International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V73I6P112

Volume 73 Issue 6, 123-148, June 2025

Original Article

Integrated Defense Training Framework for Countering Gradient-Based Miniature Attacks on Deep Image Recognition Systems

Lavanya Sanapala[1], Lakshmeeswari Gondi[2]

[1,2]*Department of Computer Science and Engineering, GITAM Deemed to be University, Andhra Pradesh, India.*

[1]*Corresponding Author : lavanya.sanapala@gmail.com*

Received: 27 January 2025 Revised: 20 May 2025 Accepted: 03 June 2025 Published: 28 June 2025

Abstract - *Deep Learning Models (DLMs) have become indispensable in Deep Image Recognition Systems (DIRS) due to their ability to automatically extract intricate features and maintain high performance over time. Despite their effectiveness, DLMs are intrinsically susceptible to gradient-based attacks, in which adversaries subtly alter input data to trick the model and produce inaccurate predictions. Adversarial Machine Learning (AML), which investigates diverse attack strategies and develops countermeasures, has yielded numerous techniques. However, advanced gradient-based attacks remain a persistent challenge, underscoring the need for more effective detection and mitigation strategies. This paper presents the Gradient-based Adversarial Miniature Attack (GMA), a sophisticated gradient-based attacking technique that thoroughly assesses the resilience of DIRS models against hostile assaults. This research suggests the Model Integration Approach (MIA), a defense training approach significantly improving DIRS resilience to combat GMA and other well-known threats. According to experimental results, MIA has a remarkable 99.71% detection accuracy, indicating its potential as a strong countermeasure. This work lays a solid foundation for advancing defenses against sophisticated gradient-based adversarial attacks while fostering innovation in developing secure and reliable DIRS models.*

Keywords - *Computer vision, Security, Adversarial Machine Learning, Advanced threat detection, Robust Deep Neural Network.*

1. Introduction

Deep Image Recognition Systems (DIRS) are a key application area for a variety of image identification tasks, including autonomous traffic recognition [1], fingerprint spoof detection [2], cancer cell categorization [3], and image classification [4]. These applications actively automate their duties using Deep Learning Models (DLMs). While some DIRS applications use them to categorize inputs as benign or malignant, others concentrate on anomaly identification. DIRS takes images as inputs, processes them, and produces outputs based on the choices made by the underlying DLMs. Convolutional Neural Networks (CNNs), VGG, and ResNet architectures are some of the most popular models for accomplishing these goals. However, numerous researchers in the AML field have demonstrated that these DLMs are susceptible to adversarial poisoning assaults.

This research area aims to identify the weaknesses in the current DLMs and provide suitable, strong defenses. In keeping with this objective, numerous studies have shown how susceptible DLMs are to various adversarial poisoning assaults, including the Fast Gradient Sign Method by Ian Goodfellow et al. (FGSM) [5], Carlini and Wagner (CW) [6], and Projected Gradient Descent (PGD) [7]. The specified attacks aim to manipulate the pre-trained DLMs' decisions, resulting in incorrect categorization or prediction outcomes. For instance, a DIRS model under adversarial attack may mistakenly identify a cancerous cell as benign. It may cause an incorrect diagnosis of the illness and can trigger fear. Considering this, research demonstrates that there is every chance to mislead the DLMs using sophisticated threat models. Several adversarial poisoning attacks' effects on their target DLMs in diverse application areas were mentioned in [8-16].

This situation raises significant security issues with DLMs in their respective application domains, particularly in the cyber security and healthcare industries, where protecting sensitive data from adversarial attacks is essential. Numerous studies have produced strong defense and detection techniques against a variety of adversarial attacks as a result of the developments in adversarial machine learning. Defensive distillation, a technique developed by N. Papernot et al. [17], uses two feed-forward deep networks (a teacher model and a student model) to defend against FGSM attacks successfully. Despite achieving broad defense capacity, this approach could

not fend off stronger adversarial attacks (CW) proposed by Carlini and Wagner et al. [18]. The denoising [19] defense technique effectively identifies FGSM and PGD. However, it has several drawbacks, including a gradient obfuscation issue, a failure to respond to sophisticated attacks, and a decline in clean accuracy. Kuzlu M et al. [20] developed a hybrid approach that counteracts FGSM, PGD, Momentum Iterative Method (MIM), and Basic Iterative Technique (BIM) by combining distillation and adversarial training strategies.

This approach has limitations against unknown attacks such as CW, yet it provides significant compute costs for creating different adversarial cases for retraining. Using the consistency checking method [21] to identify adversarial examples is effective against FGSM and PGD, but advanced attacks and CW are still undetectable. Hardware acceleration parameters are used for detection in GPU monitoring [22], but this requires a lot of processing power, which is not ideal. Although GAN-based defenses [23, 39] are efficient methods for identifying different types of attacks, their intricate structures make it challenging to implement and maintain them in actual scenarios. The most widely used and successful method for identifying known attack routes is adversarial training [24].

At the same time, it has drawbacks, such as the production of adversarial data and vulnerability to invisible attacks. Numerous hybrid approaches have been developed by [20, 25, 26, 29] that work well for particular assault areas. For academics interested in this area, the adaptability of current defenses to hidden flaws remains an open problem. The following is a summary of the difficulties that the most advanced defense techniques encounter, taken from the recent literature:

- The defense or detection techniques that have been outlined are still vulnerable to unnoticed or very sophisticated hostile attacks.
- Even with the use of sophisticated training architectures and many models for training, specific techniques are still unable to identify CW attacks and other sophisticated and hidden attack routes.
- The main issue with adversarial training, the best defense technique available today, is that it requires much computing to generate adversarial examples. It is time-consuming, particularly for iterative methods like CW examples.
- Various hybrid techniques lead to a loss of clean accuracy rates, indicating that the defense or detection model cannot identify benign samples, lowering the model's efficiency.

To protect against such sophisticated attack vectors, searching for hidden vulnerabilities in DLMs, evaluating defense models regularly, and creating robust techniques and defense models are imperative.

This study aims to solve two issues that state-of-the-art defense techniques have concerning DIRS.

1. A robust assessment of defense strategies against an advanced adversary attack that is not yet discovered.
2. To lower the number of adversarial examples needed for training.

