I.J. Intelligent Systems and Applications, **2026, 4, 138-160**
Published Online on August 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2026.04.08

Enhanced Robustness of Neural Network Models against Adversarial Attacks and Performance Analysis

Surekha M.*
Computer Science and Engineering, Sharda University, Greater Noida, Uttar Pradesh, India
JSS Academy of Technical Education, Noida, Uttar Pradesh, India
E-mail: surekhashivakumar@gmail.com
ORCID iD: https://orcid.org/0000-0002-7338-8267
*Corresponding author

Anil Kumar Sagar
Computer Science andEngineering, Sharda University, Greater Noida, Uttar Pradesh, India
E-mail: anil.sagar@sharda.ac.in
ORCID iD: https://orcid.org/0000-0002-5991-2835

Vineeta Khemchandani
Computer Science and Engineering, Galgotias University, Greater Noida, Uttar Pradesh, India
E-mail: vineetakh05@gmail.com
ORCID iD: https://orcid.org/0000-0002-7934-8493

Received: January 4, 2026; Revised: March 15, 2026; Accepted: June 7, 2026; Published: August 8, 2026

Abstract: Machine learning models, particularly deep learning architectures, achieve high performance in prediction tasks but remain susceptible to adversarial attacks. This study aims to enhance the robustness of Convolutional Neural Networks (CNNs), Deep Neural Networks (DNNs), and Recurrent Neural Networks (RNNs), thereby improving the security of machine learning systems. A three-step approach is adopted. First, benign sample classification is performed using the MNIST benchmark dataset. Second, adversarial attacks, namely Projected Gradient Descent (PGD), DeepFool (DF), and the Fast Gradient Sign Method (FGSM), are launched on the trained models, resulting in significant performance degradation. Based on the biased outputs induced by adversarial perturbations, an adversarial detection model is subsequently established. Third, to counteract these attacks, various defense strategies, including adversarial training, defensive distillation, autoencoder-based denoising, ensemble methods, and feature squeezing are employed and evaluated using standard performance metrics and graphical analyses. The results indicate that, in the absence of defense mechanisms, PGD attacks lead to accuracy drops of approximately 27% in CNNs, 83% in DNNs, and 90% in RNNs, demonstrating severe model vulnerabilities. However, when defense strategies are applied, all models recover to an accuracy of at least 98.9%, with adversarial training improving performance under attack by up to 90%. Among the evaluated models, CNNs exhibit the highest baseline robustness, whereas DNNs and RNNs rely more heavily on defense mechanisms to maintain performance. These findings provide valuable insights into the development of secure and resilient machine learning systems capable of mitigating adversarial threats.

Index Terms: Defense Strategy, Adversarial Attack, Neural Network Models, Machine Learning Robustness, Performance Evaluation Metrics.

1. Introduction

Machine Learning (ML) systems are being used in many critical areas, but they can also be affected by hazards that can render them less reliable and may endanger public safety. For example, adversarial examples in Natural Language Processing can affect how text classifiers interpret input and how well they perform overall [1]. When the input to a diagnostic system is changed or modified, it may result in improper clinical decisions and therefore put patients at risk [2]. Autonomous vehicles and surveillance systems are particularly susceptible to

adversarial examples that could cause safety-critical system malfunctions [3]. Financial institutions rely on ML systems for applications such as high-frequency trading and fraud detection. If these systems are subverted through adversarial means, they could potentially suffer significant financial losses [4]. The malicious adversarial threats mentioned above create a demand for ML systems that can continue to function as originally designed when faced with adversarial scenarios. Mitigating these persistent risks requires a comprehensive strategy for the proactive governance and implementation of technical countermeasures, as well as a continuous improvement process to enhance the adaptability of these systems over time. This research compares multiple neural network architectures, including CNNs, DNNs, and RNNs, to evaluate their performance under adversarial inputs [5-11]. Ultimately, this research aims to develop robust and flexible defense mechanisms to enhance the security and resilience of neural network models.

This research is designed to systematically evaluate the robustness of various neural network architectures against adversarial attacks and provides a comprehensive evaluation of various existing defense mechanisms to address current research gaps. We generated adversarial examples using the MNIST dataset for classification tasks using FGSM, PGD, and DeepFool to evaluate the susceptibility of CNN, DNN, and RNN architectures to adversarial attacks. We quantified the susceptibility of our models by comparing the predictions made by the model before and after the adversarial attack and measuring the degree of performance loss. All models achieved a baseline accuracy of greater than 98.9% on clean inputs prior to experiencing any adversarial attacks; however, adversarial attacks greatly diminished these accuracies. In the case of models that utilized PGD, the accuracy loss was approximately 27% for CNNs, 83% for DNNs, and 90% for RNNs, when no defense had been applied. The application of five defensive strategies significantly improved the models' performance across all three architectures. Post-defense accuracy for each model was greater than 98.9% with adversarial training providing the greatest resistance to attacks and restoring accuracy to over 98.9%. While CNNs exhibited an increased degree of inherent robustness, greater reliance on defensive mechanisms was observed within the DNN and RNN architectures. Our findings present a useful starting point for the design of secure, resilient machine learning solutions capable of operating in adversarially influenced environments.

2. Related Work

Collectively, the aforementioned articles provide an understanding of Adversarial Machine Learning (AML) attacks and methods of defence against those attacks across many different domains. Reference [12] expands on AML theories by establishing a structured categorization and definition system to enhance clarity. However, it lacks sufficient guidance for the practical implementation of AML. Reference [13] examines AML from an industrial perspective, highlighting the growing significance of adversarial attacks on real-world ML systems. Reference [14] surveys major attacks and defensive measures in deep learning applied to cybersecurity. However, because no experimental verification was performed, the practical effectiveness of these defensive measures remains unverified. Reference [15] presents a detailed analysis of adversarial effects in physical-world applications. Reference [16] describes the application of RNN-Guard, a certified defence framework, to improve RNN robustness against multi-frame attacks. RNN-Guard is supported by strong theoretical guarantees and empirical validation. Reference [17] confirms that both statistical and deep learning time series models remain vulnerable to adversarial perturbations, thus demonstrating the need for temporally adaptive methods of defence. Reference [18] describes TSFool, an effective multi-objective attack that generates imperceptible perturbations across multiple domains. Reference [19] introduces a type of stealthy sequential attack which maintains temporal consistency while achieving high attack success rates. Reference [20] evaluates the threat of adversarial attacks in Natural Language Processing (NLP), with an emphasis on the issue of robustness with regard to embeddings and transformer-based models, as well as future directions for research in this area. While many resources are available for research on this topic (e.g., [21]), most prior studies have focused on physical-world threats, leaving a gap in our understanding of purely digital forms of threat. Most existing studies on adversarial risk and vulnerabilities in healthcare lack specific adversarial machine learning (AML) defenses for clinical systems or workflows. Overall, the existing literature indicates a need for practical, adaptive, and empirically validated AML defenses across various domains, including critical infrastructure and healthcare, to address these gaps. In addition, significant gaps remain in the availability of real-time frameworks for testing, robust assessments of ML models, and deployable AML systems in critical techno-national security areas, including cybersecurity and healthcare.

3. Methodology

3.1. Conceptual Framework and Experimental Design

In order to assess and improve the robustness of neural networks, the proposed methodology utilizes a three-phase framework. The first phase establishes a baseline for the performance of CNNs, DNNs, and RNNs using the MNIST dataset with no adversarial modifications. The second phase involves the generation of adversarial examples through FGSM, PGD, and DeepFool attacks to assess the susceptibility of each model to these forms of attacks. These attacks allow researchers to conduct a more detailed analysis of a model's vulnerability when subject to different levels of attack complexity. Finally, in the third phase, five different defense strategies (adversarial training, defensive distillation, feature squeezing, denoising autoencoders, and ensemble learning) are applied to the models to improve their resilience to adversarial examples. After applying the defense strategies, the models were analyzed based on performance

degradation and detection-based evaluations comparing benign and adversarial predictions. This analysis was conducted using standard classification metrics and graphical representations of the results. Based on the analysis conducted for all models tested, CNNs exhibited the highest baseline and post-defense performance; DNNs ranked second in robustness; and RNNs were the most susceptible to attacks. Adversarial training consistently provided the best overall robustness improvement across all defensive strategies.

3.2. Methodology Steps Followed in Proposed Model

Fig. 1 illustrates both the proposed framework and the baseline training performance of CNN, DNN, and RNN models on the MNIST dataset. The goal of the pipeline was to implement a system that accurately predicts handwritten digits (0-9), with performance evaluated using a set of metrics. Subsequently, white-box attacks (FGSM, PGD, and DeepFool) were conducted against the trained models to quantify adversarial vulnerability by measuring the mislabeling of original classes. By measuring the performance disparity between baseline and attacked systems, we demonstrate that attacks introduce prediction biases, enabling adversarial detection by comparing pre-attack and post-attack labeling accuracy. To counter this reduction in performance and to increase the overall safety of these systems, several defense strategies were employed, including adversarial training, defensive distillation, denoising autoencoders, ensemble learning, and feature squeezing. The framework consists of interconnected stages: data collection and preprocessing, attack evaluation, and defense implementation to improve robustness.

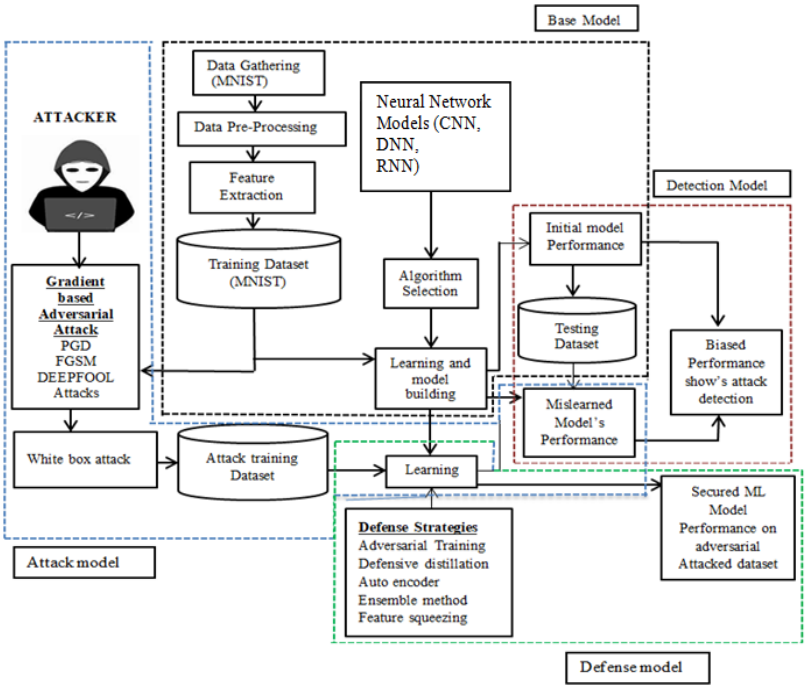


Fig.1. Proposed framework for enhancing the robustness of neural network models (CNN, DNN, RNN) against adversarial attacks

A. Data Gathering

MNIST is a standard benchmark dataset containing 70,000 grayscale (single-channel) images of handwritten digits (0–9). The dataset was split into 60,000 training images (further divided into 54,000 for training and 6,000 for validation) and 10,000 test images.

B. Data Preprocessing and Feature Extraction

To prepare the MNIST dataset for CNN, DNN, and RNN inputs, pixel intensities were normalized and reshaped into appropriate formats (samples, time steps, features). The images retained their original 28 × 28 grayscale resolution for training. Of the 70,000 digit images in the dataset, 60,000 were used for training (divided into 54,000 training and 6,000 validation images) and 10,000 for testing. Digit classification was performed using CNN, DNN, and RNN architectures to learn distinguishing features from the grayscale images. To determine robustness, we generated white-box adversarial examples, measured accuracy degradation, and evaluated model recovery using various defense mechanisms.

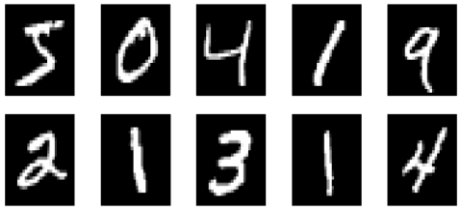


Fig.2. MNIST training samples

C. Algorithm Steps

The CNN model, based on previous studies [22-26], was implemented with several adversarial attack and defense strategies for MNIST data samples. Input image is represented as a 2D matrix $X \in R^{H \times W}$, where H and W denote height and width respectively. The CNN model is denoted as f_θ *where* θ includes convolution filter weights, biases, and fully connected layer parameters. The final output is a probability vector $y \in R^C$ for C classes.

