification accuracy, noise robustness, and convergence, while avoiding overfitting. The results show that CNN_DWT outperforms classical CNNs and BPNNs in both classification performance and generalization ability for underwater acoustic signals. [10]

This study aims to assist hearing-impaired individuals by developing a system that detects emergency sounds and communicates them effectively. Using a Multi-Channel Convolutional Neural Network (CNN), the model differentiates between emergency and non-emergency audio signals. Various data augmentation techniques, including Mixup, were employed to enhance model performance. The system achieved cross-validation accuracy of 88.28% and testing accuracy of 88.09%. To integrate this model into daily life, an Android application was developed that alerts users by vibrating the phone during emergency sounds. The app can also connect to wearable devices like smartwatches, ensuring that hearing-impaired individuals are notified in real-time during emergencies. [11]

III. METHODOLOGY

A. *Proposed Methodology:*

This study's technique employs an organized workflow Figure 1 created to overcome the difficulties associated with employing convolutional neural networks (CNNs) for time-series classification. The method starts with the data collecting and visualization stage, which involves loading, combining, and examining datasets from two different sources to evaluate their distribution and quality. To make sure the data is ready for model training as best as possible, pretreatment activities are then carried out. This entails managing missing data, standardizing features to ensure consistency between samples, and using wavelet denoising algorithms to minimize noise. Subsequently, the preprocessed data is reorganized into a manner appropriate for a 1D CNN model.

The training and testing phase is then put into action. Convolutional layers are used in the model architecture to extract pertinent features from the time-series data. Fully connected and pooling layers are then used for classification. Using a training-validation split on the preprocessed data, the model is trained with callbacks such as early stopping and learning rate reduction to guarantee optimal convergence and prevent overfitting. The model evaluation step starts once it has been trained. Important measures like accuracy, precision, recall, F1 score, and ROC are used to evaluate the model's performance. A confusion matrix is produced in order to shed light on the classification outcomes. The identification stage also concentrates on recognizing noteworthy patterns in the dataset and appreciating the significance of features. The categorization of unseen data, where the model predicts the labels of fresh samples, completes the classification task, is the last step in the workflow.

From data preprocessing to model evaluation, this organized technique guarantees a thorough and methodical approach, guaranteeing the robustness and accuracy of the CNN-based classification model's predictions.

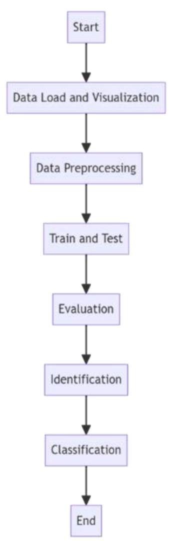

Figure 1: Workflow Diagram

B. Experimental Dataset:

For this study, time-series signal data from two different sources were combined to create the dataset. There are 5052-time steps in a signal, which corresponds to a series of data points recorded over a predetermined period of time. A binary classification system is used to organize the data, designating each signal as either coming from "Source 1" or "Source 2" in Figure 2. The main objective is to categorize these signals according to the features and patterns found in the time-series data. In order to make the signal values in the dataset appropriate for input into a Convolutional Neural Network (CNN), the dataset has been preprocessed by scaling them. The dataset is divided into training, validation, and testing subsets for the purpose of assessing model performance. The model is trained to identify the fundamental differences between the two sources based on signal properties. The model avoids bias towards any particular class during training and learns significant patterns thanks to the two classes' balanced distribution.

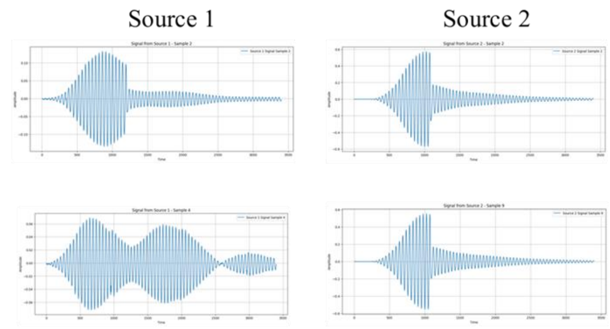

Figure 2: Visualization of Dataset of both sources.