In this regard, this paper presents the significant contributions of the study as follows:

1. Design and implement a novel, advanced, Gradient-based adversarial Miniature Attack (GMA).
2. Perform robustness evaluation of the state-of-the-art defense models for DIRS applications against GMA attacks.
3. Propose a novel training framework - Model Integration Approach (MIA) to train GMA and other state-of-art attacks (PGD, FGSM, CW).
4. Evaluate the effectiveness of the proposed MIA method rigorously against state-of-art methods (bilateral adversarial training [40], fast adversarial training [32], convolutional filter statistics [31], and GAN-based training [39]).
5. The GMA attack is an unseen and advanced attack on the existing DIRS defense models. The novelty of the attack lies in the miniature perturbations used for the attack. The proposed training method helps the MIA defense model detect the GMA, PGD, FGSM, and CW attacks efficiently with higher accuracy rates. The novelty lies in training the defense model in phases, and thus MIA lessens the number of adversarial examples essential for training. MIA efficiently withstands advanced adversarial attacks like GMA, PGD, FGSM, and CW, enhancing the DIRS system's robustness and ensuring dependable performance in adversarial environments.

The study is segregated and presented in different sections of this paper. Section 2 gives background information in brief. Section 3 presents the GMA attack and related analysis techniques. Section 4 describes the suggested training approach - the Model Integration Approach (MIA). Section 5 provides experimentation details. Section 6 discusses the experimental results. The conclusions are drawn in section 7.

2. Background

2.1. Gradient Learning and Gradient Optimization

Gradient optimization is one of the steps involved in the DLM training process. Figure 1 depicts the DLM training process involved in learning patterns of ‘image 3’ and recognizing its corresponding ‘label 3’ for a multiclass handwritten image recognition classification problem. The input image undergoes pre-processing steps like feature mapping, activating convolutional layers, max_pooling, and flattening.

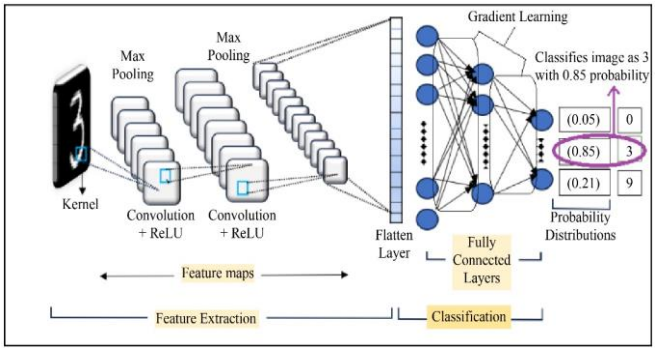


Fig. 1 The process of building a deep neural network to solve a multiclass classification problem using an image of handwritten numbers

Thus, pre-processed information is fed to the neural network nodes, where gradient learning happens. Gradient learning helps activate appropriate nodes through the network layers that could lead the model to predict the corresponding label accurately. Gradient learning is done by assigning two model parameters - weights (w) and bias (θ). Initially, these parameters are assigned with random values. Then, DLM computes a decision function shown in Equation 1 iteratively to choose the best parameters of w and θ that produce the best probability distribution values for predicting the appropriate class label for an input image.

$$y = f(x) = \ w.x \ + \ \theta \qquad (1)$$

The function f(x) to iteratively calculate and update values of w and θ is called 'gradient optimization.' A metric called cost function 'J' measures the differences between the actual and expected model outcomes. Adjusting model parameters, J acts as a feedback loop and guides the optimization process. Gradient optimization is a complex and iterative process. It involves iteratively adjusting weights (w) to minimize J, with gradients computed to determine the direction and magnitude of weight updates.

This process enables the model to converge toward optimal parameters that minimize J progressively. DLM models often employ gradient-optimization methods such as SGD, Adam, RMSProp, and Adagrad [30] to traverse the optimization landscape and guarantee convergence. These methods provide stable and dependable model performance, serving as the foundation of contemporary deep-learning training pipelines [31]. Understanding gradient optimization is relevant because the gradient-based attacks target the learned gradients of the DLM and generate appropriate gradient-based attacks to deceive the DLMs widely.

2.2. Gradient-based Adversarial Attack

The gradient-based adversarial attacks target the learned gradients of the model and deliberately manipulate them such that the model under attack fails to recognize the correct label and makes false predictions.

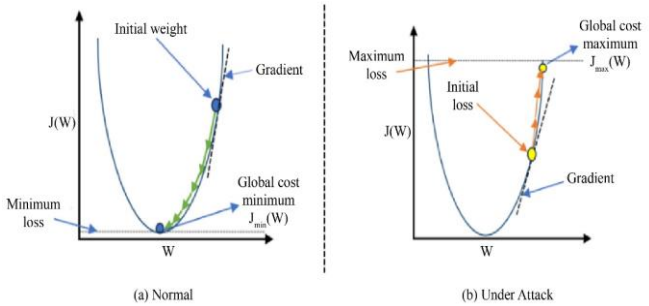


Fig. 2 Gradient optimization dynamics of the DNN model under (a) Normal, and (b) Gradient-poisoning attack conditions

Figure 2 displays the gradient propagation under two extremes. 1. under normal condition shown in Figure 2(a) and 2. under attack condition shown in Figure 2(b). Under normal conditions, the model used random values to establish the model weights w, and gradient descent refines them iteratively. The procedure guides (x) in the direction of the global minimum, $J_{\min}(w)$ in steps, as seen in Figure 2(a). This approach guarantees that the model's choices align with its stated goal. Under attack conditions, gradient ascent is initiated by purposefully manipulating the gradients to move in the other direction using attack samples. As seen in Figure 2(b), this modification directs the optimization process toward a global maximum, $J_{\max}(w)$, which in turn causes the decision function to perturb to $f'(x)$. As a result, the model's behavior deviates from its intended purpose, compromising its accuracy and dependability.

2.3. Adversarial Training (AT)

AT is one of the defense methods that enhance the robustness of DLMs against adversarial attacks. The initial step in this process is to prepare a training dataset that includes clean (original) samples and adversarial examples using specific attack algorithms, such as the PGD or FGSM. Since adversarial datasets are typically not publicly available, these examples must be generated synthetically. Each sample-whether clean or adversarial-must be correctly labeled with its corresponding class, and the combined trainset is fed for model training and promotes adversarial robustness. Following the training phase, the model is expected to effectively identify and mitigate adversarial inputs intended to compromise its performance. A plethora of adversarial training methods have been developed to address various adversarial situations, such as domain-specific [33-40], transferability nature [41], handling data and model complexities [42-45], etc.