Its steps are as follows: Initialize Model Parameters: Initialize convolution filter weights W_c, biases b_c , fully connected layer weights W_{fc}, and output layer weights W_y, with corresponding biases b_{fc}, b_y.

Step 1. Process the Input Image:

- Apply convolution operation:

$$Z^{(1)} = W_c * X + b_c \tag{1}$$

- Apply activation function (ReLU):

$$a^{(1)} = \text{ReLU}(z^{(1)})$$

- Apply pooling (Max Pooling):

$$p^{(1)} = \text{MaxPool}(a^{(1)})$$

Step 2. Flatten and Fully Connected Layer:

- Flatten pooled feature map and compute hidden activations

$$h = ReLU(W_{fc} \cdot Flatten\ (p^{(1)} + b_{fc}) \tag{2}$$

Step 3. Compute Output:

- Compute final output using softmax activation:

$$y = softmax(W_y \cdot h + b_y) \tag{3}$$

Step 4. Train the CNN (if training):

- Compute the loss (cross-entropy loss for classification).
- Perform backpropagation to compute gradients.
- Update model parameters θ using gradient descent or Adam optimizer.

Step 5. Return Output:

Return the predicted output $y \in \mathbb{R}^c$

D. Algorithm Steps

DNN model studied through few references [27-30] and implemented in work with few adversarial attack and defense strategies for MNIST data samples. The input is a flattened vector $x \in \mathbb{R}^d$, where d = H·W (784 for MNIST). The DNN model f_θ contains multiple fully connected layers with nonlinear activations, and the output is a probability vector $y \in \mathbb{R}^c$.

Its steps are as follows: Initialize Model Parameters: Initialize weight matrices $W^{(l)}$ and bias vectors $b^{(l)}$ for each layer l=1,2,...,L where L is the total number of layers.

Step 1. Process the Input through Layers: Let the input to the first layer be $h^{(0)} = \boldsymbol{x}$. For each layer l=1 to L−1:
Step 2. Compute hidden layer activation using ReLU:

$$h^{(l)} = ReLU\,(\mathrm{W}^{(l)}\ \ h^{(l-1)} + b^{(l)}) \tag{4}$$

Step 3. Compute Output: For the output layer L:

$$y = softmax\,(W^{(L)}\,h^{(L-1)} + b^{(L)}) \tag{5}$$

Step 4. Train the DNN (if training):

- Compute the loss using cross-entropy between predicted and true labels.
- Perform standard backpropagation to compute gradients.
- Update parameters $\theta = \{\mathrm{W}^{(l)}, \mathrm{b}^{(l)}\}$ using gradient descent or similar optimization algorithm.

Step 5. Return Output: Return final output vector $\mathrm{y} \in \mathbb{R}^{\mathrm{c}}$

E. Algorithm Steps

The RNN model, building on previous work [31–33], was tested on MNIST against various adversarial attacks and defense methods for that model, although it was not able to utilize its complete ability to model the temporal aspect of the data. The 28 × 28 images in the MNIST dataset were relatively restructured so that each 28 × 28 image became a sequence of 28 time steps, each representing an individual row of 28 pixel values from the image. It used the final state of the RNN's hidden layer to classify a sequence of time steps into a digit class label.

Sequence of inputs are $\{x_1, x_2, \ldots, x_T\}$ where T is the sequence length, f_θ is RNN model with parameters θ(weights and biases), number of hidden units H and its sequence of outputs will be $Y=\{y_1,y_2,\ldots,y_T\}$. Its steps are as follows.

Step 1. Initialize Model Parameters:

- Initialize weight matrices W_h, U_h, W_y and bias vectors b_h, b_y randomly.
- Initialize the hidden state h_0 to zeros or a small random value.

Step 2. Process the Sequence:

- For each time step t =1,2,…,T:

Compute Hidden State: Update the hidden state using the current input x_t and the previous hidden state h_{t-1}

$$h_t\ \ = (\,W_h\,x_t\ +\ U_h h_{t-1}\ \ + b_h) \tag{6}$$

- *Compute Output:* Compute the output for the current time step:

$$y_t = \mathrm{softmax}\,(W_y h_t\ \ + b_h) \tag{7}$$

Step 3. Store Outputs: Save y_t for each time step.
Step 4. Train the RNN (if training):

- Compute the loss over the sequence (e.g., cross-entropy loss for classification).
- Perform backpropagation through time (BPTT) to update model parameters θ using gradient descent.

Step 5. Return Outputs*:* Return the sequence of outputs $Y = \{y_1,y_2,\ldots,y_T\}$

Optimizer: Adaptive Moment Estimation (Adam), is one of the best choices for resolving issues linked with Trainings of (NN) and improving Convergence Performance as well as easing the hyperparameter tuning difficulty associated with complicated and Diverse Models - Neural Networks (NN), by minimising or eliminating the need to perform manual learning rate tuning. Adam takes advantage of momentum, Adaptive learning rates and also demonstrates a sense of resilient to sparse, Noisy gradients. These combined advantages result in improved performance, Convergence, and lower complexity with respect to hyperparameter tuning, when utilising Adam within a wide range of NN-Based Applications.

3.3. Original Model Performance

The performance of the neural networks trained with MNIST data was evaluated via evaluation metrics that assess

the overall model's classification capabilities using traditional performance metrics such as accuracy, precision, recall, F1-score, and AUC ROC curve. For quantitative performance statistics for the CNN networks, please see Table 2, for DNN networks refer to Table 3 and finally for recurrent networks tables 4 & 10. In addition to the tables, you can see the AUC values calculated using the ROC Curve shown in Fig. 9 (a). In this figure, we compare the two ROC curves between our unmodified MNIST training set (baseline) and those obtained by our networks after they were modified by the adversarial example generation. You can also view examples of adversarial outputs generated by gradient-based white-box attacks against the MNIST dataset for CNN networks in Fig. 8.

3.4. Adversarial Attack Model

The generation of adversarial examples consists of making small perturbations to input data to trick neural networks to predict incorrectly. All methods used to create adversarial examples are classified as white-box attacks, which rely on the use of gradients.

The Fast Gradient Sign Method (FGSM) is one of the earliest and simplest approaches to generating adversarial examples. This method utilizes a small perturbation (in this case epsilon set at 0.1) in the direction of the loss function's gradient to create a maximum prediction error for the model. FGSM can be computed easily, and many more advanced adversarial techniques have evolved from this initial idea. Projected Gradient Descent (PGD) is a method that generates adversarial images with many iterations and which is very similar to FGSM. Like PGD, FGSM produces very small perturbations (in a fixed sized range of input values) that create adversary images of a specified model. The Deep Fool (DF) method produces the most minimal changes to images so that the classifier would incorrectly identify them as being another category. In contrast to the larger perturbations created by other gradient-based attacks or FGSM, Deep Fool strive to produce the smallest amount of change required for crossing into another category. This creates an excellent baseline for measuring robustness in deep neural networks.

Severity of the Attack: When the CNN model was tested on the MNIST dataset that had been manipulated, the results are in Table 2. The ROC AUC scores of the images used in the experiment and their associated ROC graphs (shown in Fig. 9) demonstrate inferior classification ability. Fig. 9 shows a graphical comparison between the models used for generating adversary images versus models that had been subjected to this attack. A summary of the performance results of the DNN and RNN against adversaries generated in Tables 3, 4, and 10 is presented along with the ROC graphs in Figures 12 and 16, respectively.

3.5. Defense Methods (DM)

In the work presented here are five defensive strategies that have been implemented and discussed, including "Adversarial Training," "Defensive Distillation," "Autoencoders," "Ensemble Methods" and a technique known as "Feature Squeezing," that were used independently against different forms of adversarial attacks such as PGD, DeepFool attack and FGSM attacks.

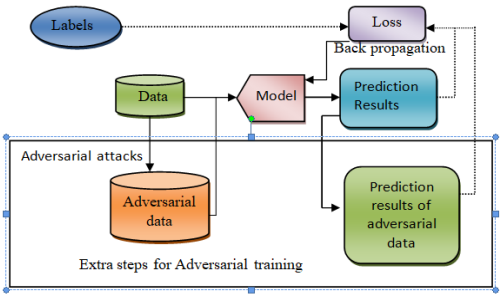


Fig.3. Adversarial training defense method model

A. Adversarial Training (AT)

The adversarial training framework is shown in Figure 3, which illustrates how clean input X is transformed through various forms of adversarial attack (i.e., FGSM, PGD, and DeepFool) into an adversarial sample. This model is retrained using both clean and adversarial data in order to develop an improved capacity for generalisation under perturbations, while still being able to maintain its accuracy for "benign" clean inputs [39–43]. During the training of the model, predicted outputs from the model are compared to the true label and the amount of loss calculated, and backpropagation then occurs to adjust the parameters of the model. This adversarial pipeline tests the performance of the model on each of the perturbed test inputs and includes the adversarial predictions of the perturbed inputs in the retraining of the model. The combination of these adversarial generation and retraining phases, constitute the primary means by which the overall robustness and classification confidence of the model is enhanced in the presence of an adversarial attack.

B. Defensive Distillation (DD)

Using defensive distillation, you can train your model to yield soft probabilities (or soft labels) instead of simply

providing hard classification labels. In this technique, the model consists of two separate stages: In the first stage, a teacher network is created, which produces softened, high temperature logit outputs against which the student model will be trained. Because they contain greater amounts of information regarding the relationship between classes, the condensed logits provide better constructed decision boundaries and are less sensitive to small perturbations in input data than their original counterparts. To optimize the student model, the student loss must be minimized in relation to the outputs generated by the teacher network; this optimizes the influence of noise present in adversarial examples on the correct classifications made by the distilled student model. This allows the distilled student network to be much more robust and can be used for both clean and adversarial examples.

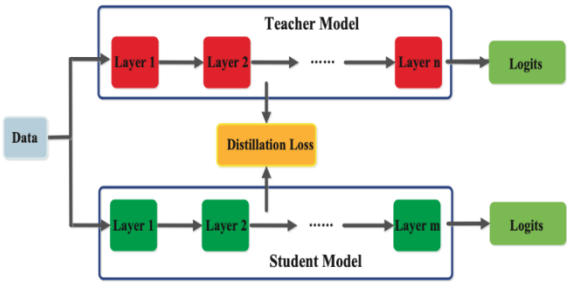


Fig.4. Defensive distillation defense method model

C. Ensemble Methods (EM)

The use of ensemble methods has been shown to add robustness through combining multiple model predictions, providing that the mis-influence of one model does not have a detrimental effect on the overall outcome. A great amount of the diversity in this study is created through training multiple models based upon different hyperparameters, architectures, and/or subsets of the training set and aggregating the model outputs using voting, averaging or weighted rules. Furthermore, using this type of aggregation in the ensemble methods decreases each classifier's individual vulnerability and makes it more difficult for an attacker to simultaneously deceive all classifiers. In this study, a large number of RNNs with LSTM and GRU layers have been incorporated into an ensemble to create various decision boundaries [45]. As can be seen in Fig. 5, the final prediction is derived from aggregating the independent predictions made by the models, which provides increased robustness compared to a single classifier with respect to the effects of adversarial perturbations.

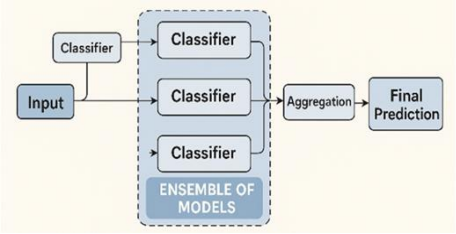


Fig.5. Ensemble defense method model

D. Autoencoders as Defense (AD)

Using denoised input before classification is how autoencoders reduce the chance of adversarial examples being misclassified [46–47]. Autoencoders consist of two parts; an encoder and a decoder. The encoder encodes the input into a latent representation, while the decoder decodes this representation back to the original input format (if possible). The latent space is designed to allow for capturing the underlying regularities associated with clean data. Therefore, high-frequency noise associated with the adversarial inputs will be removed during the reconstruction process. The MNIST example shows how one original image X was transformed into latent representation Z, then reconstructed as image X'. The reconstructed image should not only be similar to the original but also retain information relevant to classification, i.e., denoised enough to allow for better classification robustness in future use cases.