C. Data Preprocessing:

The preprocessing of the data included multiple stages to ensure the quality and usability of the dataset for model training. To keep only the relevant features, we first merged two datasets and filtered out any unnecessary columns. In order to improve the clarity of the signals for better analysis, we next used wavelet denoising to remove noise in the signal data. Then, in order to preserve data integrity, we replaced zeros for any missing values. The model was then able to learn more efficiently by normalizing the denoised signals using StandardScaler to standardize the feature values. Ultimately, we transformed the information to match the Conv1D model's input specifications, preparing it for both training and assessment.

D. Proposed CNN Architecture:

Our goal with our suggested CNN architecture in Figure 3 is to efficiently classify one-dimensional signal data. Feature extraction and classification make up the two primary sections of the model's structure. Three convolutional layers are utilized in the feature extraction process, with each one replaced by a max-pooling layer to minimize dimensionality while maintaining crucial information. (samples, sequence length, 1) is the structure of the input shape, where "samples" denotes the number of signal samples and "sequence length" denotes the number of time steps per sample.

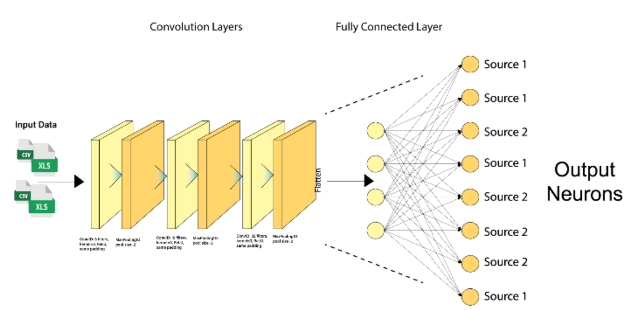

Figure 3: Proposed CNN Model Architecture

The input layer is the first convolutional layer, which maintains the input size by using'same' padding and 8 filters with a kernel size of 3. In order to add non-linearity and enable the model to learn intricate patterns, ReLU activation is used. A max-pooling layer with a pool size of two comes next, aiding in the downsampling of the feature maps to concentrate on the most significant features and lower dimensionality.

The second convolutional layer employs 16 filters, again with a kernel size of 3 and 'same' padding, followed by another max-pooling layer to further reduce the size of the feature maps while maintaining significant information. The third convolutional layer expands the feature extraction process with 32 filters, maintaining 'same' padding and max pooling to ensure that the network can capture more complex features from the signal data.

A fully connected layer with 64 hidden nodes is used following the flattening of the output from the convolutional layers. Every node in this layer is linked to every activation from the layer before it. To prevent overfitting, a dropout layer is added after this. During training, it randomly sets a portion of the input units to 0. With a sigmoid activation function, the final output layer—which is intended for binary classification—produces probabilities ranging from 0 to 1, signifying the likelihood of each class.

To provide effective training and high classification accuracy, this architecture achieves a compromise between depth and complexity. To achieve optimal performance without overshooting during optimization, the training process can be stabilized by using a small learning rate (1e-5). This enables incremental changes to the model weights. In Table the architecture description is given.

