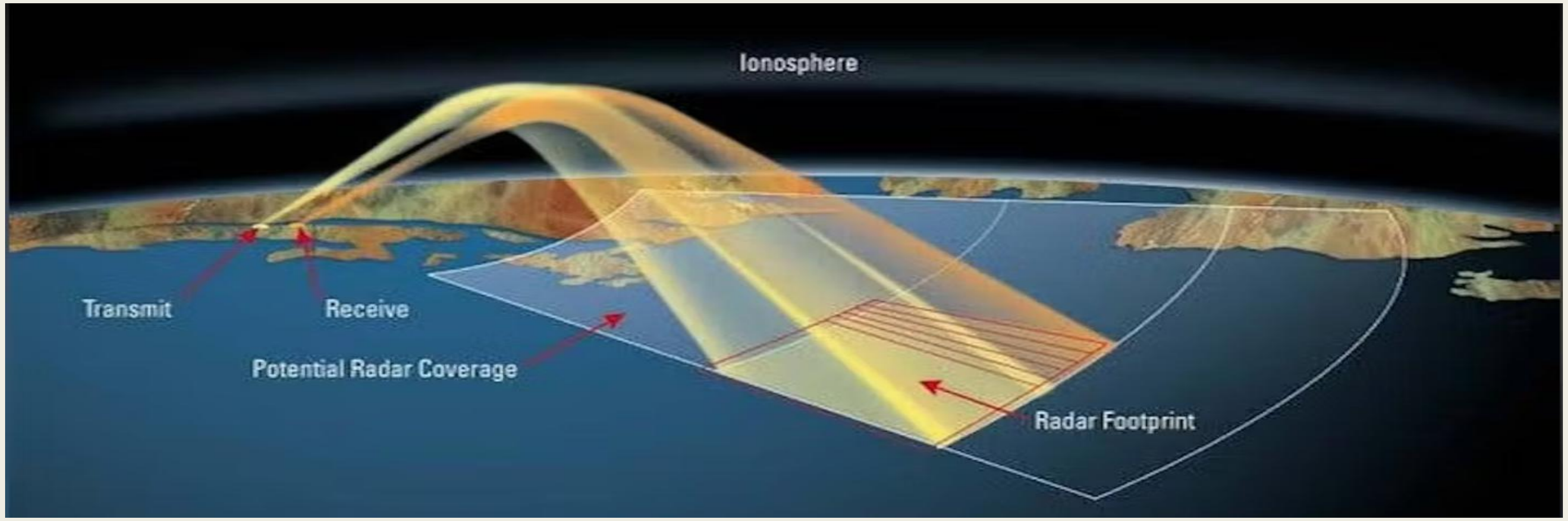
RADAR PULSE SIGNAL DETECTION USING IONOSPHERE DATASET

Group 3:
Parkavi R (CB.SC.U4AIE24338)
Maalika P(CB.SC.U4AIE24332)
Mahalakshmi I(CB.SC.U4AIE24322)
Aparna Bharani(CB.SC.U4AIE24304)

INTRODUCTION

Radar pulse signal detection plays a crucial role in modern electronic warfare, spectrum monitoring, and aerospace communication systems. Traditional signal detection techniques often rely on hand-crafted features and threshold-based decision methods, which may falter under low signal-to-noise ratio (SNR) or complex signal environments.

To overcome these limitations, this project proposes a robust deep learning-based approach that leverages Fourier Transform-Convolutional Neural Network (FT-CNN) for radar pulse signal detection.This model is compared with two other models Fourier Transform-Probabilistic Neural Network(FT-PNN) and Fourier-Transform Back Propagation Neural Network(FT-BPNN) in frequency domain and BPNN , GRNN , PNN and RBFNN in time-domain



Objective

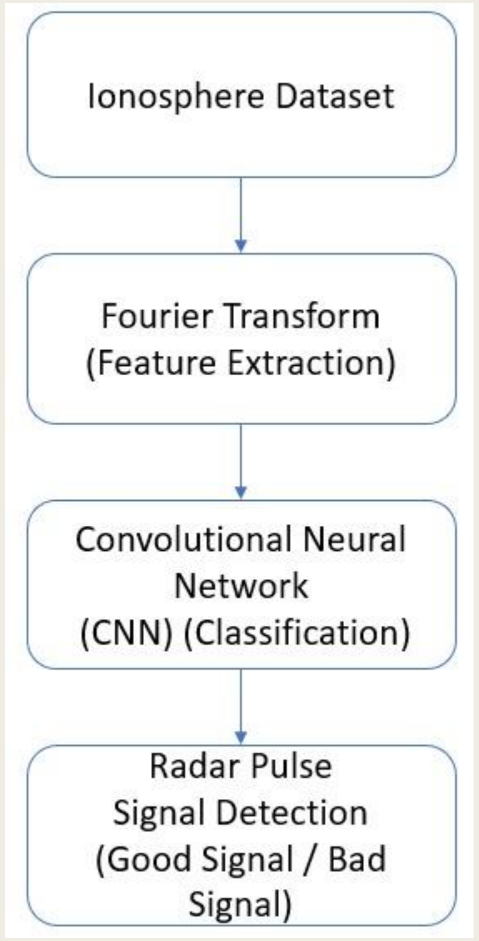
•To design an efficient model for automatic radar pulse signal detection.

•To apply Fourier Transform for signal feature extraction from the Ionosphere dataset.

•To use Convolutional Neural Networks (CNN) for classifying radar signals into Good or Bad categories.

•To improve the detection accuracy compared to traditional signal processing methods.

•To implement a simple, scalable, and efficient solution suitable for radar communication environments.


Ionosphere Dataset
Fourier Transform (Feature Extraction)
Convolutional Neural Network (CNN) (Classification)
Radar Pulse Signal Detection (Good Signal / Bad Signal)

DATASET DESCRIPTION

Dataset: Ionosphere Dataset (UCI Machine Learning Repository)

- Collected from: Radar System at University of California, San Diego
- Total Samples: 351 Radar Signal Instances
- Number of Features: 34 Continuous Numerical Features
- Target Classes:
  - *Class 1: "Good" Signal → Strong Radar Pulse Detected*
  - *Class 2: "Bad" Signal → Weak or No Radar Pulse Detected*
- Data Format:
  - *Each Sample = Set of 34 Signal Attributes*
  - *Captures Radar Wave Reflection Characteristics*
- Data Characteristics:
  - *Real-Valued Data*
  - *No Missing Values*
  - *Imbalanced Dataset*

Fourier Transform

- The Fourier Transform (FT) is a mathematical operation that transforms a time-domain signal into its frequency-domain representation.
- Mathematical Formula:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

Where

X(f) is the Fourier Transform of the signal $x(t)$.

f represents frequency.

$e^{-j2\pi ft}$ is the complex exponential that represents the frequency components of the signal.

- Frequency-domain conversion reveals key patterns in ionospheric signals. It highlights spectral features useful for classification. This helps the CNN detect anomalies more effectively.