3. The Gradient-Based Miniature Attack (GMA) Procedure

The GMA is an advanced adversarial attack technique to gently tamper with the gradients of a deep learning model in order to impair the model's overall performance. First, the attack generates GMA samples inducing miniature perturbations, as described in Algorithm 1. The training

dataset is then purposefully supplemented with these samples. What distinguishes GMA from traditional attacks-such as CW, PGD, and FGSM-is its use of miniature perturbations.

Miniature Perturbations: The miniature perturbations are the truncated sizes used to craft GMA attack samples. These are significantly smaller than those typically employed in other standard attacks (sizes used are 3, 2, 1, 2.7, 1.5) but are sufficient to degrade the performance of deep learning classifiers. Miniature perturbations are denoted by Tr(ϵ), a set of unique perturbation values computed as shown in Equation 2.

$$Tr(\epsilon) = V_n * e^{-i} \mid V_n \in [0.1, 3.0], e^{-i} \in [1,6] \quad (2)$$

Where V = {0.1, 0.5, 1.0, 2.0, 3.0}, n denotes the index number of elements in V (starting from 1), and e^{-i} represents the exponential term, with 'i' taking the values from the set {1,2,3,4,5,6}. For instance, if n=1 and i=1, the truncated perturbation size Tr(ϵ) used to craft the attack sample would be $V_1*e^{-1} = 0.1*e^{-1}$. Each value obtained after computing V_n*e^{-i} is in miniature set Tr(ϵ) used for generating synthetic GMA adversarial examples as defined in algorithm 1. The maximum and minimum values obtained for sets for 'V' and 'i' are $3e^{-1}$ and $0.1e^{-6}$. These are chosen based on the attack success rate, the significant effect of perturbation size on the target model, and the visible similarity between the original and adversarial images. The values beyond the above-mentioned limits have little effect on model accuracy, are visibly distinguishable, and are discarded for this study. The fundamental premise of the GMA is that the attacker has complete access to the dataset and DLM in order to optimize the attack plan successfully. Even though these perturbations are small, they are efficient enough to skew the model's gradient calculations, which results in notable errors during inference and training. These are subtle enough to evade detection by most existing DIRS defense models yet powerful enough to skew model learning dynamics and degrade classification accuracy. As a result, models trained with GMA-contaminated data may produce unreliable predictions, posing serious risks to the reliability, integrity, and availability of the DIRS system. GMA's creative use of tiny perturbations provides a flexible method for researching sophisticated adversarial vulnerabilities in various DLMs, bridging the gap between stealth and attack effectiveness.

3.1. Attack Formulation

Gradient-based Miniature attack leverages the gradients of the loss function for the miniature data to craft GMA samples. Let:

- Training data: $D = \{(a_i, b_i)\}_{i=1}^{n}$
- GMA Data: $D_g = D \cup \{(a_g, b_g)\}$, where (a_g, b_g) are the GMA samples.
- Model Parameters: θ, learned by minimizing the training loss $L(\theta, D_g)$.
- Target Data: (a_t,b_t), which the attack aims to misclassify.

The attack algorithm solves the following bi-level optimization problem:

$$\max_{(a_g, b_g)} L_{gama}(\theta^*, (a_t, b_t)) \quad (3)$$

subject to: $\theta^* = \underset{\theta}{argmin} L(\theta, D \cup \{(a_t, b_t)\})$ where, L_{gama} is the attack-specific loss (e.g., causing misclassification of (a_t,b_t)), and L is the model's standard training loss (e.g., cross-entropy or MSE).

To solve the bi-level optimization problem, Gradient-based miniature attacks approximate the impact of the GMA sample (a_g,b_g) on the model's performance. This involves computing meta-gradients.

3.1.1. Computing the Meta-Gradients

The impact of (a_g,b_g) on the target loss L_{GMA} can be computed as:

$$\frac{\partial L_{gama}}{\partial (a_g,b_g)} = \frac{\partial L_{gama}}{\partial \theta^*} \cdot \frac{\partial \theta^*}{\partial (a_g,b_g)}, \quad (4)$$

Where:

- $\frac{\partial L_{gama}}{\partial (a_g,b_g)}$: Gradient of the attack loss with respect to model parameters.
- $\frac{\partial \theta^*}{\partial (a_g,b_g)}$: Influence of GMA samples on the model parameters.

Substituting back, the attack algorithm adjusts (a_g,b_g) iteratively as:

$$(a_g, b_g) \leftarrow (a_g, b_g) + \eta \cdot \frac{\partial L_{gama}}{\partial (a_g,b_g)} \quad (5)$$

Where 'η' is the learning rate for crafting the GMA samples.

3.2. GMA Synthetic Data Generation Procedure

Algorithm 1 outlines the procedure for generating gradient-based miniature attack datasets synthetically. The algorithm uses miniature perturbations of varying intensities drawn from a predefined list of perturbation sizes (*Tr(ϵ)*) to the training dataset. Starting with initial parameter settings, it generates attack samples for each perturbation size and stores them in the variable GMA_ds. The algorithm cycles through five perturbation levels (n = 0 to 4) for up to six intensity levels (i = 1 to 6). The function grad_attack uses the Attack samples thus generated to perform a GMA attack on the targeted model (M). This algorithm produces a GMA attack dataset containing miniature or GMA attack samples.

Algorithm 1: Synthetic GMA Attack Data Generation

Input: ML model(M), original dataset (D), miniature perturbation sizes (*Tr(ϵ)*).

Output: GMA_ds #Gradient-Poisoning miniature attack dataset (GMA dataset)

Begin Procedure

1. Parameter initialization
   values: n=0, i=1 then perturbation value = V_0e^{-1} as per (3).
2. while i<=6 && n<=4
3. Loop
4. attack_ds := grad_attack(
5. model = M,
6. ds = train_set,
7. Perturbation_size = $Tr(\epsilon)_n$, # value of $Tr(\epsilon)$ at index n
8. target=none)
9. GMA_ds = GMA_ds.append(attack_ds)
10. if (n >4) then i++, n=0, repeat from step 3
11. else if i==6 exit loop
12. else n++, repeat from step 3
13. End loop
14. return GMA_ds

End Procedure

3.3. GMA Efficiency Analysis Procedure

The GMA attack samples must satisfy the following two necessary and sufficient conditions for a successful attack.