Layer (Type)	Output Shape	Param #	Description
Input Layer	(None, 5052, 1)	0	Input layer for sequences of length 5052 with 1 feature/channel
Conv1D	(None, 5052, 8)	32	8 filters, kernel size=3, ReLU activation, same padding
MaxPooling1D	(None, 2526, 8)	0	MaxPooling1D with pool size=2
Conv1D	(None, 2526, 16)	400	16 filters, kernel size=3, ReLU activation, same padding
MaxPooling1D	(None, 1263, 16)	0	MaxPooling1D with pool size=2
Conv1D	(None, 1263, 32)	1,568	32 filters, kernel size=3, ReLU activation, same padding
MaxPooling1D	(None, 631, 32)	0	MaxPooling1D with pool size=2
Flatten	(None, 20192)	0	Flatten layer to convert multi-dimensional data to 1D
Dense	(None, 64)	1,291,200	Fully connected layer with 64 units and ReLU activation
Dropout	(None, 64)	0	Dropout layer with rate=0.5 to prevent overfitting
Dense	(None, 1)	65	Output layer with 1 unit and Sigmoid activation for binary classification

Table 1: CNN Model Architecture Description

D. *Training and Performance Evaluation of the Proposed CNN Model:*

Our proposed Convolutional neural networks (CNNs) were trained by fitting them to training data and periodically evaluating their performance on a validation set. To provide the model enough iterations for learning, a maximum of 35 epochs were used for training. For consistent gradient updates and effective computing, a batch size of 64 was used. In order to guarantee prediction resilience, the model's performance was verified at the end of each epoch to track its generalization capacity. One of the most important techniques was early stopping, which prevented overfitting by terminating training if no improvement in validation loss was shown for five consecutive epochs.

Furthermore, a learning rate scheduler with a lower constraint of 1×10^{-3} was used, which cut the learning rate in half if the validation loss did not improve after three epochs. As a result, the model was able to better align and converge. When taken as a whole, these replies improved the model's efficiency and prevented overfitting or protracted training. From an initial value of 41.97% to 97.00% at the last epoch, the training accuracy increased steadily throughout the course of the epochs. Similar trends were seen in validation accuracy, which stabilized at 96.67% after the second epoch. The model showed good generalization to unknown data, with a final training accuracy of 98.50% and a validation accuracy of 96.67%.

IV. Result Analysis

The analysis of training and validation performance over the training epochs in Figure 4 is the first step in evaluating the suggested model. The model's learning from the training set and generalization to the validation set are illustrated by the loss and accuracy curves. Effective learning and the avoidance of overfitting were demonstrated by the training and validation accuracy increasing gradually as the loss reduced. By preventing over-training, the early halting method enhanced the model's generalization skills.

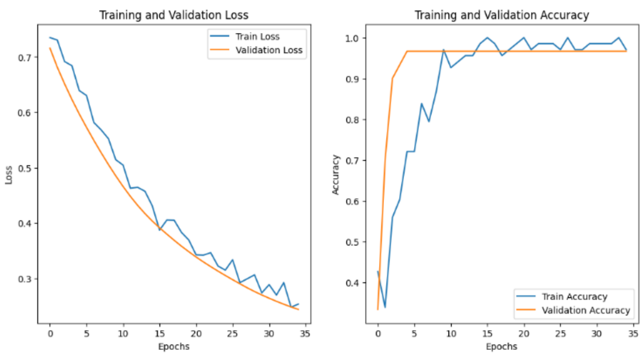

Figure 4: Proposed CNN Model Architecture

Based on the validation data, a confusion matrix in Figure 5 was constructed to evaluate the classification performance. The matrix offers a thorough analysis of how successfully the model distinguishes between the two classes by comparing its predictions to the actual labels. Most samples had minor misclassifications and were correctly identified, demonstrating the model's resilience to handle unknown data.