BACKEND WORKFLOW

Step 1: Dataset

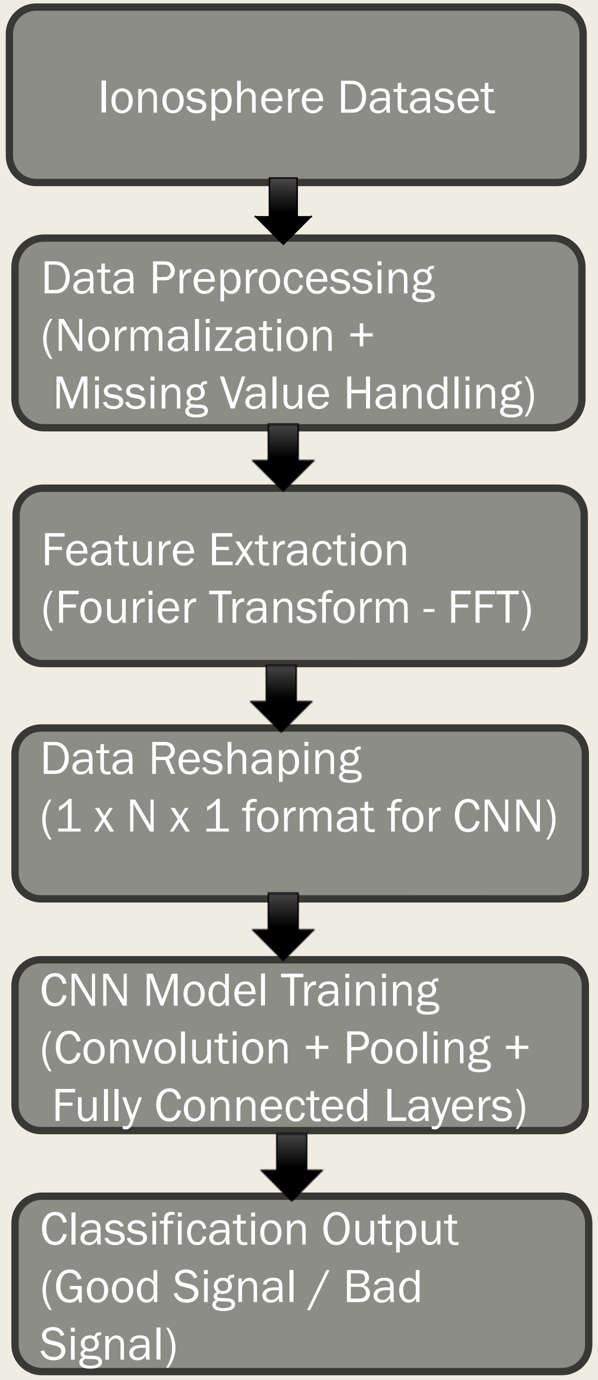
- Ionosphere Dataset from UCI Machine Learning Repository
- Contains Radar signal characteristics (Features)
- Target Label → Classifies signals as:
  - *"Good" → No noise, clear signal*
  - *"Bad" → Noisy or unclear signal*

Step 2: Data Preprocessing

- Split the dataset → 80% for training, 20% for testing
- Normalized data → To ensure all feature values are in the same range
- Missing values → Filled with mean values

Step 3: Feature Extraction – Fourier Transform (FT)

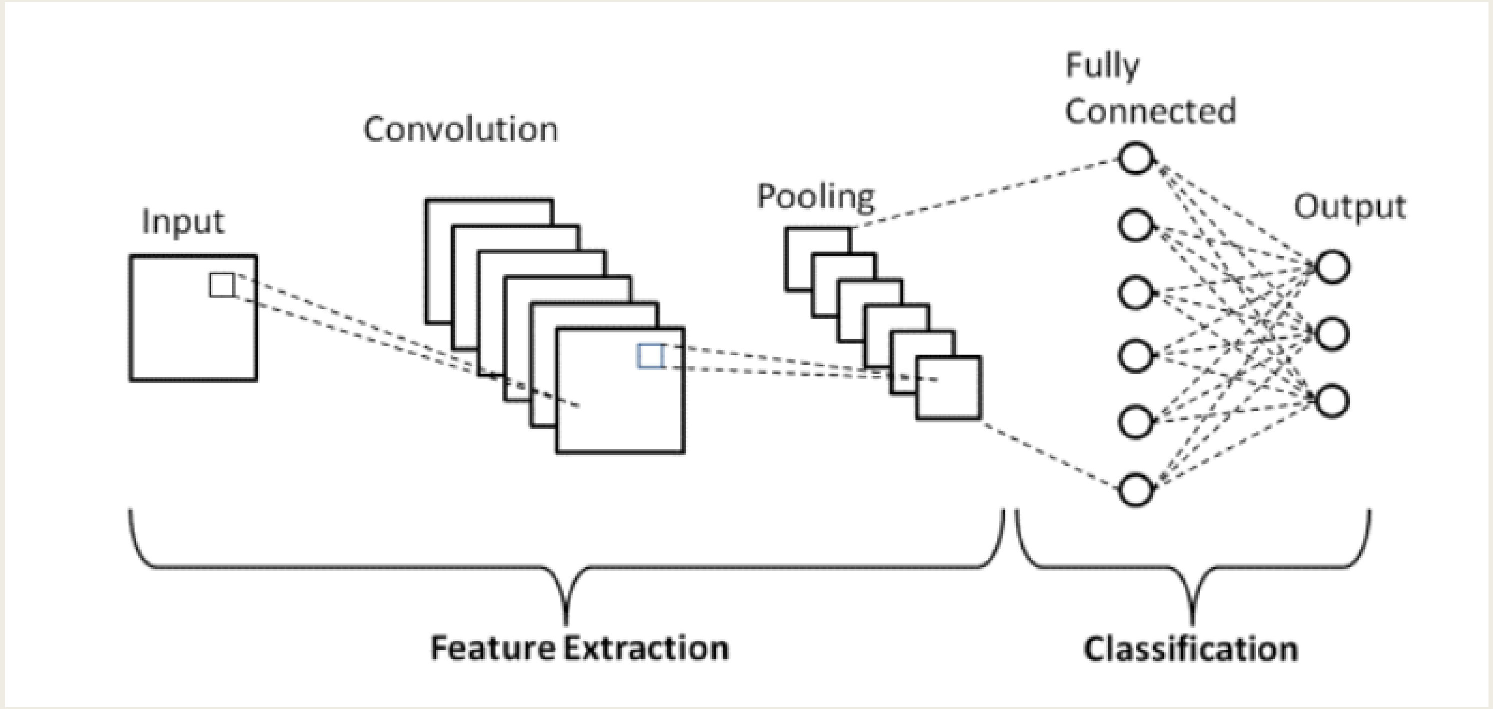
- FT converts time domain to frequency domain
- Better capture of signal patterns and periodic features
- Used absolute value of FT output.


Ionosphere Dataset
Data Preprocessing (Normalization + Missing Value Handling)
Feature Extraction (Fourier Transform - FFT)
Data Reshaping (1 x N x 1 format for CNN)
CNN Model Training (Convolution + Pooling + Fully Connected Layers)
Classification Output (Good Signal / Bad Signal)

CNN ARCHITECTURE

In this project, we have designed a specialized 1D Convolutional Neural Network (CNN) model for detecting radar pulse signals from the Ionosphere dataset. The dataset after preprocessing (FT feature extraction) was reshaped for compatibility with a 1D CNN.