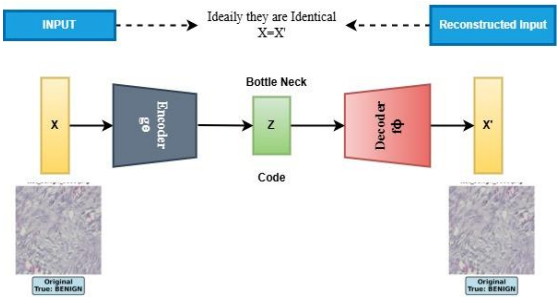


Fig.6. Autoencoder defense model

E. Feature Squeezing (FS)

The process of feature squeezing reduces how much effect an adversary can have by decreasing the amount of complexity involved in the input and increasing its compression. This defense assumes that adversarial images are comprised of very small and finely-tuned perturbations (changes) to normal images; if these small changes can be eliminated, either through quantization or smoothing (for example), then the adversarial image no longer exists. For instance, shown in Figure 7, when an MNIST adversarial image is processed through a feature squeezer, it undergoes a reduction in degrees of freedom that is related to the accuracy of the classification model. Different types of common squeeze operations include bit depth reduction, smoothing, and median filtering to remove adversarial noise. The classification of the denoised image leads to increased stability in predicting labels because the input is represented more consistently than an example processed through an adversary's local perturbation (as depicted by the thick black line in the image).

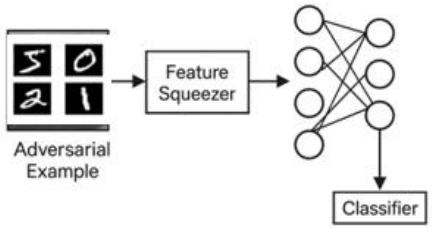


Fig.7. Feature squeezing defense method model

Each of these defensive strategies has a different way to avoid enemy attacks:

- Adversarial training using adversarial cases makes the model stronger.
- Defensive distillation tries to make it harder to tell the difference between choices.
- Ensemble methods use more than one model to make sure predictions are correct.
- Autoencoders clean up inputs before they are used to get rid of any adversarial noise.
- Squeezing Features: This method makes it less sensitive to small changes in the input.

Every method has its pros and cons, so picking one usually mean thinking about the application's needs and the threat model it is likely to face.

F. Defense Model Performance

Performance of CNNs, DNNs and RNNs under baseline and adversarial attack conditions, as seen in Tables 2, 3, & 13. Tables 5-9 provide a summary of the effectiveness of each of the defense methods on RNNs when they have been attacked, through their respective metrics. These visual representations include metric-based comparison of models and defenses on attacks, overall performance assessment of the models through Figures 10 (a-e), 13 (a-e), 17 - 21 and 22-25.

G. Detection Model

The performance of NN models on defense strategies to combat adversarial attacks, as measured by ML evaluation metrics, is illustrated in Results tables 2, 3, and 10. The degree of difference in predictive capability of the NN model prior to and following the attack indicates a strong possibility of adversarial input(s) used on the trained NN model.

3.6. Performance Evaluation Matrices

The established evaluation criteria were employed to measure and analyze the suggested robust neural network structures, adversarial attack models and defense models using the MNIST dataset. The performance was determined through the components of the confusion-matrix (TP, FP, FN, TN) which are the conventional metrics in classification

tasks. These parameters provide direct insights into the models' classification accuracy on clean, attacked, and defended data.

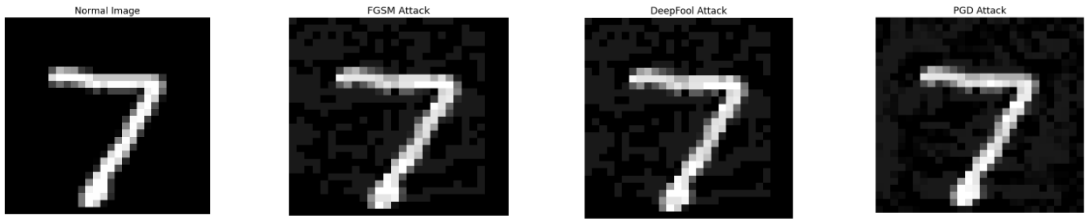
Table 1. Evaluation metrics of ML models

Metric	Formula	Description	Usage / Applicability
[50] Precision	Precision= $\frac{TP+FP}{TP}$	Percentage of correctly predicted positive instances based on all instances that were predicted as positive by the system.	Most important when a false positive is expensive.
[49] Accuracy	Accuracy= $\frac{TP+FP+FN+TN}{TP+TN}$	Percentage of total correct predictions based on all examples in the database.	Most relevant when the distribution of classes is relatively even.
[50] Recall (Sensitivity)	Recall= $\frac{TP+FN}{TP}$	Percentage of correctly identified actual positives by the predictive model.	Most relevant when a false negative has a large cost.
F1-Score [51]	F1=2· $\frac{Precision+Recal}{Precision\cdot Recall}$	The harmonic mean of precision and recall, providing a balance between both quantities.	A very relevant measure with respect to imbalanced datasets.
[50–51]ROC–AUC (Area Under the ROC Curve)	Area under the ROC curve.	Captures the trade-off between the true positive rate (TPR) and the false positive rate (FPR) across different threshold settings.	Provides threshold-tuning ability for probabilistic classifiers.

4. Findings and Discussion

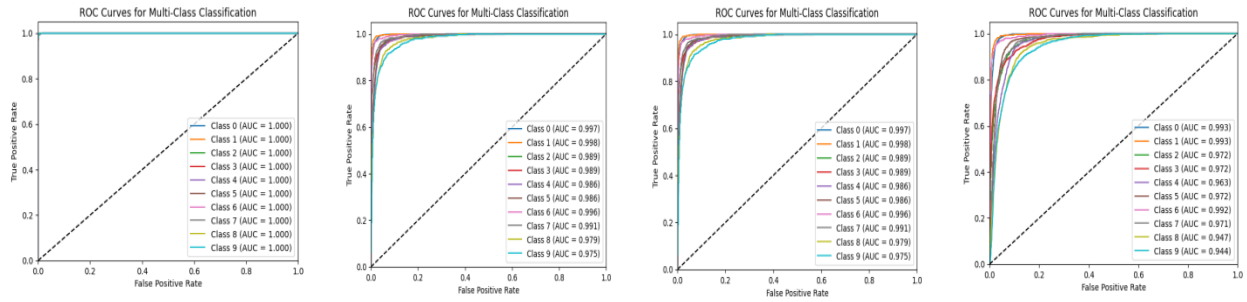
4.1. Evaluation of CNN Model Attack and Defense Strategies

Fig. 8 demonstrates sample predictions of the CNN classification model on pristine MNIST images and relevant adversarial samples created by gradient-based white-box attacks, including FGSM, DeepFool, and PGD.



(a) (b) (c) (d)

Fig.8. CNN model sample output: (a) Normal image, (b) FGSM attacked image, (c) DeepFool attacked image, (d) PGD attacked image



(a) (b) (c) (d)

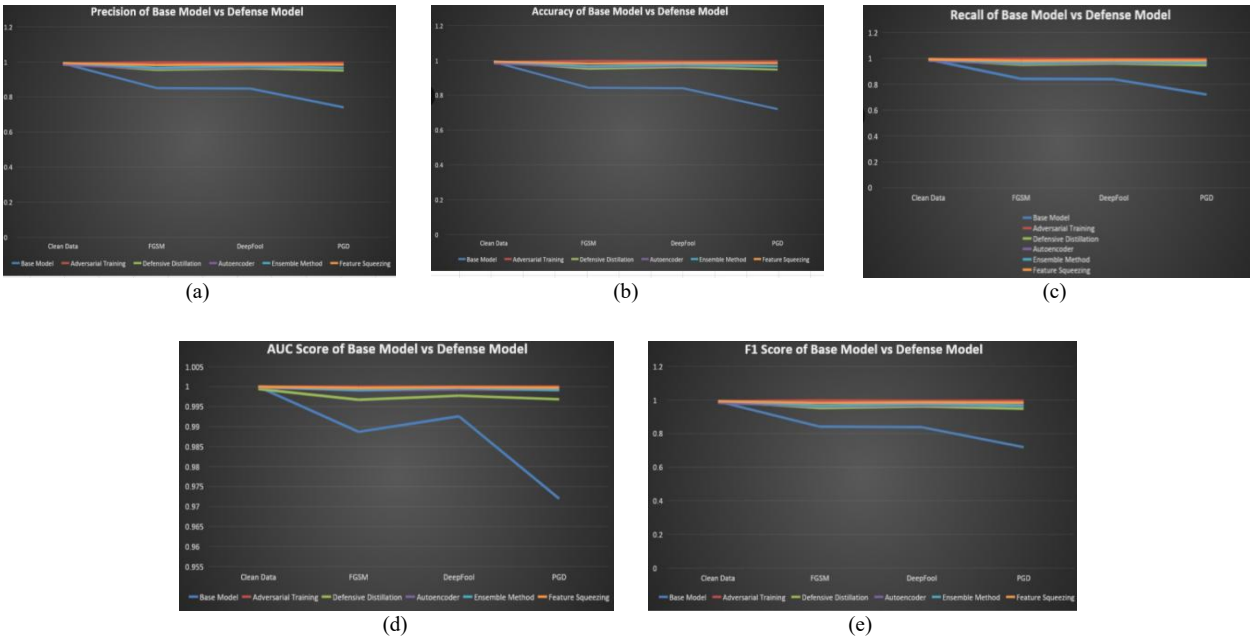
Fig.9. CNN model ROC curves and AUC scores: (a) Base model, (b) FGSM attack, (c) DeepFool attack, (d) PGD attack

Figures 9(a)-(d) illustrate ROC curves for multi-class classification using CNN under clean and adversarial conditions. Fig. 9(a) shows perfect classification for the clean MNIST dataset, with each class achieving an AUC of 1.000, indicating an ideal decision boundary and no confusion. Fig. 9 (b) presents the ROC curves under FGSM attack, where the model performance degrades slightly, but still maintains high AUC values between 0.960 and 0.987 for most classes. Fig. 9(c) displays the results under the DeepFool attack, where AUC values range from 0.9191 to 0.9731, indicating more noticeable misclassification across classes. Fig. 9(d) shows CNN under the PGD attack, with further reduced AUCs ranging from 0.8441 to 0.9993, revealing substantial vulnerability of CNN to iterative adversarial perturbations. The declining trend across Figures 9 (a) - (d) underscores how adversarial attacks progressively reduce the model's class-wise separability. Although the CNN performs well on clean data, its performance degrades under PGD attacks, particularly for some classes. The ROC analysis highlights the importance of effective defense strategies in maintaining model reliability against adversarial attacks.

Table 2. Performance of the CNN model with defense mechanisms against adversarial attacks, evaluated using standard classification metrics

S.No	Model	Evaluation Type	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Base Model	Clean Data	0.9915	0.991475042	0.991372512	0.999959029	0.99393127
2	Base Model	FGSM	0.8437	0.850972263	0.841901364	0.988703899	0.841404244
3	Base Model	DeepFool	0.8407	0.847920989	0.838914576	0.99259089	0.838263171
4	Base Model	PGD	0.7206	0.740916627	0.720594012	0.971975036	0.71921246
5	Adversarial Training	Clean Data	0.9913	0.991232207	0.991213589	0.999909096	0.99121707
6	Adversarial Training	FGSM	0.9975	0.997456609	0.997501286	0.999992403	0.997476575
7	Adversarial Training	DeepFool	0.9942	0.994170786	0.994129401	0.999973371	0.994145784
8	Adversarial Training	PGD	0.995	0.994981875	0.995007361	0.999970513	0.994987303
9	Defensive Distillation	Clean Data	0.9918	0.991738099	0.991712628	0.99941547	0.991712545
10	Defensive Distillation	FGSM	0.9521	0.953955727	0.951951161	0.996724381	0.952108205
11	Defensive Distillation	DeepFool	0.9614	0.963046852	0.961257434	0.997742004	0.96143598
12	Defensive Distillation	PGD	0.948	0.950485304	0.947837689	0.996832325	0.948103869
13	Autoencoder	Clean Data	0.9833	0.983495007	0.983048809	0.999822784	0.983190714
14	Autoencoder	FGSM	0.9622	0.962568857	0.961609408	0.999041344	0.961785423
15	Autoencoder	DeepFool	0.9686	0.968839506	0.968077743	0.999343106	0.968225662
16	Autoencoder	PGD	0.9661	0.966308656	0.965550198	0.999185093	0.965669997
17	Ensemble Method	Clean Data	0.9939	0.993872865	0.99383606	0.999978759	0.993848681
18	Ensemble Method	FGSM	0.9664	0.966272803	0.966070545	0.99921135	0.966076897
19	Ensemble Method	DeepFool	0.977	0.976922193	0.9767775197	0.999721614	0.97677358
20	Ensemble Method	PGD	0.9666	0.966431326	0.96624718	0.999149704	0.966271692
21	Feature Squeezing	Clean Data	0.9928	0.992812107	0.992764147	0.999952959	0.992784761
22	Feature Squeezing	FGSM	0.9824	0.98240522	0.982308446	0.999694492	0.98234185
23	Feature Squeezing	DeepFool	0.9863	0.986285413	0.986247693	0.999855842	0.986258333
24	Feature Squeezing	PGD	0.9842	0.984181633	0.984146823	0.999744606	0.984149772

Table 2 presents the performance metrics of the CNN model on MNIST under clean and adversarial conditions, with and without defenses. The base model performs excellently on clean data (accuracy: 0.9915, AUC: 0.9999), but shows sharp degradation under PGD (accuracy: 0.7206, AUC: 0.9791). Among defenses, Adversarial Training consistently yields the highest recovery, with F1 scores above 0.994 across all attacks. Defensive Distillation, Autoencoder, and Ensemble Methods also significantly improve robustness, while Feature Squeezing shows moderate effectiveness, especially under FGSM.