1. The class label Y' of the GMA sample data X′ must not be the actual class label X, i.e., C(X) as shown in Equation 6.

$$Y' = f'(X') = (x' + Tr(\epsilon)) + b' != C(X) \quad (6)$$

Where,

Y′ : Class label predicted by the attacked model.
f′(X′) : Function learned by the model under attack.
x′+Tr(ϵ) : Adversarial sample generated with added truncated Tr(ϵ) perturbations.
b′ : Bias vector.
C(X) : True class label of the original sample X.

2. The generated GMA attack samples X′ of miniature perturbation sizes Tr(ϵ) must closely resemble its original counterpart, X.

These two conditions ensure that the perturbed sample X′, crafted using Tr(ϵ), induces the attacked model to misclassify it while maintaining high similarity to the original sample X. Algorithm 2 outlines the procedure for a GMA attack and analyzes if the attack meets the first condition mentioned above. This algorithm assesses the robustness of a deep learning model against a GMA attack. It begins by calculating the baseline accuracy (B_{acc}) of the clean model and establishes thresholds for best-case accuracy (>80%) and worst-case accuracy (<50%) to classify the model's performance. The model is then exposed to attack samples from the GMA_ds dataset in an iterative process, with the accuracy recorded after each attack (denoted as Fool(M)acc).

The attack function, using equations (3), (4), and (5), learns the gradient of the model (M) and performs gradient poisoning. If the post-attack accuracy falls below the worst-case threshold, the attack is considered successful, and the algorithm returns a value of 1.

If the attack does not succeed after a maximum of three iterations, the algorithm returns 0. The final output, Fool(M)acc, quantifies the model's vulnerability to adversarial perturbations, providing an essential metric for evaluating the DLMs' resilience under attack conditions.

Algorithm 2: GMA attack Analysis Procedure

Input: Deep neural network (M), GMA_ds.

Output: $Fool(M)_{acc}$ #Target model accuracy under GMA attack.

Begin Procedure

1. B_{acc} = Baseline classification accuracy of clean model (M)
2. $Fooled(M)_{acc}$ = Fooled model accuracy # DLM accuracy after attack
3. Determine the case behavior of model (M)
   1. Best case accuracy when $(M)_{acc} > 80\%$
   2. Worst-case accuracy when $(M)_{acc} < 50\%$
4. i = 1
5. Loop
6. Attack (M) using GMA_ds
7. $Fool(M)_{acc}$ = Attack (M, GMA_ds)
8. Check if the attack is a success or failure
   1. If $Fooled(M)_{acc}$ = worst case accuracy
      flag = 1 # Attack success
   2. else flag = 0 # Not a successful attack
   3. i++
9. if $i<=3^i$ then continue in the loop else, return the flag and exit from a loop
10. End Loop
11. Return $Fool(M)_{acc}$

End Procedure

Visual analysis is performed to ascertain whether there is a discernible similarity between legitimate and equivalent GMA adversarial images. This analysis was done manually to verify whether the output images satisfied our second necessary condition.

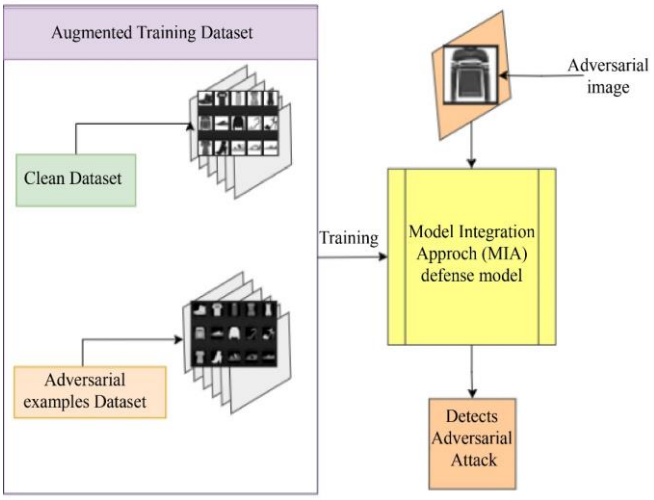


Fig. 3 The proposed model integrated approach MIA defense model overview diagram

4. The proposed Model Integration Approach (MIA)

The model integration approach (MIA) is a novel, flexible adversarial training framework. MIA strives to reduce the number of adversarial examples required for training. Figure 3 shows the MIA method overview. The defense model (MIA) is trained on the augmented dataset containing clean and adversarial examples. The trained MIA model detects the adversarial attack when an adversary tries to attack the MIA defense model. The MIA concept applies to training any defense model that wants to reduce the requirement for generating several adversarial images. This paper proposes the MIA training framework rather than the defense model because MIA concentrates on novel training techniques suitable for training any other defense models. The novelty of this approach lies in training the defense model by sequentially and intelligently integrating two training methods-Enhanced Adversarial Training (EAT) and the EAT integrated Ensemble training. The MIA effectively enhances robustness while reducing training complexities, including the need for complex model architectures, many attack samples, and reliance on third-party DNNs.

4.1. MIA Training Framework

The hardcore MIA is involved in training the defense models. It trains the defense model in two phases.

The first phase applies Enhanced Adversarial Training (EAT), and the second phase applies hybrid model integration, i.e., EAT-trained Ensemble training.