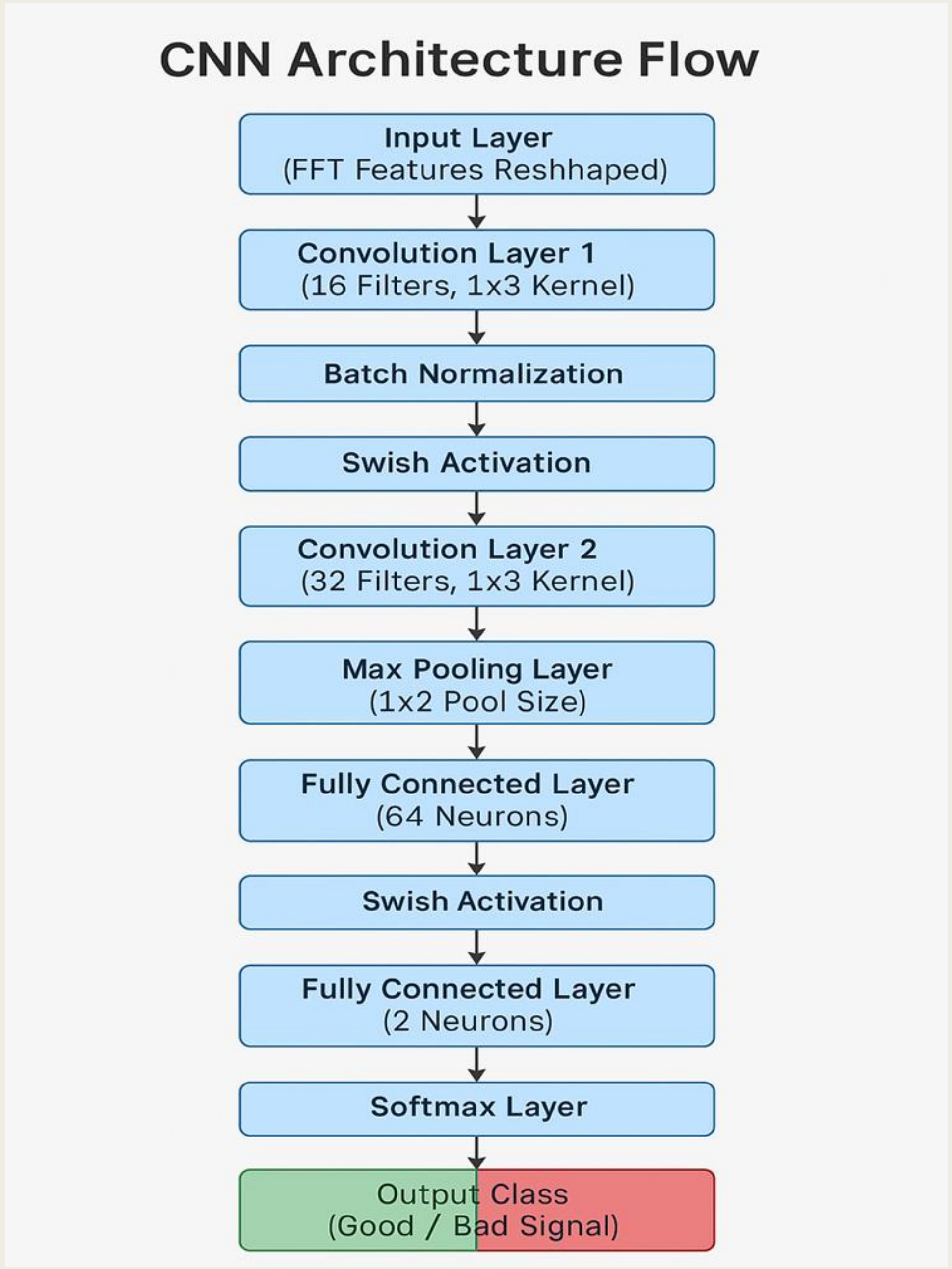
We used Swish activation function instead of ReLU to enhance the non-linearity capability of the model, improving the learning of complex signal features.



Layer No.	Layer Name	Type	Configuration	Purpose
1	Input Layer	Image Input Layer	Size = [1 × Number of Features × 1]	Accepts the reshaped input FFT data for the CNN.
2	Conv Layer 1	2D Convolution	Filter size = [1×3], Filters = 16, Padding = Same	Detects local patterns in the 1D signal across 3 neighboring values.
3	Batch Normalization	Batch Norm Layer	Normalizes data	Accelerates training, reduces sensitivity to weight initialization.
4	Swish Activation	Activation Layer	Swish(x) = x × sigmoid(x)	Provides smoother gradient flow and better performance than ReLU.
5	Max Pooling 1	Pooling Layer	Pool Size = [1×2], Stride = [1×2]	Reduces feature map size, retains important features.
6	Conv Layer 2	2D Convolution	Filter size = [1×3], Filters = 32, Padding = Same	Learns higher-level features from pooled output.
7	Batch Normalization	Batch Norm Layer	Normalizes data	Helps stabilize learning.
8	Swish Activation	Activation Layer	Swish(x) = x × sigmoid(x)	Ensures non-linear learning capability continues.
9	Max Pooling 2	Pooling Layer	Pool Size = [1×2], Stride = [1×2]	Further reduces dimension, controls overfitting.
10	Fully Connected Layer 1	Dense Layer	64 Neurons	Learns complex feature combinations for classification.
11	Swish Activation	Activation Layer	Swish(x)	Enables complex non-linearity.
12	Fully Connected Layer 2	Dense Layer	2 Neurons	Final feature mapping for binary classification (Target / Non-Target Signal).
13	Softmax Layer	Classification Layer	Softmax Activation	Converts output to probability distribution.
14	Output Layer	Classification Layer	Cross Entropy Loss	Final output prediction (Class 1 or Class 2).

CNN Architecture Flow

1. Data Preprocessing:
    1. *FFT applied on Ionosphere Dataset*
    2. *Data Normalization*
    3. *Reshape into 1xNx1 format suitable for CNN*
2. Feature Extraction:
    1. *Convolution Layers detect local patterns*
    2. *Swish Activation improves gradient flow*
3. Dimensionality Reduction:
    1. *MaxPooling reduces feature size*
    2. *Controls overfitting*
4. Classification:
    1. *Fully Connected Layers learn global patterns*
    2. *Softmax Layer provides final class probability*


CNN Architecture Flow
Input Layer (FFT Features Reshhaped)
Convolution Layer 1 (16 Filters, 1x3 Kernel)
Batch Normalization
Swish Activation
Convolution Layer 2 (32 Filters, 1x3 Kernel)
Max Pooling Layer (1x2 Pool Size)
Fully Connected Layer (64 Neurons)
Swish Activation
Fully Connected Layer (2 Neurons)
Softmax Layer
Output Class (Good / Bad Signal)

Back Propagation Neural Network

- BPNN is a type of artificial neural network that learns from labeled data using supervised learning.
- It uses the **backpropagation algorithm** to minimize the error by adjusting the weights through gradient descent.
- Consists of input, hidden, and output layers where each neuron applies an activation function (e.g., sigmoid, ReLU)
- The network performs **forward propagation** to make predictions and **backward propagation** to update weights based on error.