(a) (b) (c)

(d) (e)

Fig.10. DNN base model vs. defense model performance against adversarial attacks: (a) Accuracy, (b) Precision, (c) Recall, (d) AUC score, (e) F1 score

The results shown in Figure 10 compare the performance of the CNN model trained on the MNIST dataset under various adversarial attacks (FGSM, DeepFool, and PGD) and defense measures (Gordon, 2019). Even when trained using only the original image data, the performance metrics of the CNN model are very good, with the base CNN model achieving high values for precision, accuracy, recall, AUC and F1 Score (all are close to 0.99), indicating nearly complete classification. However, under adversarial attacks, especially PGD, all performance metrics of the base model decline sharply, with the F1 score and AUC dropping below 0.85. Among the defenses, Adversarial Training and Defensive Distillation most effectively preserve model performance across all attacks. Autoencoder and Ensemble Method also show strong resilience, maintaining scores above 0.95 for most metrics, even under PGD. Feature Squeezing, while helpful against FGSM, performs weakest overall, especially under PGD, where metrics drop significantly. The gap between the base model (blue line) and the defense strategies (colored lines) widens as attack intensity increases, highlighting the importance of complementary defense mechanisms.

Overall, the graphs demonstrate that robust defense mechanisms significantly recover CNN performance under adversarial perturbations, with Adversarial Training performing most consistently.

4.2. Evaluation of DNN Attack and Defense strategies

Fig. 11 displays the DNN classification model sample output of normal images from the MNIST dataset, along with gradient-based white-box attacked images, such as FGSM, DeepFool, and PGD.

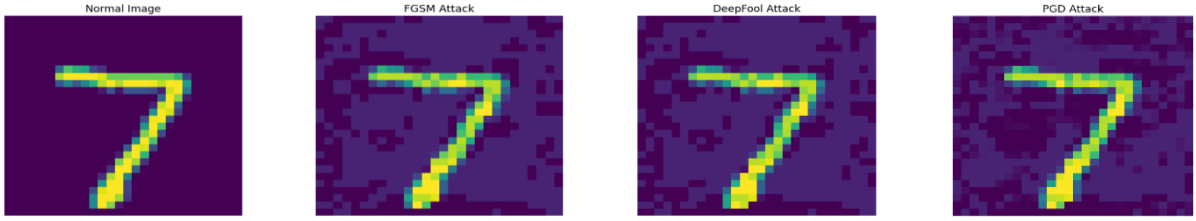


Fig.11. DNN model sample output images: (a) Normal image, (b) FGSM attacked image, (c) DeepFool attacked image, (d) PGD attacked image

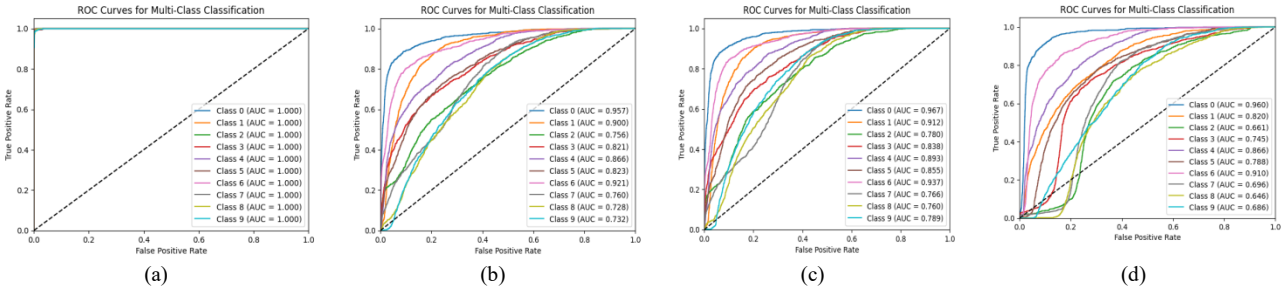


Fig.12. DNN model ROC Curve performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

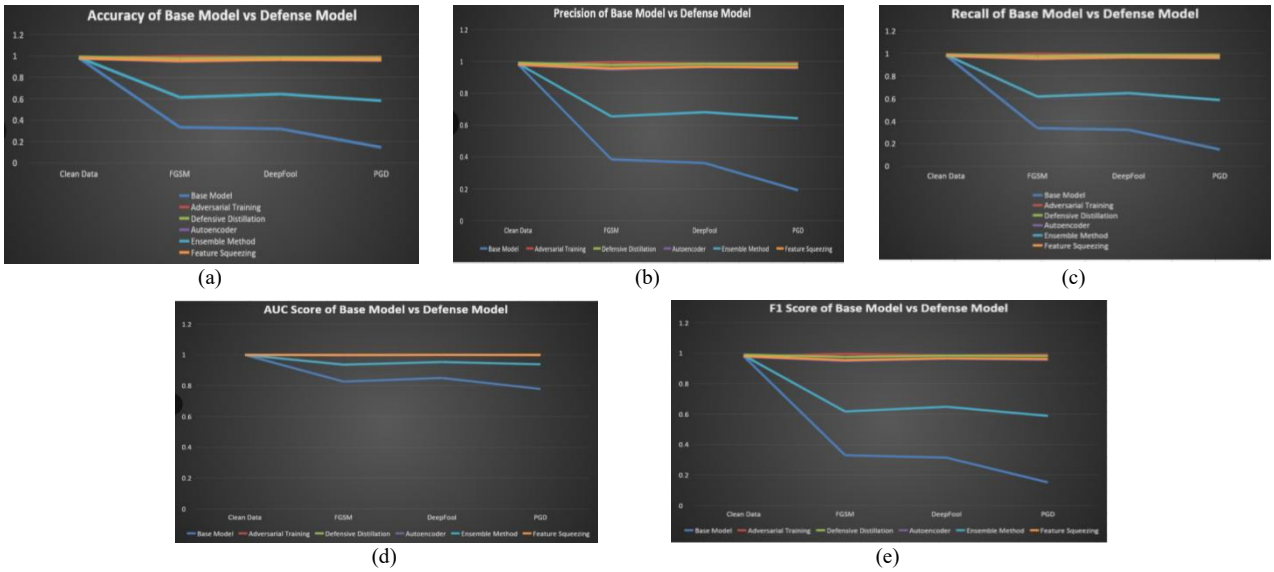
Table 3 shows that white-box attacks such as FGSM, DeepFool, and PGD significantly degrade the AUC performance of the base model, which in this case is a DNN trained on the MNIST dataset. Under clean conditions, the DNN achieves an AUC score of approximately 0.9998, confirming high separability between classes and reliable classification capability. However, the AUC drops significantly under adversarial attacks, with FGSM and DeepFool reducing it to 0.989 and 0.991, respectively. The PGD attack causes the largest AUC drop, down to 0.973, reflecting increased uncertainty in the model's predictions. Even when there are strong attacks, defense strategies like Adversarial Training and Defensive Distillation can bring AUC performance back to almost clean levels. Autoencoder and Ensemble Methods also work well for AUC recovery, especially when DeepFool is used. Feature Squeezing, on the other hand, has a lower AUC performance under PGD but is still moderately resistant to FGSM. Figure 12 clearly shows this AUC trend for all tested models, underscoring the importance of defense mechanisms in maintaining model robustness under attack.

Table 3. Performance of the DNN model with defense mechanisms under adversarial attacks, evaluated using standard classification metrics

S.No	Model	Evaluation Type	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Base Model	Clean Data	0.9811	0.981316761	0.980769405	0.999712173	0.980965125
2	Base Model	FGSM	0.3346	0.38502493	0.338166137	0.826478178	0.330130243
3	Base Model	DeepFool	0.3195	0.361637469	0.323043288	0.849705613	0.314265603
4	Base Model	PGD	0.147	0.191037442	0.148892337	0.777836926	0.151846934
5	Adversarial Training	Clean Data	0.9776	0.977933817	0.977193258	0.999455081	0.977439857

6	Adversarial Training	FGSM	0.996	0.996022284	0.995898794	0.999959546	0.995956972
7	Adversarial Training	DeepFool	0.9853	0.985452831	0.985033728	0.999824517	0.985210641
8	Adversarial Training	PGD	0.9898	0.989846746	0.989586847	0.999893289	0.989700882
9	Defensive Distillation	Clean Data	0.9898	0.99002214	0.989661679	0.999659766	0.989800625
10	Defensive Distillation	FGSM	0.9761	0.976485802	0.975741916	0.998697271	0.976023704
11	Defensive Distillation	DeepFool	0.9843	0.984521747	0.984110262	0.999480987	0.984262804
12	Defensive Distillation	PGD	0.9818	0.982185199	0.981505607	0.999097888	0.981774338
13	Autoencoder	Clean Data	0.9774	0.977505898	0.977030842	0.999557531	0.977211713
14	Autoencoder	FGSM	0.9495	0.949725492	0.948723587	0.997931766	0.948986488
15	Autoencoder	DeepFool	0.9654	0.965480567	0.964889275	0.999067002	0.965077463
16	Autoencoder	PGD	0.9578	0.957980666	0.957111894	0.998550558	0.957365185
17	Ensemble Method	Clean Data	0.986	0.986006613	0.985834685	0.999817245	0.985913489
18	Ensemble Method	FGSM	0.6138	0.654475638	0.617126936	0.936021478	0.617520978
19	Ensemble Method	DeepFool	0.6441	0.680689784	0.647510889	0.953269729	0.648105012
20	Ensemble Method	PGD	0.5822	0.642217146	0.586783371	0.938529113	0.588603363
21	Feature Squeezing	Clean Data	0.9803	0.980511496	0.980028827	0.999634389	0.980209318
22	Feature Squeezing	FGSM	0.9557	0.955864829	0.955206202	0.998161457	0.955329598
23	Feature Squeezing	DeepFool	0.9666	0.966844569	0.96625675	0.999155933	0.966355663
24	Feature Squeezing	PGD	0.9631	0.963188938	0.96270207	0.998775127	0.962778651

Table 3 shows that the DNN model performs excellently on clean MNIST data, with an accuracy of 0.9811 and AUC of 0.9997, but suffers major degradation under adversarial attacks. Under PGD, the model's accuracy drops to 0.147 and AUC plummets to 0.7778, indicating poor classification capability. Among defense strategies, Adversarial Training shows the best robustness, recovering AUC to 0.9998 under PGD, followed closely by Defensive Distillation and Autoencoder, both maintaining high scores across all attacks. Feature Squeezing offers moderate protection, while Ensemble Method fails to retain performance under strong attacks like PGD and DeepFool.



(a) (b) (c)

(d) (e)

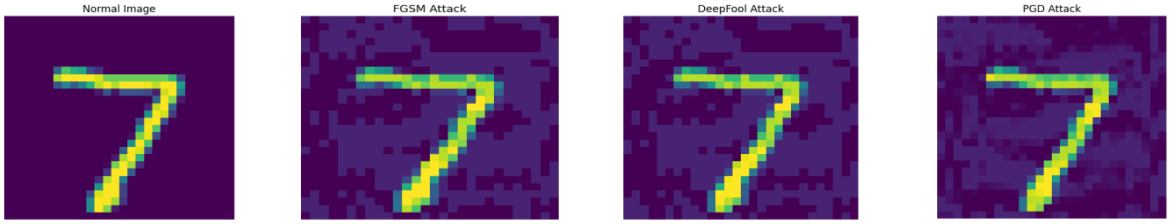
Fig.13. Performance comparison of the DNN base model and defended model under adversarial attacks: (a) Accuracy, (b) Precision, (c) Recall, (d) AUC score, and (e) F1 score

Figure 13 shows the performance of the DNN model on the MNIST dataset under clean conditions, adversarial attacks (FGSM, DeepFool, PGD), and five different defense strategies. The base model does very well on clean data, with all of its metrics close to or above 0.98. However, when PGD is used, its performance drops to 0.147 and AUC to 0.7778. FGSM and DeepFool attacks also make the DNN's precision, recall, and F1 score much worse, which shows how weak it is. Adversarial Training consistently provides the best protection, maintaining accuracy and AUC above 0.98 across all attacks. Defensive Distillation and Autoencoder also effectively prevent degradation, maintaining metrics in the 0.96–0.98 range. Feature Squeezing works moderately well under FGSM but not so well under PGD, as shown by the F1 and AUC scores going down. The Ensemble Method, on the other hand, shows sharp drops under all three attacks, especially PGD. This indicates that the Ensemble Method is ineffective as a defense in this case. These plots show that DNNs work very well on clean data, but strong defense strategies, especially Adversarial Training, are

necessary to keep classification performance high in adversarial situations.