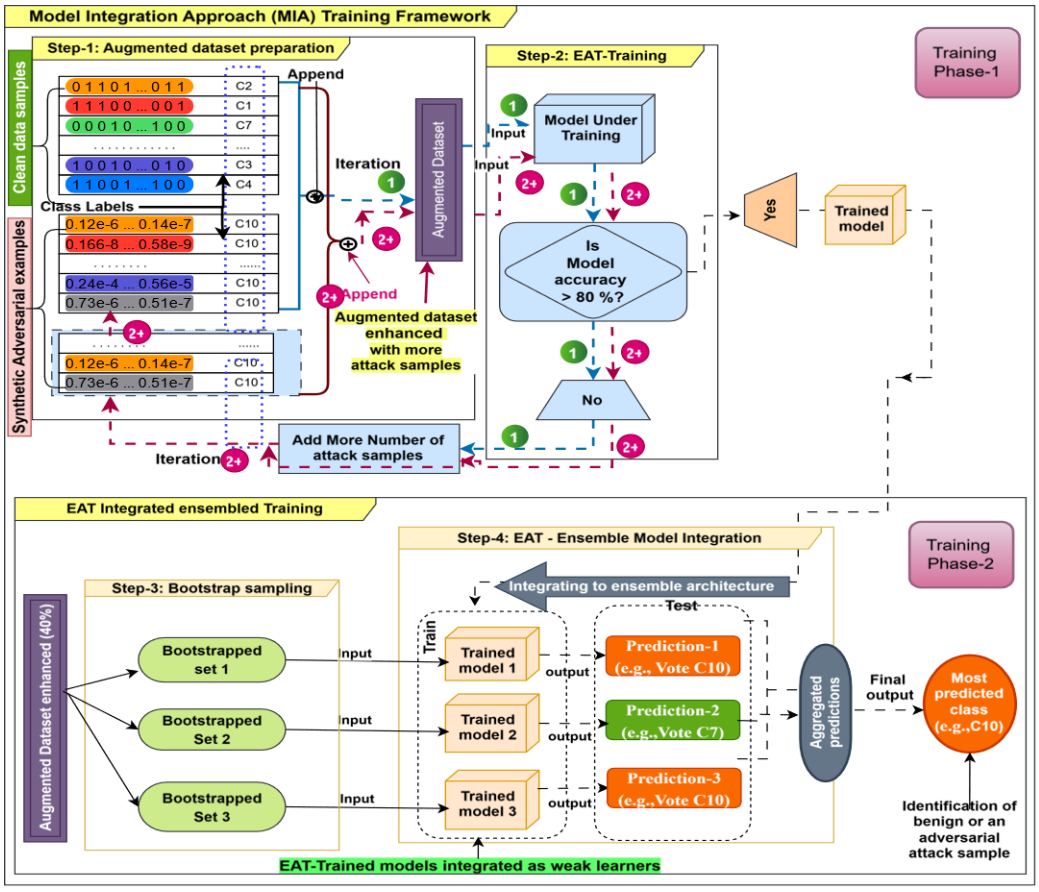


Fig. 4 MIA training details, including EAT and EAT-integrated Ensemble model training

The MIA method involves four steps: 1. Augmented data preparation, 2. Enhanced adversarial training, 3. Bootstrap dataset preparation, and 4. EAT-integrated ensemble training. Figure 4 illustrates a detailed view of the training phases involved in the MIA training framework.

4.1.1. Augmented Data Preparation

An augmented dataset is composed of both original and attack data samples. The clean dataset contains the original data records, each labeled with predefined class labels, while the attack dataset consists of adversarial data samples and their manually assigned labels. For example, the clean image samples in Figure 4 are labeled with classes such as C1 to C9 in the case of MNIST, whereas the GMA image samples are labeled as C10 as the 10th class label added to the dataset.

4.1.2. Enhanced Adversarial Training (EAT) Procedure

The EAT process involves several cycles of training the defense model over multiple cycles using an augmented dataset. In the first training cycle, a small proportion of adversarial examples (e.g., 15%) are incorporated into the augmented dataset. This initial training cycle is represented by iteration 1, as indicated by the green and blue arrows in Figure 4. In subsequent cycles (iterations 2+), the augmented dataset is progressively enhanced by adding increasing percentages of adversarial examples with each consecutive training cycle, as shown by the pink arrows. Iteratively adding proportionate adversarial examples to the augmented dataset in each iteration and training on the updated 'augmented dataset' until the model achieves optimal performance (i.e., an accuracy greater than 80%) is called Enhanced Adversarial Training.

This approach ensures that the defense model is exposed to a sufficient number of adversarial examples, allowing it to learn the attack patterns effectively. Once the EAT-trained defense model has sufficiently learned the adversarial image patterns, it is pushed forward for integration with the MIA training process.

4.1.3. Bootstrapped Dataset Preparation

The source of the bootstrapped datasets is the augmented dataset of Training phase-1 that has been enhanced iteratively with the increased proportion of adversarial examples where the EAT-trained model attains its optimal performance. The necessary percentage of adversarial cases must be present in the expanded dataset for the second training phase. The expanded dataset with 40% adversarial images is selected in this study, shown in Figure 4 (the results section explains why).

The expanded dataset is now divided into n bootstraps, each with "m" samples. Figure 4 only displays three bootstrapped sets and the accompanying EAT-trained models for simplicity's sake. Both hostile and valid samples are included in the bootstrapped datasets. The EAT-integrated defensive model uses these datasets as input.

4.1.4. EAT Integrated Ensemble Training

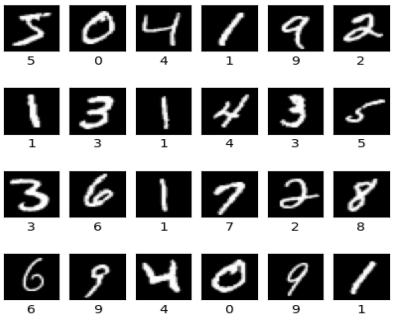
The EAT-integrated ensemble model intellectually integrates the EAT-trained models as weak learners into the ensemble model to enhance the performance of the previously trained EAT defense models in training phase 1, as shown in Figure 4. The bootstrapped datasets are used to train the EAT-integrated ensemble model. The ensemble model used here is the bootstrap aggregation ensemble technique, and hence, the datasets were bootstrapped in the earlier step. Each weak learner (EAT model) is a voting classifier for the classification task. Each classifier casts a vote for its predicted outcome, and the final prediction is determined by aggregating all votes, with the outcome receiving the highest number of votes. This aggregated result serves as the final output of the EAT-integrated Ensemble model, also named the MIA defense model. The performance of the trained defense model is subsequently evaluated on test datasets.