Probabilistic Neural Network

- PNN is a supervised neural network based on Bayes’ theorem and Parzen window estimation for classification.
- Consists of four layers — input, pattern, summation, and output layers — where each pattern neuron represents a training sample.
- Best suited for classification tasks, especially when the decision boundary is non-linear and datasets are noisy.

General Regression Neural Network

- GRNN is a type of neural network based on statistical estimation, suitable for **both regression and classification** problems.
- Composed of four layers — input, pattern, summation, and output — using a **Gaussian kernel** to compute distances from training samples.
- GRNN works like a smart **pattern matcher** that looks at the examples it has seen before and finds the most similar ones to make a decision, which makes it easy to understand and apply for classification.

Radial Basis Function Neural Network

- RBFNN looks for patterns by comparing new data to examples it has already seen.
- It has different layers: the **input layer** brings in the data, the **hidden layer** finds patterns, and the **output layer** gives the result.
- It’s great for solving problems where the data isn’t a straight line, like classifying images or predicting outcomes.

RESULTS

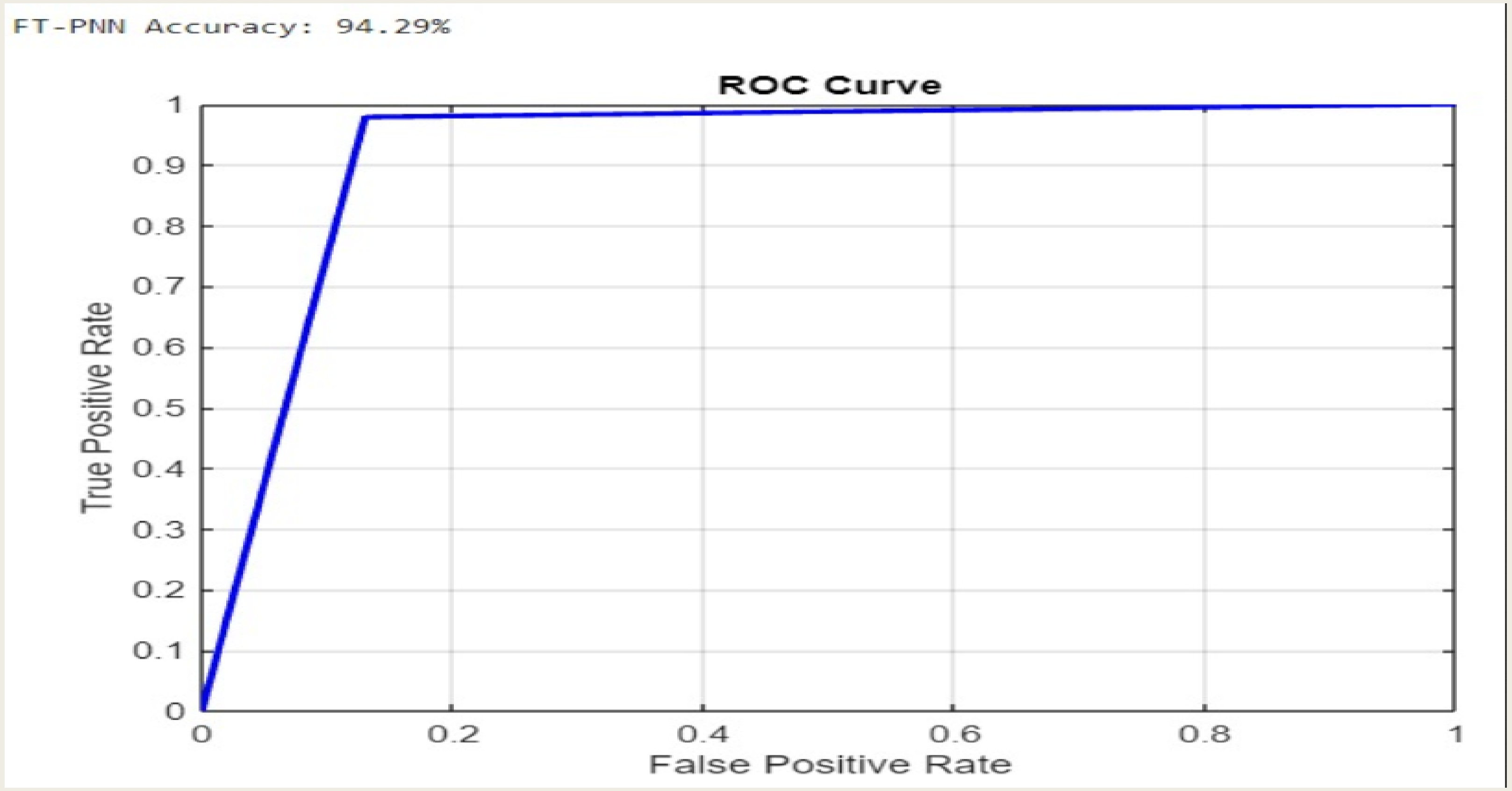
The performance of the proposed Fourier Transform-based Convolutional Neural Network (FT-CNN) model was evaluated using the Ionosphere Dataset. The model was trained using the Adam optimizer with over 50 epochs.

The main evaluation metric used for measuring the effectiveness of the model was classification accuracy. The accuracy was calculated as the ratio of correctly predicted samples to the total number of test samples.

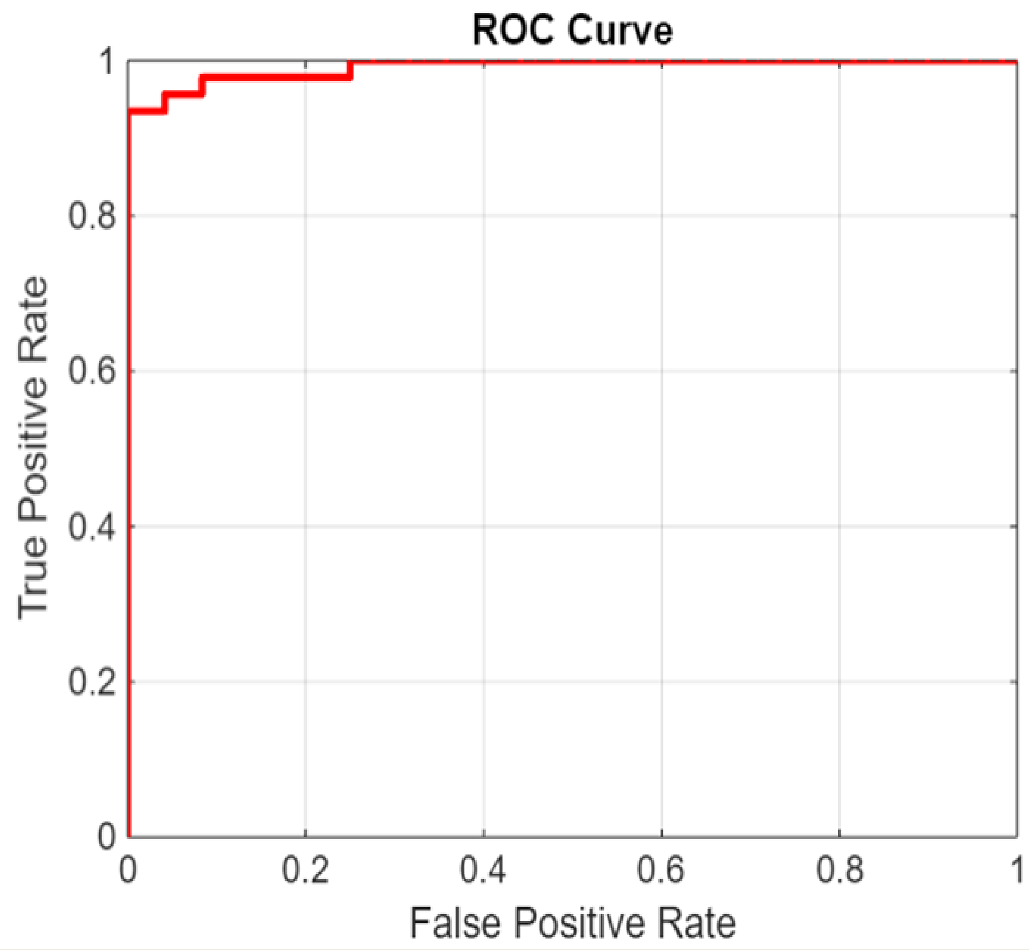
Upon completion of training, the FT-CNN model achieved an impressive accuracy of 98.57%, outperforming conventional machine learning models like FT-PNN (94.29%) and FT-BPNN(91.43%).