4.3. Evaluation of RNN Attack and Defense strategies

Fig. 14 displays the RNN classification model sample output of normal images from the MNIST dataset, along with gradient-based white-box attacked images, such as FGSM, DeepFool, and PGD.



(a) (b) (c) (d)

Fig.14. RNN model sample output images: (a) Normal image, (b) FGSM attacked image, (c) DeepFool attacked image, (d) PGD attacked image

In Table 4, an evaluation of both performance and resilience of a recurrent neuron network (RNN) model subjected to various types of adversarial attacks is presented. Such attacks include white-box (i.e., access to all internal model parameters) gradient-based attack methods, namely FGSM, deep fool and PGD, utilizing the MNIST dataset processed for RNN input.

Table 4. Performance and resilience of RNN model against base model (BM) and gradient-based white-box attacks such as FGSM, DeepFool, and PGD attacks on the MNIST dataset

S.NO	Model	Evaluation Type	Accuracy	Precision	Recall	AUC Score	F1 Score
0	Base Model	Base Model	0.9747	0.974583654	0.974679323	0.999320202	0.974507649
1	Base Model	FGSM	0.36	0.408796215	0.363737372	0.795277354	0.3566102
2	Base Model	DeepFool	0.3654	0.412083164	0.369337241	0.851323521	0.362333474
3	Base Model	PGD	0.0777	0.065870469	0.079225377	0.599602498	0.068196394

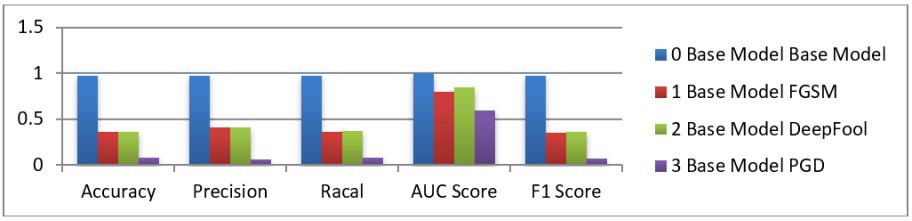
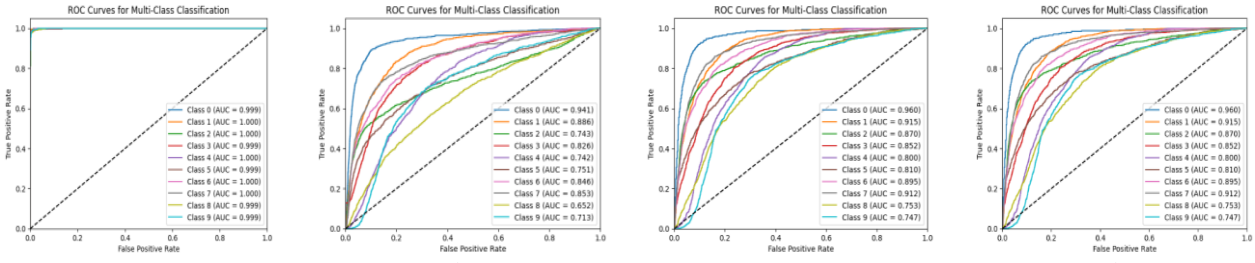


Fig.15. Visualization of performance degradation comparing the base model and adversarial attack models



(a) (b) (c) (d)

Fig.16. RNN model ROC Curve performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

As depicted in Table 4, the base model (an RNN trained on the MNIST dataset) is relatively easy to attack using gradient-based white-box attack methods (FGSM, DeepFool, PGD). The base model achieves high accuracy (0.9747, 0.9746, 0.9747), AUC (0.9993), and F1 Score (0.9745) under no perturbation, thereby demonstrating robust classification performance. In contrast, with FGSM, the performances drop dramatically with accuracy (0.36), precision (0.4088), and recall (0.3637). The AUC value decreased drastically to 0.7953, while the F1 score decreased to 0.3566, demonstrating the negative impact of this attack type on classification performance. When using DeepFool, the results were somewhat better than FGSM with accuracy (0.3654), precision (0.4121), and recall (0.3693) all increasing slightly over those reported for FGSM, although still lower than the base model's performance. The area under the curve (AUC) increased from 0.8513 to 0.8519, while the F1 score decreased significantly from 0.3622 to 0.0617. The PGD attack

reduced the model's performance by reducing the accuracy to 0.0777, precision to 0.0659, and recall (sensitivity) to 0.0792. The AUC Score fell to 0.5996, while the F1 score was just 0.0682. As such, the model is virtually unable to provide meaningful classifications in the presence of this type of attack.

The observations presented in Figures 12-14 demonstrate that RNN models are vulnerable to these types of attacks, indicating a need for improved defensive measures. The visuals in Figure 16(b)-(d) show the performance degradation of the base model under various attacks.Figure 16(a) shows the ROC curves for the RNN model on the clean MNIST dataset, with an AUC score of 99.9%. This high AUC score indicates superior class separability compared to other metrics, as detailed in Table 4.

A. Evaluation of Defense Mechanisms for Adversarial Attacks

Adversarial training has been evaluated against existing baseline models in terms of both the performance and robustness of its results when examined under various forms of attack, such as FGSM attacks, DeepFool (DF) and PGD attacks.

Table 5. Performance and resilience of adversarial training against base model and adversarial attacks

Serial Number	Defensive Strategy	Model	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Adversarial Training	Clean Data	0.9665	0.9667	0.9661	0.9661	0.9663
2		FGSM	0.9922	0.9923	0.9921	0.9999	0.9922
3		DeepFool	0.9772	0.9773	0.9770	0.9996	0.9771
4		PGD	0.9696	0.9700	0.9693	0.9995	0.9694

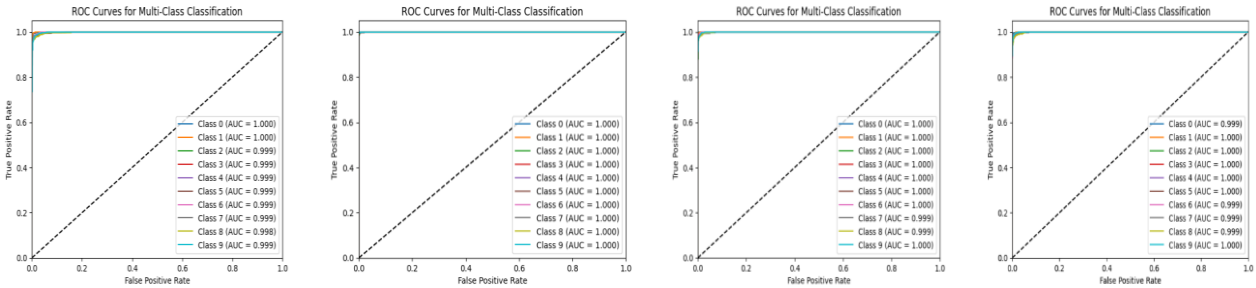


Fig.17. RNN model ROC Curve for AT performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

Table 5 shows the adversarial training of RNN model performances against gradient-based adversarial attacks. Figure 17 illustrates the AUC scores of ROC curves for multi-class classification for the base model and all adversarial attacks. Table 5 presents the AUC scores, which demonstrate the best performance among the evaluated metrics for both base and attacked models.

The summary of the efficacy of the base model compared to how effective adversary training is under several attacks is shown in Table 5. With regard to overall performance, the base model had an overall performance level that was moderate, with relatively high values for accuracy, precision, recall, AUC, and F1-scores averaging around 0.96. The model trained on adversarial examples showed only a negligible decrease in performance from what it would have achieved using clean data while still demonstrating high levels of efficacy. The adversarial training model demonstrated a high degree of robustness under FGSM testing. The model generated an F1 score and accuracy of approximately 0.992 and AUC of almost 1.0, thus representing an impressive improvement over the base model. The performance with respect to DeepFool was slightly less than with clean data, but overall remained high, with an accuracy of nearly 0.977 and an AUC of approximately 0.9996. With regard to PGD testing, the adversarial training approach provided the highest capability of mitigating this attack type, with an accuracy and precision of approximately 0.97 and an AUC of nearly 0.9995. In general, it can be concluded that adversarial training improved the model's ability to resist attacks of all types, specifically demonstrating a high capacity for resistance against FGSM-type attacks and a good level of mitigation capability for PGD and DeepFool attack types.

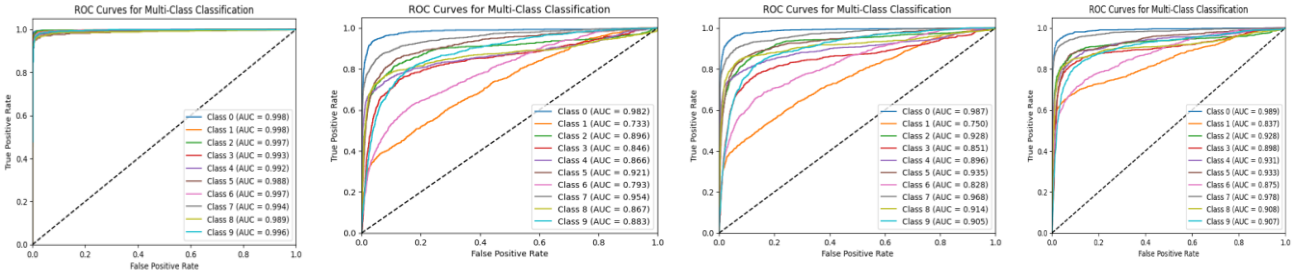
B. Defensive Distillation Performance and Resilience against the Base Model and Adversarial Attacks (FGSM, DeepFool, and PGD)

Table 6 and Figure 18(a–d) illustrate the performance of the RNN model using Defensive Distillation for both clean and adversarial examples with different performance metrics, including accuracy, precision, recall, F1-score, and ROC–AUC. The distilled model performed well on clean data with accuracy 0.9508 and AUC 0.9941. However, the distilled model was still vulnerable to adversarial attacks, especially the FGSM attack, which resulted in accuracy being reduced to 0.6074 and AUC being reduced to 0.8740. DeepFool and PGD attacks resulted in less degradation than FGSM but still led to significant reductions in accuracy (PGD attack = 0.7221, AUC = 0.9183). F1-scores exhibited a

similar downward trend under all attack conditions. The overall conclusion is that Defensive Distillation offers limited benefits for creating robust models, as demonstrated by the AUC being the most stable measure, while FGSM and DeepFool attacks highly impact performance.

Table 6. Performance and resilience of Defensive Distillation against base model and adversarial attacks

Serial Number	Defense Strategy	Model	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Defensive Distillation	Clean Data	0.9508	0.9510	0.9502	0.9941	0.9504
2		FGSM	0.6074	0.6658	0.6127	0.8740	0.5895
3		DeepFool	0.6529	0.6980	0.6578	0.8961	0.6396
4		PGD	0.7221	0.7525	0.7248	0.9183	0.7221



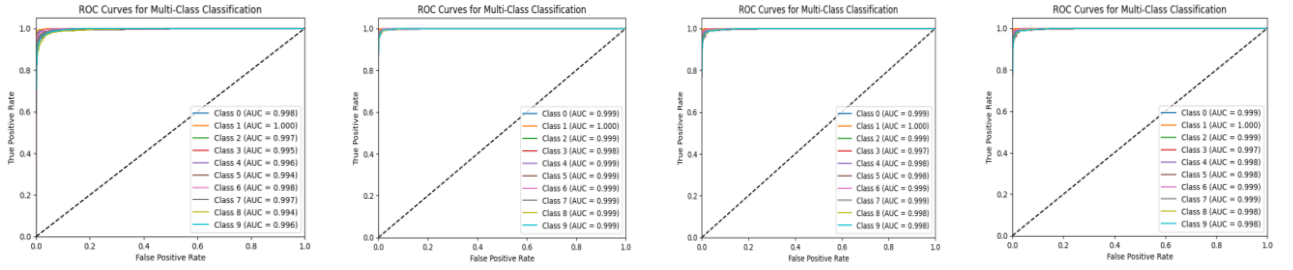
(a) (b) (c) (d)

Fig.18. RNN model ROC Curve for DD performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

C. Autoencoder Methods (AEM) Performance and Resilience against the Base Model and Adversarial Attacks (FGSM, DeepFool, and PGD)

Table 7. Performance and resilience of AEM against base model and adversarial attacks