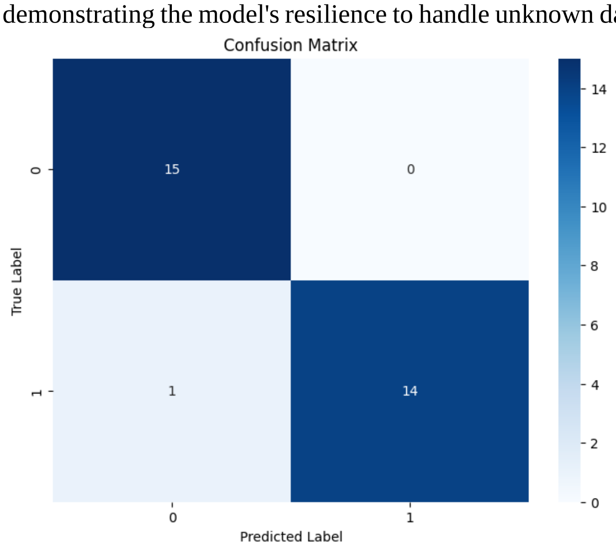

Figure 5: Confusion Matrix

Random samples from unobserved data were used to test the model, and the true labels and predictions were compared. The unseen in Figure 6 data was used to display the results, which demonstrated a high degree of accuracy in these predictions. This provided additional support for the model's capacity to generalize.

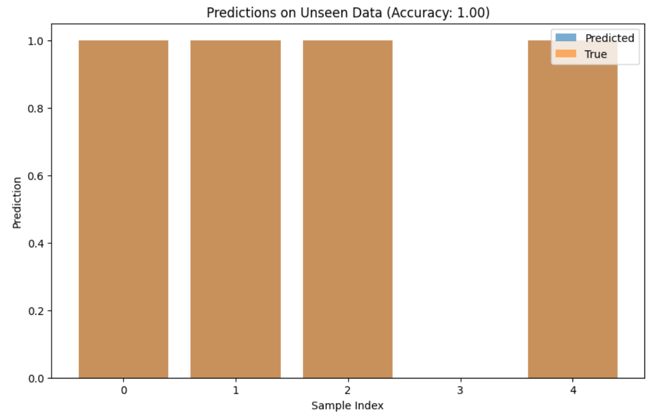

Figure 6: Unseen Data Prediction

Finally, the Receiver Operating Characteristic (ROC) curve is a crucial tool for evaluating the performance of the proposed model, especially in binary classification tasks. It plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings, offering insight into the model's ability to distinguish between classes. A well-performing model exhibits a ROC curve that curves towards the top left corner, indicating high sensitivity and specificity.

For this model, the ROC curve in Figure 7 showed a strong performance, with an Area Under the Curve (AUC) score of 0.99. The AUC value close to 1 signifies that the model is highly effective in differentiating between the two classes, with minimal overlap between positive and negative predictions. This high AUC demonstrates the model’s capability to balance true positives and false positives across different classification thresholds, reinforcing its reliability and precision in predicting outcomes.

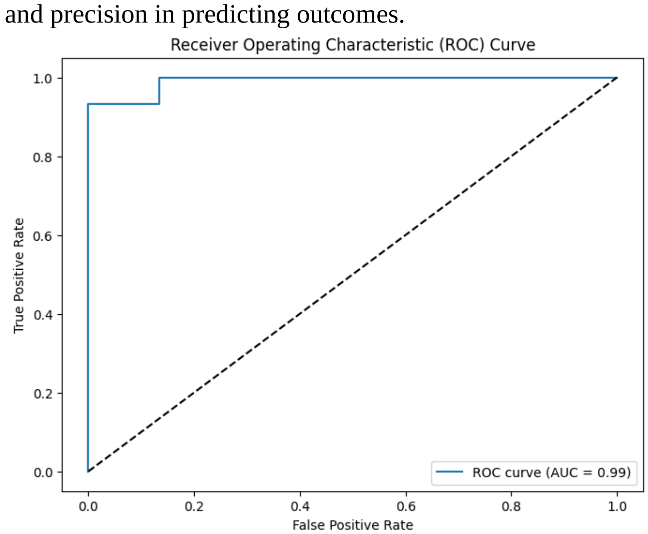

Figure 7: ROC Curve

V. INTEGRATION OF THE PRESERVED MODEL THROUGH GUI

The preserved model is made accessible through a user-friendly graphical user interface (GUI) built with the Tkinter library. This interface allows users to easily upload their data files, which are then processed for signal classification. Once the data is loaded, users can visualize the signals on the GUI, providing an intuitive way to analyze the information.