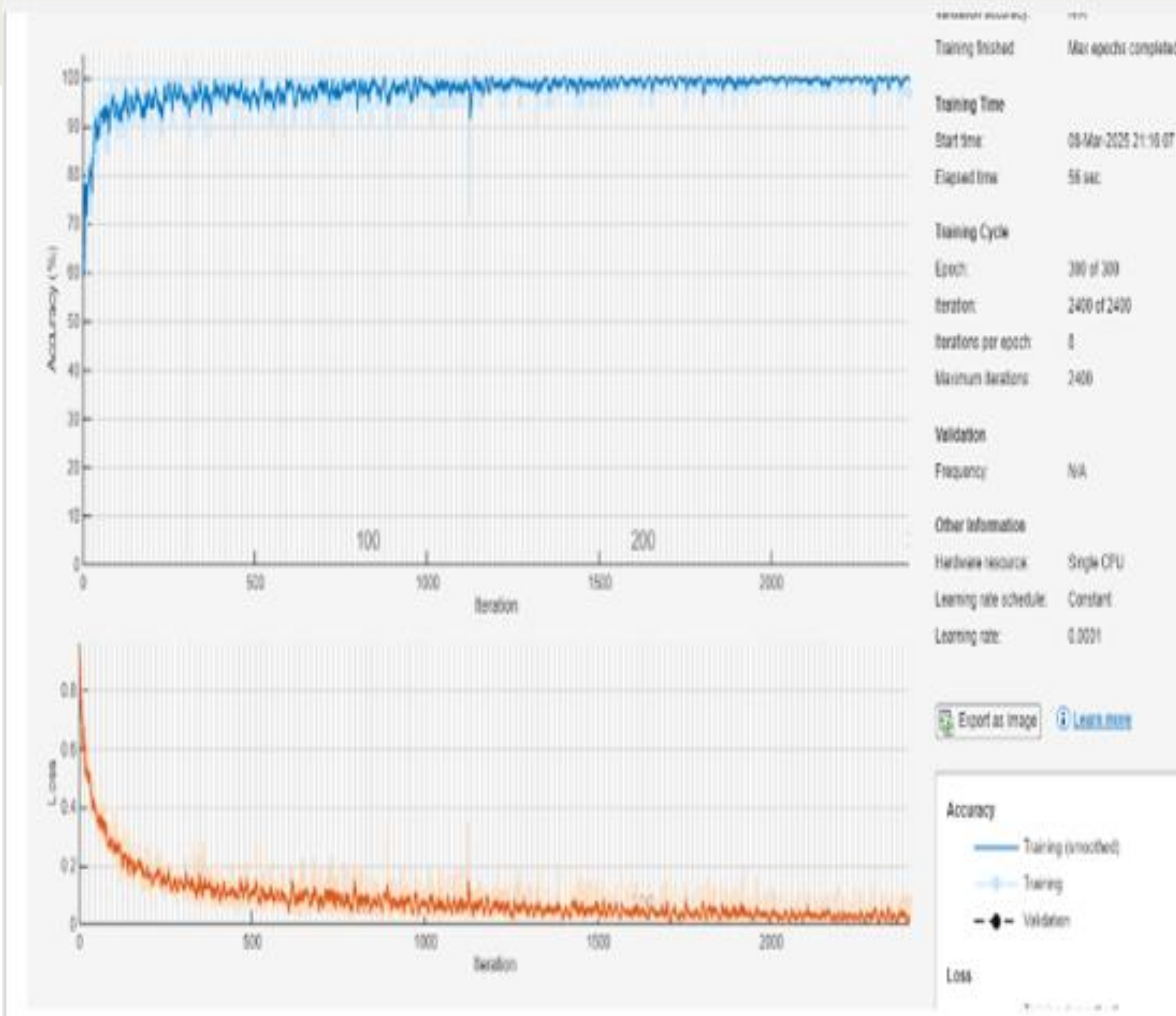
This indicates that the proposed deep learning approach, combined with frequency domain feature extraction through FT and the usage of Swish activation function, significantly enhanced the model's capability to differentiate between good and bad radar signals.

Model	Accuracy Achieved (%)
FT-PNN	94.29 %
FT-BPNN	91.43 %
FT-CNN	98.57 %


```
FT-PNN Accuracy: 94.29%
```

```
FT-BPNN Accuracy: 91.43%
```





Training finished: Max epochs completed

Training Time
Start time: 08-Mar-2025 21:16:07
Elapsed time: 56 sec

Training Cycle
Epoch: 300 of 300
Iteration: 2400 of 2400
Iterations per epoch: 8
Maximum iterations: 2400

Validation
Frequency: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.0001

```
FT-CNN Accuracy: 98.5714%
```