5. Experimental Setup

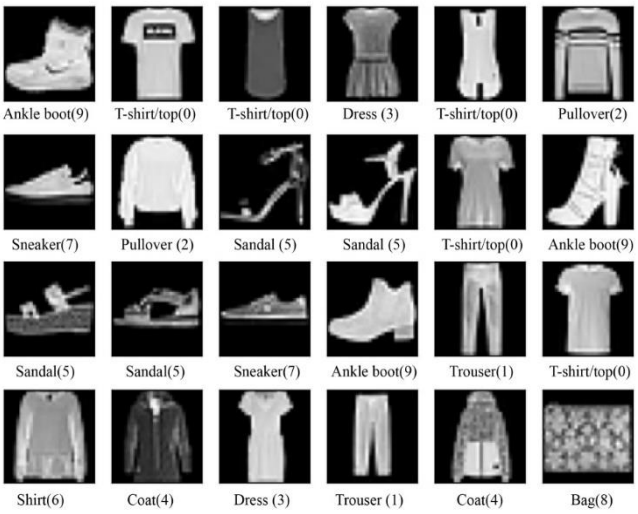
The experiments were conducted at the cyber security lab at GITAM, which is deemed to be a university in India. The hardware requirements include a Windows 11 pro for workstation, DELL Precision 7820 tower XCTO, Processor: Xeon Bronze, 3106 CPU @ 1.70 Ghz, 128 GB DDR4 RAM, ROM: 2TB PCLe NVME SSD, GPU: NVIDIA RTX A6000 48 GB. The software requirements are Spyder IDE in the Anaconda environment and several inbuilt packages of Python, sklearn, clever hands, and matplotlib for data gathering, analysis, synthetic data generation, and visualization purposes.

5.1. Data Collection

This research study conducted experiments on image recognition data, utilizing both original image samples and synthetically generated adversarial attack samples. The original image samples were obtained from two benchmark datasets: MNIST [23] and Fashion-MNIST [24]. The MNIST dataset consists of 60,000 grayscale images of handwritten digits (0-9) with a resolution of 28×28 pixels, including 50,000 training samples and 10,000 testing samples. In contrast, Fashion-MNIST contains images of various clothing items, such as skirts and shoes, categorized into ten distinct classes.



(a) MNIST images



(b) Fashion-MNIST images

Fig. 5 The image sample collection of two benchmark datasets: (a)MNIST, and (b) Fashion-MNIST datasets. Each dataset collection represents images belonging to 10 classes, along with their labels displayed below each image.

Figure 5 displays the images of (a) MNIST, and (b) Fashion-MNIST for all 10 classes, along with their class labels below each image.

5.2. Defense Model Architectures

The DLMs used in this study were custom-built, lightweight, and sequential, with varying architectural hyperparameters, including the number of convolutional and dense layers, weight initialization methods, activation functions at the input and output layers, the dimensions of input, max-pooling, and stride layers. The defense models employed sparse categorical cross-entropy loss functions and Adam optimization techniques. The detailed specifications are provided in Table 1. For the EAT-Training phase-three DNN models were employed: EAT Model 1, EAT Model 2, and EAT Model 3. The EAT-integrated Ensemble training phase utilizes the bootstrap aggregation ensembled model. The EAT-integrated Ensemble model integrated the EAT model-1, EAT model-2, and EAT-model-3 as weak learners for further training. A target DNN model is utilized for attack analysis and comparative evaluation.

Table 1. The design specifications of model architectures designed for target and defense models

Model Configuration	Target Model (DNN)	EAT Model1	EAT Model2	EAT Model3	EAT-Integrated Ensemble Model
Type	Sequential DNN	Sequential DNN	Sequential DNN	Sequential DNN	Ensembled voting classifiers
Convolutional layer(s)	1, 32 filters Size 3x3	1, 32 filters Size 3x3	1, 32 filters Size 3x3	2, 64, 32 filters Size 3x3	EAT model 1, 2, and 3 convolutions, respectively
Maxpool / Stride	1, 2x2, 1	1, 2x2,1	2, 2x2,1	2, 2x2,1	EAT model 1, 2, and 3 max pool layers, respectively
Dense layer(s)	1, 100 units	1, 100 units	3, 100 units	3, 100 units	EAT model 1, 2, and 3 dense layers, respectively
Weight initialization	he_uniform	he_uniform	he_uniform	he_uniform	he_uniform
Activation function	RELU for input layers, softmax for output layer	GELU input, softmax output	GELU input, softmax output	GELU input, softmax output	RELU, GELU input, softmax output

Table 2. Different parameters of the test datasets used for implementation and evaluation of the proposed MIA method on various defense models

Original or clean dataset name	Deep Neural Network(s)	Attack algorithm type	Perturbation size used to create an attack sample set	No. of iterations required by the attack algorithm to generate a strong attack sample set
MNIST	Target model (our's), VGG16, VGG7, ResNet32, ResNet50	FGSM, CW, PGD	2.1, 2.54, 3	1000 for CW, 10 for FGSM and PGD
FashionMNIST	Target model, VGG16, VGG7, ResNet32, ResNet50	FGSM, CW, PGD	1, 1.7, 2, 2.4	10
MNIST	Target model, VGG16, VGG7, ResNet32, ResNet50	GMA	$Tr(\epsilon) = \{0.1e^{-1}$ to $3.0e^{-6}\}$	10
FashionMNIST	Target model, VGG16, VGG7, ResNet32, ResNet50	GMA	$Tr(\epsilon) = \{0.1e^{-1}$ to $3.0e^{-6}\}$	10

MNIST	EAT model1, EAT model2, EAT model3, EAT-integrated Ensemble model (proposed)	GMA	Tr(ϵ) = {0.1e^{-1} to 3.0e^{-6}}	10
FashionMNIST	EAT model1, EAT model2, EAT model3, EAT-integrated Ensemble model (proposed)	GMA	Tr(ϵ) = {0.1e^{-1} to 3.0e^{-6}}	10
MNIST	EAT model1, EAT model2, EAT model3, EAT-integrated Ensemble model (proposed)	FGSM, CW, PGD	Tr(ϵ) = {0.1e^{-1} to 3.0e^{-6}}, 1, 2, 3	1000 for CW, 10 for GMA, FGSM and PGD
FashionMNIST	EAT model1, EAT model2, EAT model3, EAT-integrated Ensemble model (proposed)	FGSM, CW, PGD	Tr(ϵ) = {0.1e^{-1} to 3.0e^{-6}}, 1, 2, 3	1000 for CW, 10 for GMA, FGSM and PGD

Table 2 presents the list of the datasets, deep neural network architectures, and input parameters used in the experimental study. The images of MMIST and Fashion-MNIST datasets were utilized in their original form for training on clean data with a train-test split ratio of 70:30. The adversarial images of MNIST and Fashion-MNIST were generated for PGD, FGSM, and CW using their respective attack algorithms, sourced from open-source repositories [48, 50]. The GMA datasets were created using the method described in algorithm 1. For each clean dataset, adversarial samples were generated for GMA, PGD, FGSM, and CW attacks, resulting in four distinct attack datasets per clean dataset and attack type.