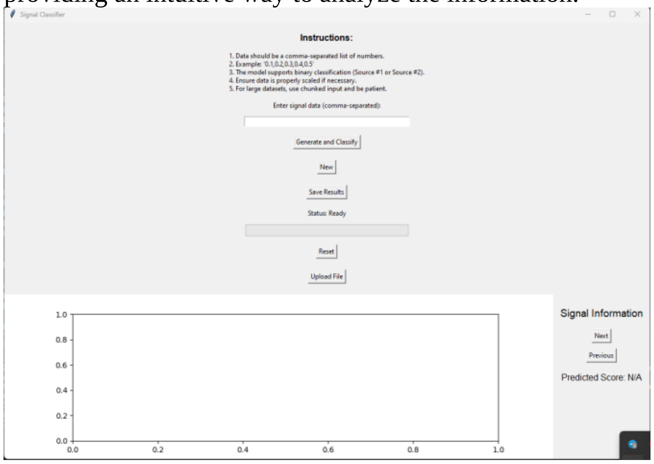

Figure 8: GUI Interface 1

The GUI in Figure 8 and 9 also features functionality for predicting signals using the trained model, allowing users to see real-time predictions alongside the plotted signals. Additionally, users have the option to save the model configurations and predictions for future reference, ensuring that all necessary information is retained and can be revisited later.

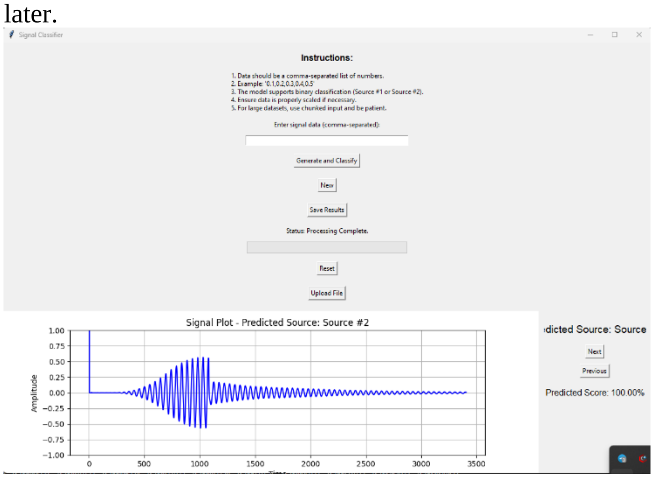

Figure 9: GUI Interface 2

Overall, this approach not only simplifies the interaction with the preserved model but also enhances user experience by integrating essential features such as data upload, signal plotting, and prediction saving, making it a comprehensive tool for signal classification tasks.

VI. CONCLUSION

This study achieved high accuracy and good generalization performance by effectively implementing a convolutional neural network (CNN) for time-series data categorization. Metrics including accuracy, unseen data prediction, ROC curve were used to assess the model's performance, and the results showed that the model performed well on both training and validation datasets. The model's good generalization to new data was further confirmed by visualizations including confusion matrices, ROC curves, and training and validation loss/accuracy curves.

One of the work's main highlights is the creation of an intuitive graphical user interface (GUI) with the Tkinter package, which enables smooth interaction with the maintained model. For a variety of signal classification applications, this tool is useful and easily accessible since it allows users to plot signals, submit data, and receive real-time predictions. The model's usefulness in real-world settings is improved by the new capability to preserve model setups and findings, which guarantees that users may save and evaluate their assessments.

In general, the suggested model together with its corresponding interface offers a flexible and efficient way to classify time-series signals. In order to increase the model's usefulness, future development may concentrate on expanding the application to more complicated datasets or adding new capabilities like batch processing or sophisticated visualization methods.