Serial Number	Defense Strategy	Model	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Autoencoder Method	Clean Data	0.97	0.9698	0.9697	0.9990	0.9696
2		FGSM	0.9373	0.9373	0.9368	0.9964	0.9365
3		DeepFool	0.9592	0.9590	0.9588	0.9983	0.9586
4		PGD	0.9516	0.9514	0.9512	0.9977	0.9509



(a) (b) (c) (d)

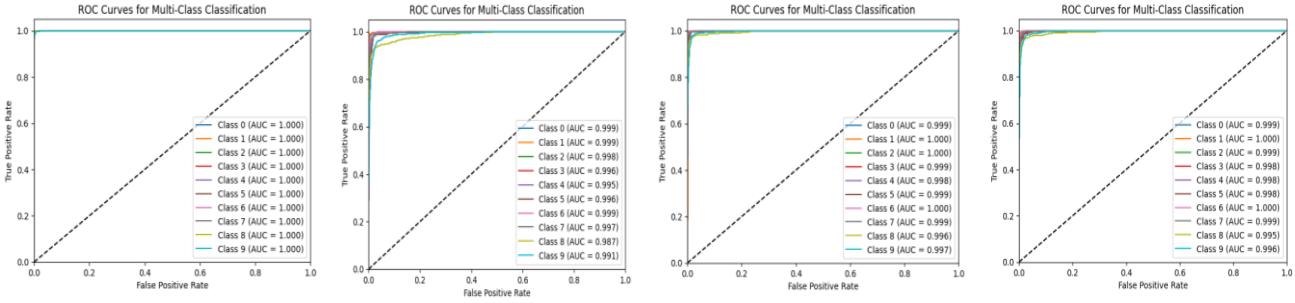
Fig.19. RNN model ROC Curve for AEM performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

According to Table 7, the autoencoder-based defense system had a good performance on the baseline data with an overall accuracy, precision, recall and F1 scores of nearly 0.97 and a maximum AUC of 0.9990 when trained on clean data. The application of adversarial examples generated by FGSM greatly reduced the model's performance of the model when using FGSM, with an overall accuracy and F1 score of approximately 0.94, but the AUC remained high at 0.9964. The second adversarial attack type of DeepFool provided a better level of resiliency to the model, achieving an overall accuracy and recall of approximately 0.959 with an AUC of 0.9983. The PGD attack type provided intermediate resiliency to the model with an overall accuracy of approximately 0.952 and an overall AUC of 0.9977. All of the ROC analysis shows consistently high AUC values across the three attack types, which indicates class separability still exists, even though overall classification accuracy was reduced. As a result, this study demonstrates that autoencoder-based models are partially resilient, but also require additional or combination of other types of defenses to better protect against adversarial attacks as shown in Figures 19 (a-d).

D. Ensemble Methods Performance and Resilience against the Base Model and Adversarial Attacks (FGSM, DeepFool, and PGD)

Table 8. Performance and resilience of EM against base model and adversarial attacks

Serial Number	Defense Strategy	Model	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Ensemble Method	Clean Data	0.9898	0.9897	0.9898	0.9999	0.9897
2		FGSM	0.8603	0.9007	0.8658	0.9956	0.8676
3		DeepFool	0.902	0.9307	0.9070	0.9985	0.9073
4		PGD	0.949	0.9496	0.9493	0.9981	0.9489



(a) (b) (c) (d)

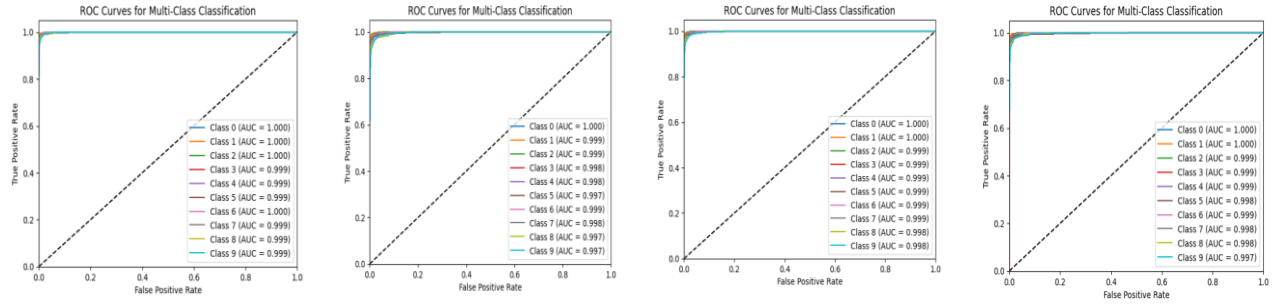
Fig.20. RNN model ROC Curve for EM performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

According to Table 8, the ensemble's performance has a strong baseline on the clean dataset, as shown by the ensemble's accuracy (approaching 0.99), precision (approximately 0.99), recall (approximately 0.99), and F1 score (approximately 0.99), as well as AUC (0.9999). As expected, when the ensemble is subjected to attacks, its performance across all of these metrics decreases significantly, with the highest decline in performance occurring due to FGSM, which caused the most significant decrease in accuracy (approximately 0.86), while still maintaining a relatively high AUC (0.9956). In contrast, the DeepFool attack caused a smaller decline in the performance of the ensemble; specifically, the ensemble's accuracy decreased to approximately 0.90, while AUC was still relatively high (0.9985). The PGD attack resulted in the least amount of decline in the ensemble's performance; however, the accuracy and recall results remained relatively consistent (approximately 0.949 for accuracy and recall) as well as AUC (approximately 0.998). In other words, while the accuracy decreases significantly when under attack, an ensemble approach to classification maintains a relatively high degree of class separation, as suggested by the consistently high AUC values. Therefore, the ensemble method has clear advantages when it comes to clean data and is relatively robust with respect to adversarial perturbations. The ROC–AUC curves illustrated in Fig. 20(a–d) further highlight the comparisons with respect to the various attack scenarios.

E. Feature Squeezing Methods (FSM) Performance and Resilience against the Base Model and Adversarial Attacks (FGSM, DeepFool, and PGD)

Table 9. Performance and resilience of FSM against base model and adversarial attacks

Serial Number	Defense Strategy	Model	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Feature Squeezing	Clean Data	0.9738	0.9737	0.9736	0.9993	0.9736
2		FGSM	0.9565	0.9563	0.9563	0.9981	0.9561
3		DeepFool	0.9659	0.9657	0.9657	0.9989	0.9656
4		PGD	0.9625	0.9623	0.9622	0.9986	0.9621



(a) (b) (c) (d)

Fig.21. RNN model ROC Curve for FS performance (a). Base Model (b) FGSM, (c) DeepFool, (d) PGD

4.4. Comparative Analysis of Defense Strategies for Recurrent Neural Networks against Adversarial Attacks

A. Performance and Resilience of Defensive Strategies against the Base Model and Adversarial Attacks

Table 10. Performance of the RNN model with defense mechanisms against adversarial attacks, evaluated using standard classification metrics.

S.No	Model	Evaluation Type	Accuracy	Precision	Recall	AUC Score	F1 Score
1	Base Model	Clean Data	0.9747	0.974583654	0.974679323	0.999320202	0.974507649
2	Base Model	FGSM	0.36	0.408796215	0.363737372	0.795277354	0.3566102
3	Base Model	DeepFool	0.3654	0.412083164	0.369337241	0.851323521	0.362333474
4	Base Model	PGD	0.0777	0.065870469	0.079225377	0.599602498	0.068196394
5	Adversarial Training	Clean Data	0.9665	0.966713733	0.966083987	0.99903555	0.966250592
6	Adversarial Training	FGSM	0.9922	0.992346691	0.992055316	0.999922789	0.992183275
7	Adversarial Training	DeepFool	0.9772	0.997295664	0.976982055	0.999618947	0.977074958
8	Adversarial Training	PGD	0.9696	0.970046855	0.969272653	0.99472822	0.969393516
9	Defensive Distillation	Clean Data	0.9508	0.951052819	0.950215226	0.994192796	0.950435111
10	Defensive Distillation	FGSM	0.6074	0.665803064	0.612769233	0.874099643	0.589534744
11	Defensive Distillation	DeepFool	0.6529	0.698068241	0.657878125	0.896180635	0.639619682
12	Defensive Distillation	PGD	0.7221	0.752595471	0.724834522	0.91836823	0.722102967
13	Autoencoder	Clean Data	0.97	0.969845925	0.969787635	0.999074293	0.969688023
14	Autoencoder	FGSM	0.9373	0.937631072	0.936824268	0.996452639	0.936580314
15	Autoencoder	DeepFool	0.9592	0.959028928	0.98816994	0.998360207	0.958689112
16	Autoencoder	PGD	0.9516	0.951477638	0.951200849	0.997724559	0.950976863
17	Ensemble Method	Clean Data	0.9898	0.989760922	0.989820236	0.999921743	0.989778318
18	Ensemble Method	FGSM	0.8603	0.90071384	0.865825403	0.995644736	0.8676258
19	Ensemble Method	DeepFool	0.902	0.930738738	0.907036354	0.99859483	0.907377607
20	Ensemble Method	PGD	0.949	0.949663934	0.949352827	0.998136422	0.948910143
21	Feature Squeezing	Clean Data	0.9738	0.973743559	0.973693011	0.999373434	0.973621056
22	Feature Squeezing	FGSM	0.9565	0.956302644	0.956341307	0.998166506	0.956166581
23	Feature Squeezing	DeepFool	0.9659	0.965786082	0.965757898	0.998974194	0.965635652
24	Feature Squeezing	PGD	0.9625	0.962378495	0.962270543	0.998633765	0.962188547

Table 10 compares different methods for defending RNN Models against various types of adversarial attacks. From the comparison, we see that the Baseline Model has good performance on clean data but has considerable decreases (e.g., FGSM accuracy ≈0.36) in performance due to FGSM, DeepFool, and PGD attacks. Adversarial Training is the most effective as it maintains very high (≈0.9922) FGSM accuracy along with good (≈0.9772) DeepFool performance. An Autoencoder-based method provides strong protection (accuracy >0.95) for all three attacks. Defensive Distillation is a method that improves on the Baseline but does so primarily against PGDs (accuracy ≈0.7221), while it is still impacted by stronger attacks. Ensemble Methods achieve excellent clean data performance (accuracy ≈0.9898) but inferior to adversarial training and have a much bigger drop in performance under all three attacks. Feature Squeezing is a robust defensive measure with good (accuracy >0.95) performance over all attacks; however, it is not as effective as Adversarial Training or Autoencoders.

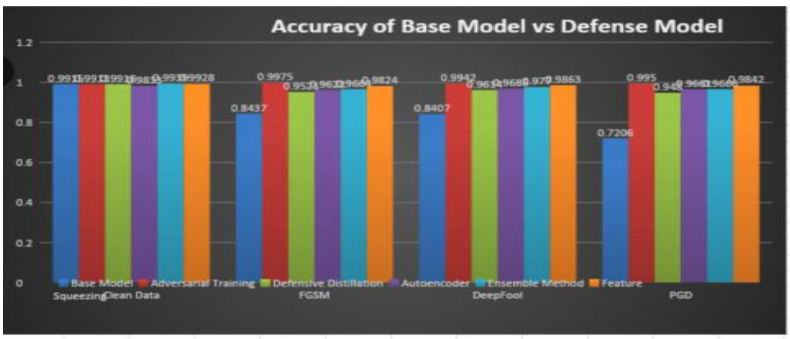


Fig.22. Comparing the accuracy of RNN baseline and defense models when they are attacked

The performance characteristics of various defense strategies against base model attacks are shown in Figure 22. The ensemble method exhibits the highest level of accuracy (0.9898) with respect to clean datasets while the accuracy

achieved by the base model alone is 0.9745. Adversarial training exhibited superior performance as a form of protection against attacks, particularly against FGSM (0.9922) and DeepFool attacks (0.9771) indicating that it is extremely resilient to such attacks. Although defensive distillation is a viable means of protecting against attacks; its capabilities are less than adversarial training when applied against PGD attacks (0.7221). The autoencoder also exhibited effectiveness against DeepFool (0.9587), and PGD (0.9510), and feature squeezing demonstrates the ability to defend against attacks but lacks the robustness of some of the above-mentioned techniques. Overall, adversarial training consistently shows the highest level of robustness to attacks; it performs significantly better against FGSM and DeepFool.