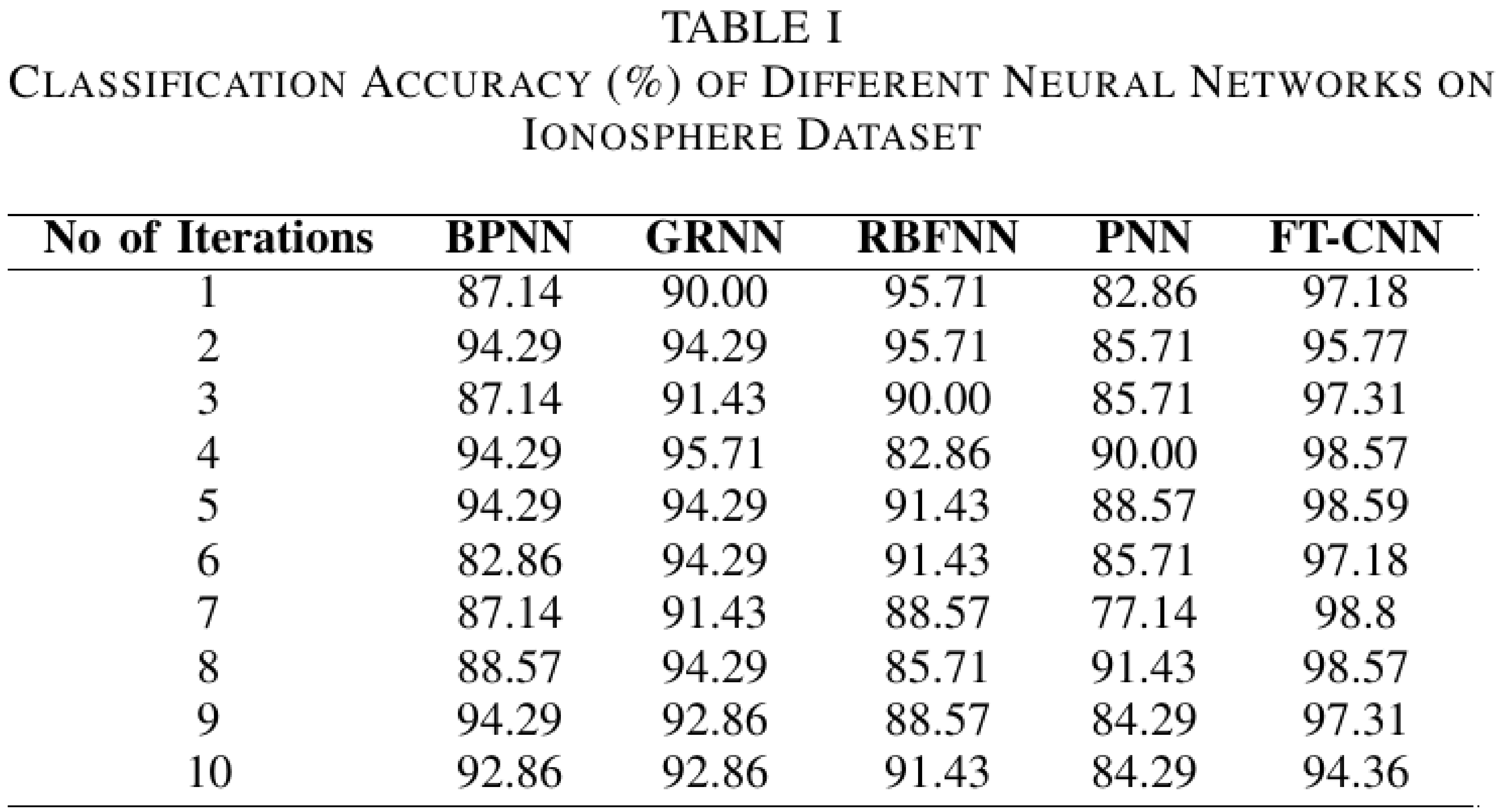
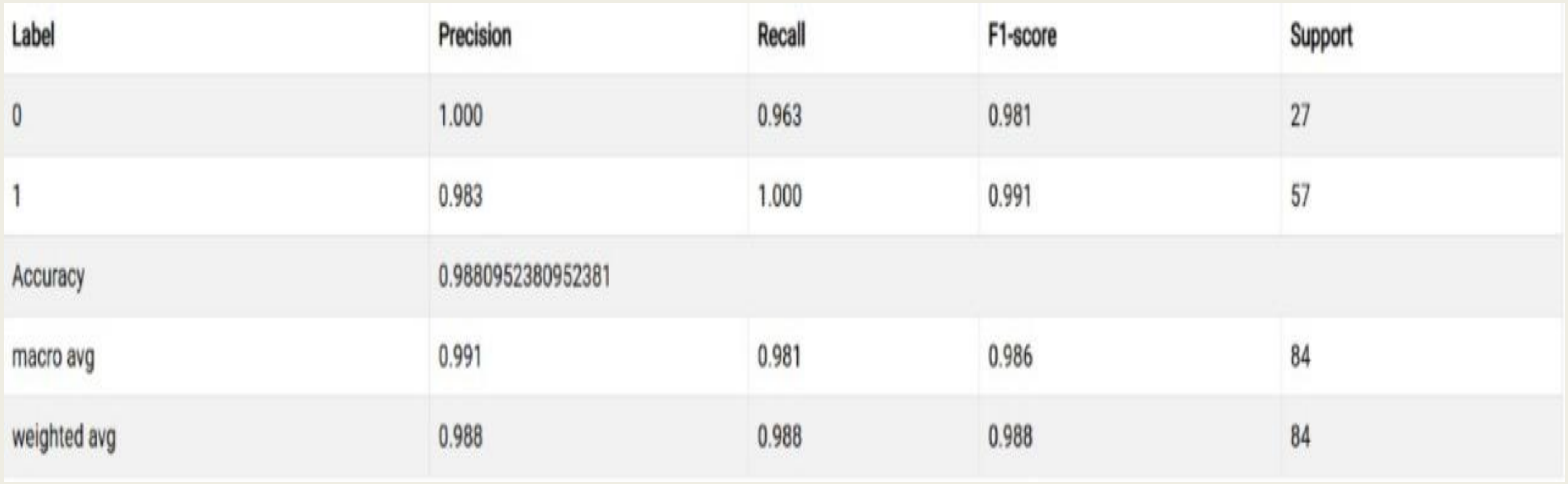