VII. AUTHOR CONTRIBUTION STATEMENT

In the preparation of this work, we both Author Bidhan Paul and Author Fahim Talukdar contributed extensively through regular discussions and collaborative efforts. Author Fahim Talukder took the lead in conceptualizing the research framework, developing the methodology, and designing the experimental setup. Author Bidhan Paul focused on implementing the models, conducting data analysis, and evaluating the results. Throughout the process, we both authors engaged in ongoing discussions to refine the approach, troubleshoot challenges, and ensure the integrity of the experiments. Author Fahim Talukdar provided critical insights on the theoretical aspects, while Author Bidhan Paul handled the technical execution. We both authors contributed equally to the writing and revision of the manuscript, interpreting the results and refining the final submission.

VIII. REFERENCES

[1] A. Lin, Z. Ma, Z. Huang, Y. Xia and W. Yu, "Unknown Radar Waveform Recognition Based on Transferred Deep Learning," in IEEE Access, vol. 8, pp. 184793-184807, 2020, doi: 10.1109/ACCESS.2020.3029192.

[2] Y. Shi, K. Davaslioglu, Y. E. Sagduyu, W. C. Headley, M. Fowler and G. Green, "Deep Learning for RF Signal Classification in Unknown and Dynamic Spectrum Environments," 2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN), Newark, NJ, USA, 2019, pp. 1-10, doi: 10.1109/DySPAN.2019.8935684.

[3] Zhang, Y., Yu, W., He, L., Cui, L., & Cheng, G. (2021). Signals classification based on IA-optimal CNN. *Neural Computing and Applications*, 33(15), 9703-9721. https://doi.org/10.1007/s00521-021-05736-x

[4] C. -K. Chang and H. -H. Chang, "Learning Entirely Unknown Classes in Time-Series Data Using Convolutional Neural Networks for Insulation Status Assessment of Partial Discharges in Power Cable Joints," in IEEE Transactions on Dielectrics and Electrical Insulation, vol. 30, no. 6, pp. 2854-2861, Dec. 2023, doi: 10.1109/TDEI.2023.3280441.

[5] S. Seo, C. Kim and J. -H. Kim, "Convolutional Neural Networks Using Log Mel-Spectrogram Separation for Audio Event Classification with Unknown Devices," in Journal of Web Engineering, vol. 21, no. 2, pp. 497-522, March 2022, doi: 10.13052/jwe1540-9589.21216.

[6] Y. Eminaga, A. Coskun and I. Kale, "Area and Power Efficient Implementation of db4 Wavelet Filter Banks for ECG Applications Using Reconfigurable Multiplier Blocks," 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), Poitiers, France, 2018, pp. 65-68, doi: 10.1109/ICFSP.2018.8552046.

[7] Mohamed, Muhidin, and Mohamed Deriche. "An approach for ECG feature extraction using daubechies 4 (DB4) wavelet." *International Journal of Computer Applications* 96.12 (2014): 36-41.

[8] Cheng, J., Zou, Q., & Zhao, Y. (2021). ECG signal classification based on deep CNN and BiLSTM. *BMC Medical Informatics and Decision Making*, 21(1), 365. https://doi.org/10.1186/s12911-021-01736-y

[9] G. Xu et al., "A Deep Transfer Convolutional Neural Network Framework for EEG Signal Classification," in IEEE Access, vol. 7, pp. 112767-112776, 2019, doi: 10.1109/ACCESS.2019.2930958.

[10] Kim, Kyong-Il, et al. "A method for underwater acoustic signal classification using convolutional neural network combined with discrete wavelet transform." *International Journal of Wavelets, Multiresolution and Information Processing* 19.04 (2021): 2050092.

[11] S. Padhy, J. Tiwari, S. Rathore and N. Kumar, "Emergency Signal Classification for the Hearing Impaired using Multi-channel Convolutional Neural Network Architecture," 2019 IEEE Conference on Information and Communication Technology, Allahabad`, India, 2019, pp. 1-6, doi: 10.1109/CICT48419.2019.9066252.