The first four rows of Table 2 outline the parameters used for attack analysis, assessing the efficacy of the GMA attack and evaluating the robustness of various defense models. The remaining four rows detail the parameters for training the EAT and MIA defense models. The target model DNN is utilized for performance analysis before and after adversarial attacks.

The EAT model-1, EAT model-2, and EAT model-3 are trained on adversarial samples generated using PGD, FGSM, CW, and GMA attacks. The MIA defense model further enhances performance by improving the detection and mitigation of these adversarial attacks. Additionally, state-of-the-art defense models, including VGG7 [35], VGG16 [41], ResNet32 [33], and ResNet50 [40], are incorporated into the study to evaluate their robustness against GMA attacks and to conduct comparative and ablation studies.

5.3. Method Evaluation

The MIA method is evaluated on the test datasets containing legitimate and adversarial attack samples. The f1 score and model accuracy are the most commonly used quality metrics that quantify the precision and robustness of a classifier and are used here to assess the performance of the proposed MIA method. The f1 score depends on precision and recall metrics, with precision measuring the accuracy of a model's predictions. A higher f1 score and model accuracy enhance performance. The formulae of precision, recall, f1 score, and accuracy are shown in Equation 7 to Equation 10, extracted from [31].

$$Precision = \frac{TP}{(TP+FP)} \tag{7}$$

$$Recall = \frac{TP}{(TP+FN)} \tag{8}$$

$$f1 = 2 * \frac{1}{((1/precision)+(1/recall))} \tag{9}$$

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ samples} \tag{10}$$

6. Results and Discussion

The GMA perturbations are significantly smaller than those used in conventional attack methods. Consequently, 10 iterations were required to generate adversarial samples that remain visually similar to the original images while effectively compromising the target model. The influence of the GMA attack is assessed with visual and attack analysis.

6.1. GMA Visual Analysis Results

Figure 6 shows the results of the visual analysis conducted to assess the visual similarity between the clean images of MNIST and Fashion-MNIST and the images resulting from GMA, PGD, FGSM, and CW attacks. The first row displays the original (clean) images from the MNIST and Fashion-MNIST datasets. The subsequent rows illustrate the perturbed images generated by GMA, PGD, FGSM, and CW attacks at different perturbation levels. The results demonstrate that the original and GMA-perturbed images appear strikingly similar across all perturbation levels of ϵ, making them indistinguishable by the human observers and target models. In contrast, adversarial images generated by PGD, FGSM, and CW attacks exhibit slight visual differences from the original images, as they utilize larger perturbation sizes. These findings confirm that the similarity between original and adversarial images is higher when the perturbation size (ϵ) is small. Despite their subtle nature, these perturbations remain highly effective in deceiving the target models.

6.2. GMA Attack Analysis Results

Figure 7 to Figure 10 shows the attack analysis results conducted on various DLMs, as outlined in the experimental

setup section of this paper. The defense models-VGG16, VGG7, ResNet32, and ResNet50-were trained on PGD, FGSM, and CW attacks, whereas the target model DNN was not trained on adversarial attacks. Figure 7 displays the impact of various attacks on the DLMs under Normal (B_{acc}) and attack conditions were tested for the MNIST dataset. The B_{acc} values of VGG16, VGG7, ResNet32, ResNet50, and target DNN show over 80% accuracy, indicating their well-trained performance for MNIST image classification under normal conditions. However, when subjected to the GMA attack, their performances deteriorate drastically, resulting in the following worst-case model accuracies ($Fool(M)_{acc}$): VGG16: 9.90%, VGG7: 5.30%, ResNet32: 5.06%, ResNet50: 9.31%, and Target DNN: 0.62%.