TABLE I
CLASSIFICATION ACCURACY (%) OF DIFFERENT NEURAL NETWORKS ON IONOSPHERE DATASET

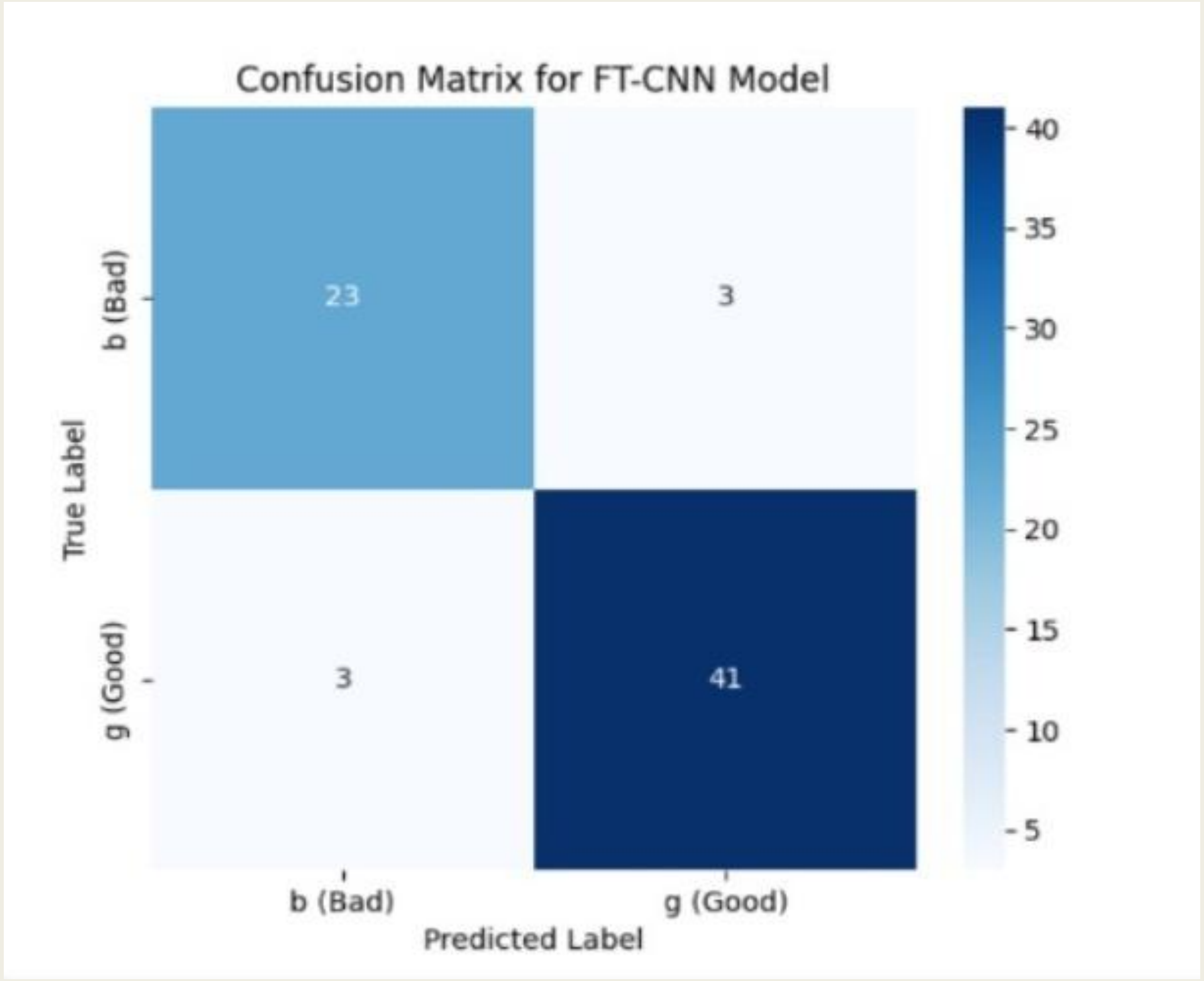
No of Iterations	BPNN	GRNN	RBFNN	PNN	FT-CNN
1	87.14	90.00	95.71	82.86	97.18
2	94.29	94.29	95.71	85.71	95.77
3	87.14	91.43	90.00	85.71	97.31
4	94.29	95.71	82.86	90.00	98.57
5	94.29	94.29	91.43	88.57	98.59
6	82.86	94.29	91.43	85.71	97.18
7	87.14	91.43	88.57	77.14	98.8
8	88.57	94.29	85.71	91.43	98.57
9	94.29	92.86	88.57	84.29	97.31
10	92.86	92.86	91.43	84.29	94.36

CLASSIFICATION REPORT

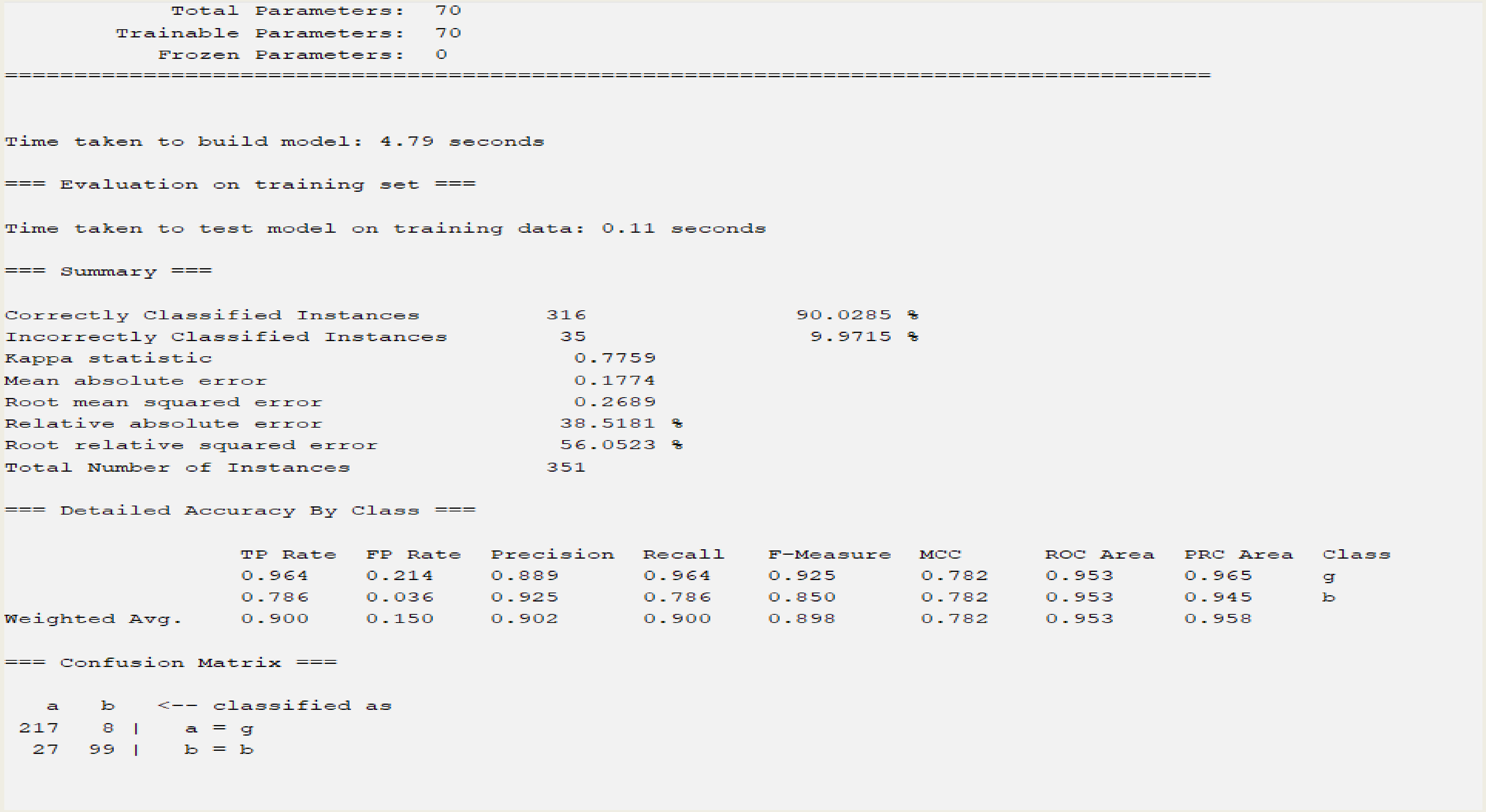


Label	Precision	Recall	F1-score	Support
0	1.000	0.963	0.981	27
1	0.983	1.000	0.991	57
Accuracy	0.9880952380952381			
macro avg	0.991	0.981	0.986	84
weighted avg	0.988	0.988	0.988	84