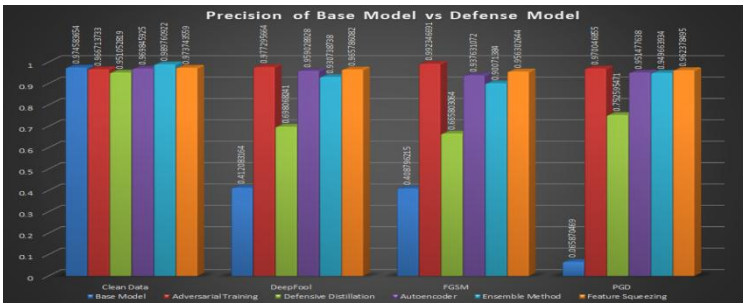


Fig.23. A precise comparison of the performance of baseline and defense models when they are under attack

The various defense strategies yield different effectiveness at repelling attacks against the base model, as shown in Fig. 23. Under clean data, the Ensemble Method has the highest total accuracy (0.9898), with Base Model coming in second place (0.9745). For defense using Adversarial Training, it is most effective in rejecting attacks with this technique (0.9922) and DeepFool attacks (0.9771). Defensive Distillation has a moderate performance but less than that of Adversarial Training, and is particularly poor against PGD attacks (0.7221). Autoencoder Methods also provide a strong performance against attacks but particularly with PGD Techniques (0.9509). Feature Squeezing has a moderate level of effectiveness but does not compare to Adversarial Training or the Ensemble Method when considering the overall robustness of the defense strategy. In summary, Adversarial Training should be considered the most reliable defense against all attack types.

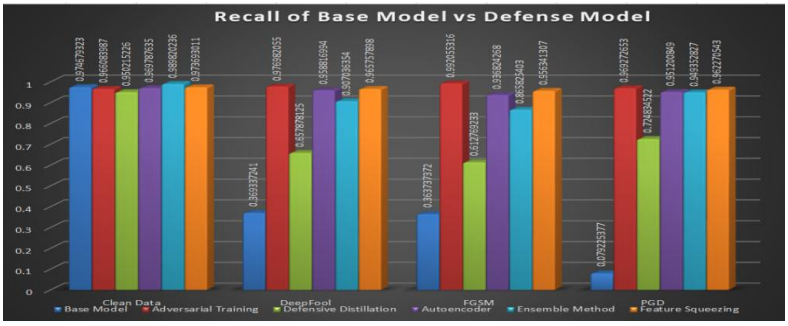


Fig.24. Comparative evaluation of recall for the base model and the defense model

Fig. 24 shows how various defense strategies were evaluated against attacks of various configurations on the base model. The ensemble method offers the highest accuracy on clean samples at 0.9898, while the base model yields 0.9747. Adversarially trained models have higher performance with respect to all attacks than any other methods tested, maintaining high accuracy against FGSM (0.9921) and DeepFool (0.9770), with slightly lower but still robust performance against PGD. Defensive Distillation offers moderate performance in the case of PGD (0.7248), but is still inferior to the results achieved using Adversarial Training. Autoencoders and feature squeezing both also guard against different types of attacks, but Autoencoders perform at 0.9512 based on their Defense against PGD. While adversarially trained models consistently achieve the highest defense against different forms of attack, the ensemble method appears best suited for use with clean datasets.

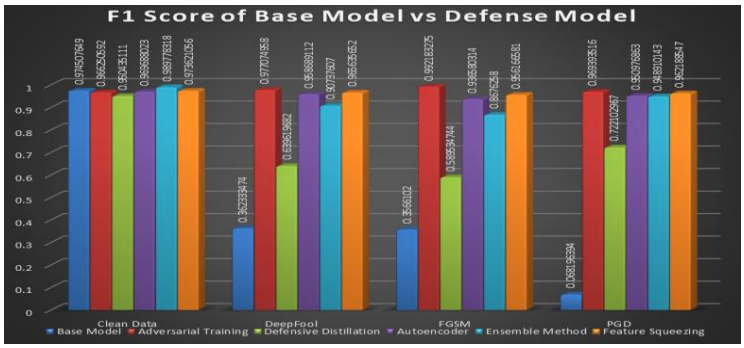


Fig.25. Visualizing the comparison and evaluation of the F1 score of the base model and the defense model

As shown in Fig. 25, different defense strategies have varying degrees of effectiveness against attacks to the base model. The Ensemble approach achieves the highest accuracy on clean data (0.9898), while the Base Model ranks second (0.9745). Adversarial Training also shows strong results across all attacks. For FGSM, it achieved an accuracy of 0.9922, and for DeepFool, an accuracy of 0.9771. This indicates that not only does it provide improved resilience for models, but it appears to be especially effective for producing more resilient models against adversarial attacks. Additionally, Defensive Distillation also performed respectably, but not nearly as well as Adversarial Training in the case of PGD (0.7221). Feature Squeezing and Autoencoder methods are considered strong defenses, while Autoencoder showed high effectiveness against PGD (0.9510). Overall, the strongest defense remains Adversarial Training, while the Ensemble method works best with clean data.

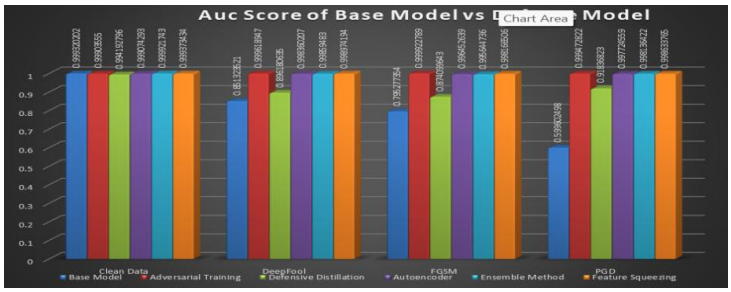


Fig.26. AUC Score visualization for the evaluation and comparison of the base model versus the defense model

Fig. 26 shows how different defense strategies work against attacks on the base model. All models perform well on clean data, with the ensemble method getting the highest accuracy (0.9999) and the base model coming in second at 0.9993. Adversarial training is very strong against all types of attacks, maintaining near-perfect accuracy against FGSM (0.9999) and DeepFool (0.9996). Defensive distillation also performs well, particularly against PGD (0.9184), although it is less effective than adversarial training. The autoencoder and feature squeezing methods also have strong defenses. Even against PGD, they are still very accurate (0.9977 for the autoencoder and 0.9986 for feature squeezing). Overall, adversarial training and the ensemble method demonstrate the best performance, with adversarial training showing superior robustness across all attack types.

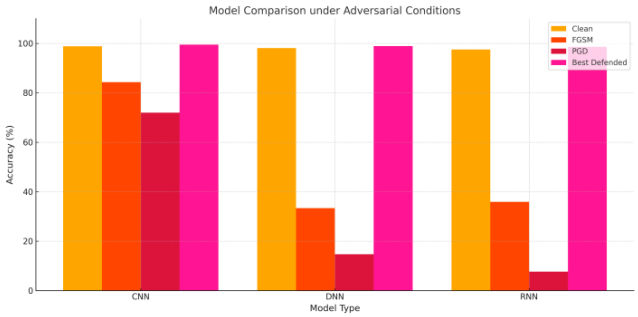


Fig.27. Comparison of neural network models in adversarial attack and defense scenarios

4.5. Neural network Models Comparison under Adversarial Conditions

Figure 27 shows that the CNN model exhibits high baseline performance and maintains strong robustness when paired with effective defenses like adversarial training. The DNN model suffers the steepest drop under adversarial attacks (e.g., PGD), but defenses significantly restore its accuracy close to clean performance. The RNN model is

highly vulnerable to strong attacks, especially PGD (dropping below 10%), yet benefits greatly from defense mechanisms. Across all models, adversarial training consistently leads to the best defense performance, restoring accuracy near 99%. Feature Squeezing and Autoencoders offer reliable, lightweight defenses, especially in CNNs and DNNs. Overall, CNNs demonstrate the best robustness and recovery balance, while RNNs require stronger defense strategies to reach comparable security levels.

5. Conclusion and Future Work

The results of this study have significant implications for policy, especially in fields where machine learning (ML) models are very important. Policymakers should consider formulating regulatory frameworks that necessitate comprehensive testing for adversarial robustness during the deployment of AI and ML systems. These frameworks should include provisions for continuous monitoring and regular updates of defense systems to keep up with how adversarial attacks change over time. This research signifies a significant progression in the security of AI and ML. This study examined the performance of Convolutional Neural Networks (CNN), Deep Neural Networks (DNN), and Recurrent Neural Networks (RNN) on the MNIST dataset under various attacks, such as FGSM, DeepFool, and PGD. The results showed a big drop in performance, especially in RNNs. CNN had the strongest baseline robustness of all the models tested, while RNN needed stronger defense strategies to get back to normal performance. Five defense mechanisms, Adversarial Training, Defensive Distillation, Autoencoder, Ensemble Method, and Feature Squeezing were put into action and tested. Adversarial Training was the most effective of these. It consistently brought model performance back to levels close to those seen under clean conditions for all architectures and attack types. These defensive strategies make the model more robust by reducing the effect of harmful inputs that are meant to trick the system. The results show that PGD attacks cause accuracy drops of about 27% in CNN, 83% in DNN, and 90% in RNN when there is no defense. This shows that these systems are very weak. But when defenses are used, all models get back to at least 98.9% accuracy, and adversarial training can make them up to 90% better at dealing with attacks. CNN is the most robust model at its base, while DNN and RNN depend a lot on defense mechanisms. These results give us important information about how to make machine learning systems that are safer and more resistant to attacks.

The results of this study can help ML professionals make systems that are safer and more reliable. They also emphasize the importance of creating flexible, multi-layered defense strategies and set the stage for more research into the complicated issues of model weaknesses and adversarial behaviors. Future research should concentrate on investigating innovative defense mechanisms, confirming their effectiveness across various application domains, and enhancing theoretical frameworks to more accurately forecast and alleviate adversarial risks. Future endeavors will seek to expand these methodologies to natural language processing (NLP), implement them on real-world datasets (e.g., CIFAR-10, ImageNet), evaluate defenses such as TRADES and MART, and enhance model interpretability in adversarial contexts.

All the Declarations and Statements

Author Contributions Statement

Surekha M – Conceptualization, methodology, data curation, code implementation, model training, validation, performance evaluation, and original draft preparation.

Anil Kumar Sagar – Supervision, review and editing, project administration, coordination of project milestones and deadlines, drafting of the initial manuscript, and contribution to the literature survey.

Vineeta Khemchandani – Supervision, review and editing, project administration, formal analysis, visualization, statistical analysis, and coordination of project milestones and deadlines.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Funding Declaration

The authors declare that no funds, grants, or other financial support were received for conducting this research.

Data Availability Statement

This study analyzed publicly available datasets. The results and datasets are available at https://www.tensorflow.org/datasets/catalog/mnist, accessed on 15 March 2025.

Ethical Declarations

This study did not involve any experiments on human participants or animals. Therefore, ethical approval was not required.

Acknowledgments
The authors gratefully acknowledge the academic support and research facilities provided by JSS Academy of Technical Education, Noida, Uttar Pradesh, India, and Sharda University, Greater Noida, Uttar Pradesh, India, which contributed to the successful completion of this study.

Declaration of Generative AI in Scholarly Writing
AI-assisted tools were used solely for language polishing and grammar checking. The authors take full responsibility for the scientific content.

Abbreviations
The following abbreviations are used in this manuscript:

ML - Machine Learning
AI - Artificial Intelligence
DL - Deep Learning
CNN - Convolutional Neural Network
DNN - Deep Neural Network
RNN - Recurrent Neural Network
MNIST - Modified National Institute of Standards and Technology
PGD - Projected Gradient Descent
DF - DeepFool
FGSM - Fast Gradient Sign Method
NLP - Natural Language Processing
AML - Adversarial Machine Learning
AUC-ROC - Area Under the Receiver Operating Characteristic Curve
DD - Defensive Distillation
EM - Ensemble Methods
LSTM - Long Short-Term Memory
GRU - Gated Recurrent Unit
AD - Autoencoders as Defense
FS - Feature Squeezing
NN - Neural Network
TPR - True Positive Rate
FPR - False Positive Rate
TP - True Positives
FP - False Positives
FN - False Negatives
TN - True Negatives
AT - Adversarial Training
BM - Base Model
AA - Adversarial Attacks
NLP - Natural Language Processing

Appendices
Not applicable.

References

[1] Z. Zhou, H. Guan, M. M. Bhat, and J. Hsu, “Fake news detection via NLP is vulnerable to adversarial attacks,” *arXiv preprint*, arXiv:1901.09657, 2019.
[2] A. I. Newaz, A. K. Sikder, M. A. Rahman, and A. S. Uluagac, “A survey on security and privacy issues in modern healthcare systems: Attacks and defenses,” ACM Trans. Comput. Healthcare, vol. 2, no. 3, pp. 1–44, 2021.
[3] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, “Darts: Deceiving autonomous cars with toxic signs,” *arXiv preprint*, arXiv:1802.06430, 2018.
[4] A. L. Caterini and D. E. Chang, “Recurrent neural networks,” in *Deep Neural Networks in a Mathematical Framework*, SpringerBriefs in Computer Science. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-75304-1_5.
[5] J. Wang, C. Wang, Q. Lin, C. Luo, C. Wu, and J. Li, “Adversarial attacks and defenses in deep learning for image recognition: A survey,” *Neurocomputing*, vol. 514, pp. 162–181, 2022.
[6] X. Yuan, P. He, Q. Zhu, and X. Li, “Adversarial examples: Attacks and defenses for deep learning,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.
[7] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, 2016.

[8] S. Zhou, C. Liu, D. Ye, T. Zhu, W. Zhou, and P. S. Yu, "Adversarial attacks and defenses in deep learning: From a perspective of cybersecurity," *ACM Comput. Surv.*, vol. 55, no. 8, pp. 1–39, 2022.
[9] N. Akhtar, A. Mian, N. Kardan, and M. Shah, "Advances in adversarial attacks and defenses in computer vision: A survey," IEEE Access, vol. 9, pp. 155161–155196, 2021.
[10] S. Kaviani, S. Shamshiri, and I. Sohn, "A defense method against backdoor attacks on neural networks," Expert Syst. Appl., vol. 213, p. 118990, 2023.
[11] O. Özdenizci and R. Legenstein, "Adversarially robust spiking neural networks through conversion," *arXiv preprint arXiv:2311.09266*, 2023.
[12] A. Oprea and A. Vassilev, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations," Natl. Inst. Stand. Technol. (NIST), NIST AI 100-2 E2023, 2023.
[13] R. S. S. Kumar et al., "Adversarial machine learning—industry perspectives," in Proc. IEEE Secur. Privacy Workshops (SPW), San Francisco, CA, USA, 2020, pp. 69–75.
[14] M. Macas, C. Wu, and W. Fuertes, "Adversarial examples: A survey of attacks and defenses in deep learning-enabled cybersecurity systems," Expert Syst. Appl., vol. 238, p. 122223, 2024.
[15] H. Ren, T. Huang, and H. Yan, "Adversarial examples: Attacks and defenses in the physical world," Int. J. Mach. Learn. Cybern., vol. 12, no. 11, pp. 3325–3336, 2021.
[16] Y. Zhang, T. Du, S. Ji, P. Tang, and S. Guo, "RNN-Guard: Certified robustness against multi-frame attacks for recurrent neural networks," *arXiv preprint*, arXiv:2304.07980, 2023.
[17] A. H. Galib and B. Bashyal, "On the susceptibility and robustness of time series models through adversarial attack and defense," *arXiv preprint*, arXiv:2301.03703, 2023.
[18] Y. Wang, D. Du, H. Hu, Z. Liang, and Y. Liu, "TSFool: Crafting highly-imperceptible adversarial time series through multi-objective attack," in *ECAI 2024*, pp. 1422–1429, IOS Press, 2024.
[19] P. Sokerin, D. Anikin, S. Krehova, and A. Zaytsev, "Concealed adversarial attacks on neural networks for sequential data," *arXiv preprint*, arXiv:2502.20948, 2025.
[20] S. Goyal, S. Doddapaneni, M. M. Khapra, and B. Ravindran, "A survey of adversarial defenses and robustness in NLP," ACM Comput. Surv., vol. 55, no. 14s, pp. 1–39, 2023, doi: 10.1145/3593042.
[21] X. Ma, Y. Niu, L. Gu, Y. Wang, Y. Zhao, J. Bailey, and F. Lu, "Understanding adversarial attacks on deep learning based medical image analysis systems," Pattern Recognition, vol. 110, p. 107332, 2021, doi: 10.1016/j.patcog.2020.107332.
[22] P. Purwono, A. Ma'arif, W. Rahmaniar, H. I. K. Fathurrahman, A. Z. K. Frisky, and Q. M. ul Haq, "Understanding of convolutional neural network (CNN): A review," Int. J. Robot. Control Syst., vol. 2, no. 4, pp. 739–748, 2022.
[23] A. A. Elngar, M. Arafa, A. Fathy, B. Moustafa, O. Mahmoud, M. Shaban, and N. Fawzy, "Image classification based on CNN: a survey," J. Cybersecur. Inf. Manag., vol. 6, no. 1, pp. 18–50, 2021.
[24] A. S. Hashemi and S. Mozaffari, "CNN adversarial attack mitigation using perturbed samples training," Multimed. Tools Appl., vol. 80, pp. 22077–22095, 2021.
[25] S. Alzaidy and H. Binsalleeh, "Adversarial attacks with defense mechanisms on convolutional neural networks and recurrent neural networks for malware classification," Appl. Sci., vol. 14, no. 4, p. 1673, 2024.
[26] Y. Watanobe, M. M. Rahman, M. F. I. Amin, and R. Kabir, "Identifying algorithm in program code based on structural features using CNN classification model," Appl. Intell., vol. 53, no. 10, pp. 12210–12236, 2023.
[27] W. Zhao, S. Alwidian, and Q. H. Mahmoud, "Adversarial training methods for deep learning: A systematic review," Algorithms, vol. 15, no. 8, p. 283, 2022.
[28] S. Zhang, H. Gao, and Q. Rao, "Defense against adversarial attacks by reconstructing images," IEEE Trans. Image Process., vol. 30, pp. 6117–6129, 2021.
[29] W. Villegas-Ch, A. Jaramillo-Alcázar, and S. Luján-Mora, "Evaluating the robustness of deep learning models against adversarial attacks: An analysis with FGSM, PGD and CW," Big Data Cogn. Comput., vol. 8, no. 1, p. 8, 2024.
[30] X. Shi, Y. Peng, Q. Chen, T. Keenan, A. T. Thavikulwat, S. Lee, and Z. Lu, "Robust convolutional neural networks against adversarial attacks on medical images," Pattern Recognit., vol. 132, p. 108923, 2022.
[31] M. Imran, A. Appice, and D. Malerba, "Evaluating realistic adversarial attacks against machine learning models for Windows PE malware detection," Future Internet, vol. 16, no. 5, p. 168, 2024, doi: 10.3390/fi16050168.
[32] H. Liang, E. He, Y. Zhao, Z. Jia, and H. Li, "Adversarial attack and defense: A survey," Electronics, vol. 11, no. 8, p. 1283, 2022.
[33] M. Ibn Khedher and M. Rezzoug, "Analyzing adversarial attacks against deep learning for robot navigation," in Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART), 2021, pp. 1114–1121, doi: 10.5220/0010323611141121.
[34] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "A survey on adversarial attacks and defences," CAAI Trans. Intell. Technol., vol. 6, no. 1, pp. 25–45, 2021.
[35] J. Sen and S. Dasgupta, "Adversarial attacks on image classification models: FGSM and patch attacks and their impact," arXiv preprint, arXiv:2307.02055, 2023.
[36] N. Kaushik, V. Bhardwaj, and H. S. Arri, "A machine learning-based survey of adversarial attacks and defenses in malware classification," in 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2023, pp. 1–7, doi: 10.1109/ICCCNT56998.2023.10306555.
[37] Y. Li, M. Cheng, C. J. Hsieh, and T. C. Lee, "A review of adversarial attack and defense for classification methods," Am. Stat., vol. 76, no. 4, pp. 329–345, 2022.
[38] M. Surekha, A. K. Sagar, and V. Khemchandani, "Adversarial Attack and Defense Mechanisms in Medical Imaging: A Comprehensive Review," in Proc. IEEE IC2PCT, vol. 5, pp. 1657–1661, Feb. 2024.
[39] Y. Wang, J. Liu, X. Chang, J. Mišić, and V. B. Mišić, "IWA: Integrated gradient-based white-box attacks for fooling deep neural networks," Int. J. Intell. Syst., vol. 37, no. 7, pp. 4253–4276, 2022.
[40] J. Li, Y. Xu, Y. Hu, Y. Ma, and X. Yin, "You only attack once: Single-step DeepFool algorithm," Appl. Sci., vol. 15, no. 1, p. 302, 2024.
[41] A. Shafahi et al., "Adversarial training for free!," in Adv. Neural Inf. Process. Syst., vol. 32, 2019.

[42] M. Zhao, L. Zhang, J. Ye, H. Lu, B. Yin, and X. Wang, "Adversarial Training: A Survey," arXiv preprint, arXiv:2410.15042, 2024.
[43] M. Wang and M. Xu, "An Adversarial Machine Learning-Based Fast Detection Method for Denial of Service-Oriented Cyber Attacks in Internet of Vehicles," J. Circuits Syst. Comput., vol. 33, no. 7, p. 2450122, 2024.
[44] I. Hong and S. Lee, "Exploring Synergy of Denoising and Distillation: Novel Method for Efficient Adversarial Defense," Appl. Sci., vol. 14, no. 23, p. 10872, 2024.
[45] M. S. Haroon and H. M. Ali, "Ensemble adversarial training based defense against adversarial attacks for machine learning-based intrusion detection system," Neural Netw. World, vol. 317, p. 336, 2023.
[46] S. N. Ashraf, R. Siddiqi, and H. Farooq, "Auto encoder-based defense mechanism against popular adversarial attacks in deep learning," PLoS ONE, vol. 19, no. 10, p. e0307363, 2024.
[47] T. van Weezel, F. van Ree, T. Bos, P. Bastiaanssen, and S. Hess, "Purifying Adversarial Examples Using an Autoencoder," in Int. Conf. Discovery Science, Cham: Springer, pp. 134–148, Oct. 2024.
[48] A. Yinusa and M. Faezipour, "A multi-layered defense against adversarial attacks in brain tumor classification using ensemble adversarial training and feature squeezing," Sci. Rep., vol. 15, no. 1, pp. 1–11, 2025.
[49] F. Pistorius, D. Grimm, F. Erdösi, and E. Sax, "Evaluation matrix for smart machine-learning algorithm choice," in Proc. Int. Conf. Big Data Anal. Pract. (IBDAP), pp. 1–6, 2020.
[50] S. M. Basha and D. S. Rajput, "Survey on evaluating the performance of machine learning algorithms: Past contributions and future roadmap," in Deep Learn. Parallel Comput. Environ. Bioeng. Syst., Academic Press, pp. 153–164, 2019.
[51] G. Naidu, T. Zuva, and E. M. Sibanda, "A review of evaluation metrics in machine learning algorithms," in Proc. Comput. Sci. Online Conf., Cham: Springer, pp. 15–25, 2023.

Authors' Profiles

Surekha M. is a Ph.D. Scholar in Computer Science and Engineering at Sharda University, Greater Noida, India. Her ORCID ID is 0000-0002-7338-8267. Her research interests include Machine Learning, Cybersecurity, and Artificial Intelligence**.** She completed her M.Tech. (CSE) from Dr. A. P. J. Abdul Kalam University, India, and is currently pursuing her Ph.D. at Sharda University. She is working as an Assistant Professor at JSS Academy of Technical Education (JSSATE), Noida, Uttar Pradesh, India. She has over 10 years of teaching experience in the field of Computer Science and Engineering and has published 2 indexed journal papers**,** 3 Scopus-indexed papers, and 1 ESCI-indexed journal paper.

Dr. Anil Kumar Sagar is currently a Professor in the Department of Computer Science and Engineering, School of Engineering and Technology, Sharda University, India. His ORCID iD is 0000-0002-5991-2835. He earned his Ph.D. from Jawaharlal Nehru University (JNU), New Delhi, with specialization in Ad-hoc Networks. He completed his B.E. in Computer Science and Engineering from G. B. Pant Engineering College, Pauri Garhwal**,** and his M.Tech**.** from JSS Academy of Technical Education (JSSATE), Noida**.** Dr. Sagar is an active member of editorial boards and review committees for several national and international journals and has served as a program and organizing committee member for multiple conferences.

Dr. Vineeta Khemchandani is currently working as DEAN of the School of Computer Applications and Technology. She has overall experience of around 27.5 years in the field of IT in various organizations. The experience spans software development, academics, administration and Research, in various capacities. She has been associated with Galgotias University since 2023. She is holding a Doctorate (Ph.D) in Computer Science (2011). Post Graduate in Computer Applications (1996). Also, she holds a Diploma in Banking Technology from the Indian Institute of Banking and Finance, Mumbai. She has published 30 research papers, authored 4 books and 6 book chapters, and reviewed 1 book. She has guided 1 Ph.D. student, 3 M.Tech students, and several B.Tech students, with 2 Ph.D. students currently in progress.

How to cite this paper

Surekha M., Anil Kumar Sagar, Vineeta Khemchandani, "Enhanced Robustness of Neural Network Models against Adversarial Attacks and Performance Analysis", International Journal of Intelligent Systems and Applications(IJISA), Vol.18, No.4, pp.138-160, 2026. DOI:10.5815/ijisa.2026.04.08