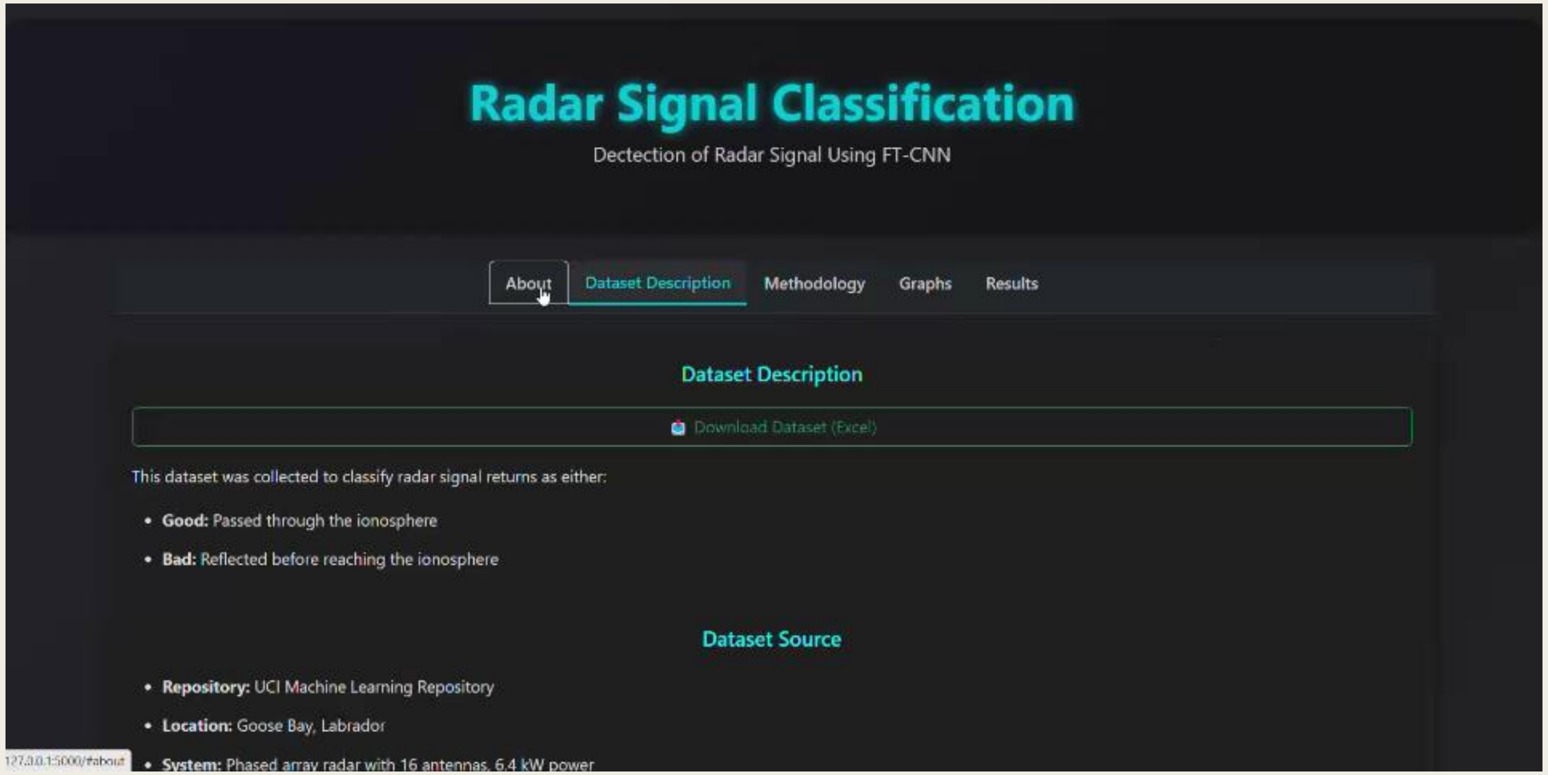
CONFUSION MATRIX



WEKA RESULTS


```
            Total Parameters:   70
        Trainable Parameters:   70
           Frozen Parameters:   0
======================================================================================


Time taken to build model: 4.79 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.11 seconds

=== Summary ===

Correctly Classified Instances         316               90.0285 %
Incorrectly Classified Instances        35                9.9715 %
Kappa statistic                          0.7759
Mean absolute error                      0.1774
Root mean squared error                  0.2689
Relative absolute error                 38.5181 %
Root relative squared error             56.0523 %
Total Number of Instances              351

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.964    0.214    0.889      0.964    0.925      0.782    0.953     0.965     g
                 0.786    0.036    0.925      0.786    0.850      0.782    0.953     0.945     b
Weighted Avg.    0.900    0.150    0.902      0.900    0.898      0.782    0.953     0.958

=== Confusion Matrix ===

   a   b   <-- classified as
 217   8 |   a = g
  27  99 |   b = b
```

USER INTERFACE



Radar Signal Classification

Dectection of Radar Signal Using FT-CNN

Dataset Description

This dataset was collected to classify radar signal returns as either:

- **Good:** Passed through the ionosphere
- **Bad:** Reflected before reaching the ionosphere

Dataset Source

- **Repository:** UCI Machine Learning Repository
- **Location:** Goose Bay, Labrador
- **System:** Phased array radar with 16 antennas, 6.4 kW power

CONCLUSION

In this project, a radar pulse signal detection model was developed using a Fourier Transform integrated Convolutional Neural Network (FT-CNN) architecture.

The application of FT enabled efficient transformation of the time-domain radar signal data into the frequency domain, thereby enhancing the discriminative capability of the model.

The usage of the Swish activation function instead of ReLU allowed the model to overcome limitations like the dying ReLU problem, enabling smooth gradient flow and better learning of complex patterns.

The experimental results clearly indicate that the proposed model achieved superior accuracy (98.57%) compared to conventional methods. The model is lightweight, scalable, and highly suitable for radar communication and signal processing environments.

FUTURE ENHANCEMENTS

Although the proposed FT-CNN model has shown promising results, there are several avenues for future enhancement. The model can be extended by integrating hybrid deep learning architectures such as CNN-LSTM, which can handle sequential dependencies in radar signal data more effectively.

Furthermore, real-time implementation of the model on embedded systems or FPGA platforms would be an ideal extension for defense applications. Incorporating more diverse radar datasets and performing transfer learning could help in improving the generalization capability of the model.

Additionally, research can be directed towards designing noise-resilient feature extraction techniques and exploring advanced optimization algorithms for further performance improvements.

REFERENCES

Detection of Radar Pulse Signals Based on Deep Learning FENGYANG GU,LUXIN ZHANG, SHILIAN ZHENG,JIECHEN ,KEQIANG YUE, ZHIJIN ZHAO AND XIAONIU YANG

https://ieeexplore.ieee.org/document/10614929/

VINCENT G. SIGILLITO, SIMON P. WING, LARRIE V. HUTTON, and KILE B. BAKER CLASSIFICATION OF RADAR RETURNS FROM THE IONOSPHERE USING NEURAL NETWORKS

https://secwww.jhuapl.edu/techdigest/content/techdigest/pdf/

THANK YOU