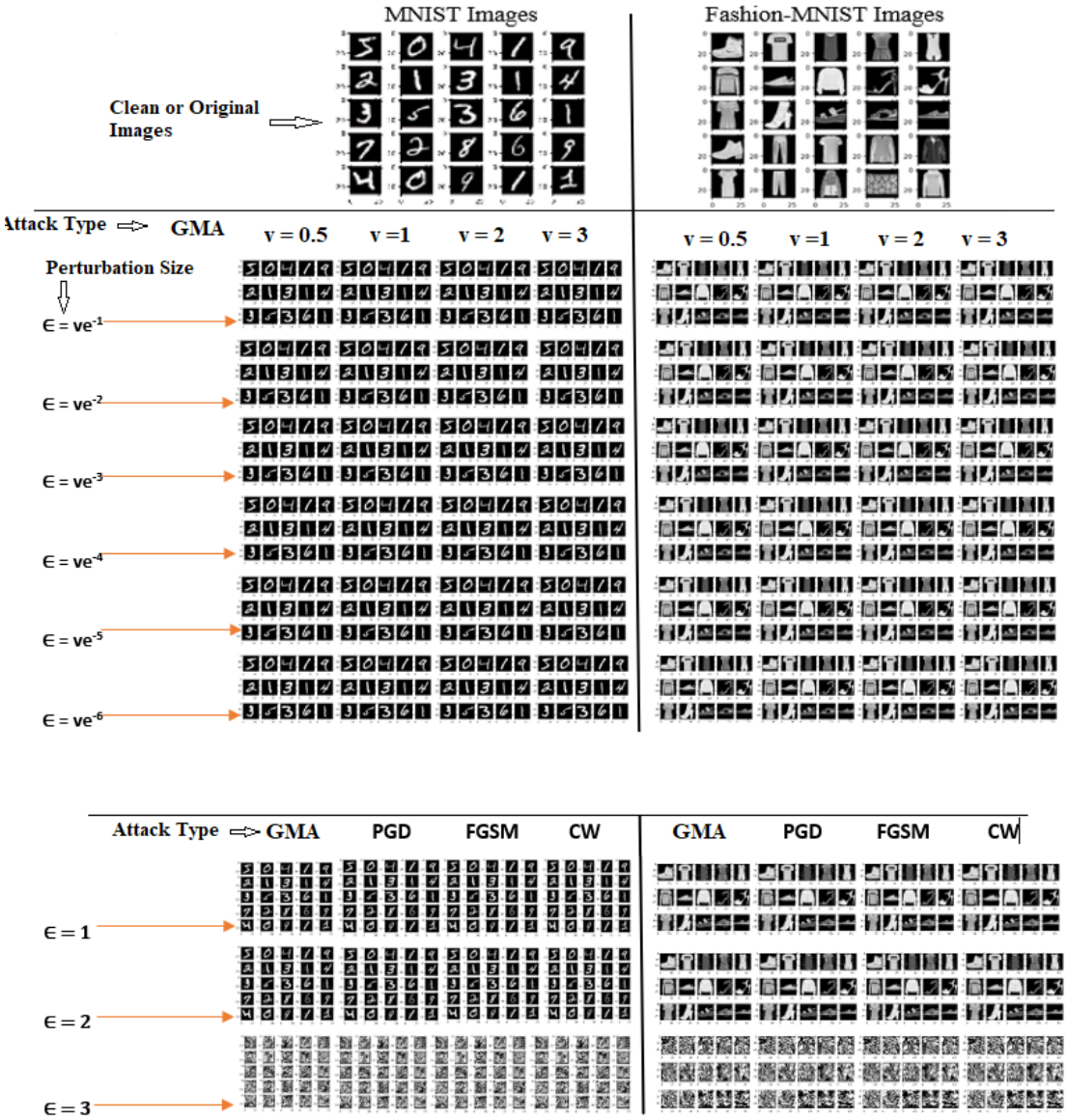


Fig. 6 The images of original and various attack samples of MNIST and Fashion-MNIST datasets presented for visual analysis

The results demonstrate that the GMA attack, with its truncated perturbations, effectively manipulates state-of-the-art defense models, leading to severe performance degradation. This confirms that the GMA attack exerts the highest influence on all targeted DLMs, making it a highly effective adversarial attack. In contrast, the other attacks-PGD, FGSM, and CW-have a minimal effect on the defense models VGG16, VGG7, ResNet32, and ResNet50, yet maintain best-case accuracies. This is likely because these models were specifically trained to defend against PGD, FGSM, and CW attacks, allowing them to perform well against them. The DNN, which has not been trained on any of the attacks, has significantly affected its performance to all the performed attacks, resulting in degraded accuracies of 1.92%

for PGD, 2.70% for FGSM and 1% for CW, respectively. Figure 8 shows the impact of various adversarial attacks on DLMs tested for the Fashion-MNIST dataset. Under normal conditions, the baseline accuracies of VGG16, VGG7, ResNet32, ResNet50, and a standard DNN are 98.14%, 97%, 98.05%, 99.40%, and 99.30%, respectively.

However, when subjected to a GMA attack, these models experience a substantial decline in accuracy, dropping to 8.90%, 7.50%, 6%, 9.66%, and 2.07%, respectively. This demonstrates that the GMA attack exerts the most significant influence on all targeted DLMs. Nevertheless, VGG16, VGG7, ResNet32, and ResNet50 exhibit relative robustness against other adversarial attacks, including PGD, FGSM, and CW.

Notably, an unprotected target model-one not trained on adversarial perturbations-suffers drastic accuracy reductions to 2.07%, 5.92%, 3.77%, and 14.20% when exposed to these attacks on the Fashion-MNIST dataset. The attack success rates comparison of different adversarial attacks on the targeted DNNs (VGG16, VGG7, ResNet32, ResNet50, and DNN) are shown in Figure 9, offering insight into the relative influence of the GMA attack on these DLMs.

The GMA attack is the most influential, with an average success rate of 91.59% on the MNIST dataset and 97.86% on the FashionMNIST dataset, outperforming the other attacks. The GMA attack samples with truncated perturbations are sufficiently strong to deceive all defense models, meeting the conditions for a successful attack.

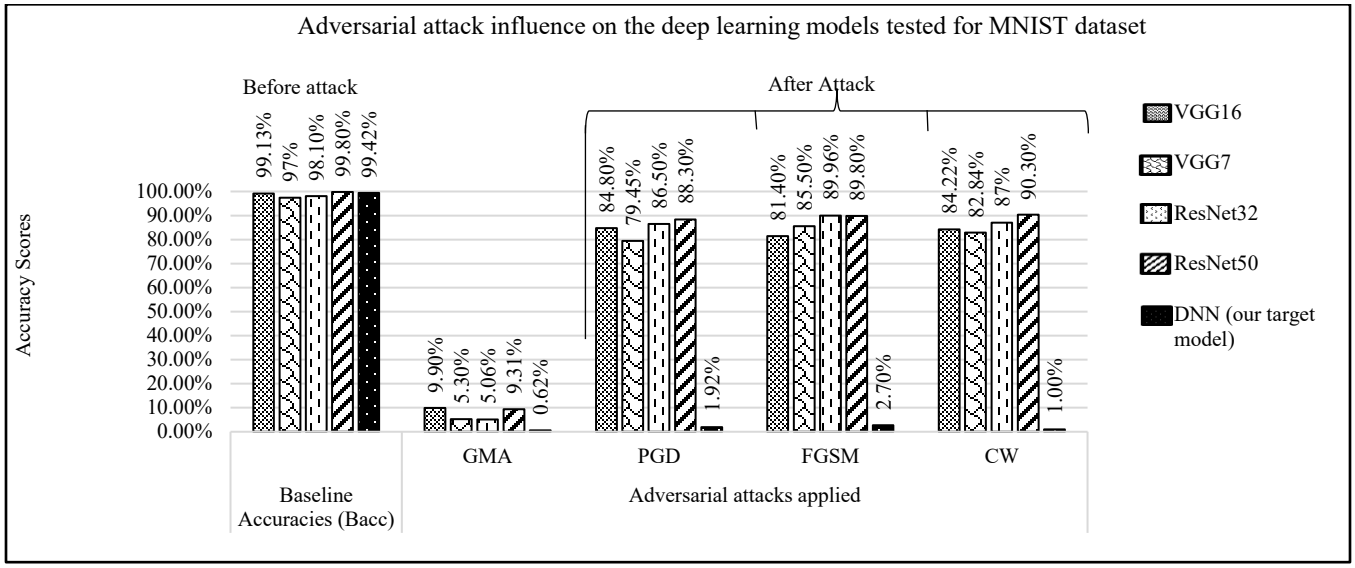


Fig. 7 The performance comparison of various DLMs before (B_{acc}) and after adversarial attacks (GMA, PGD, FGSM and CW) using the MNIST dataset

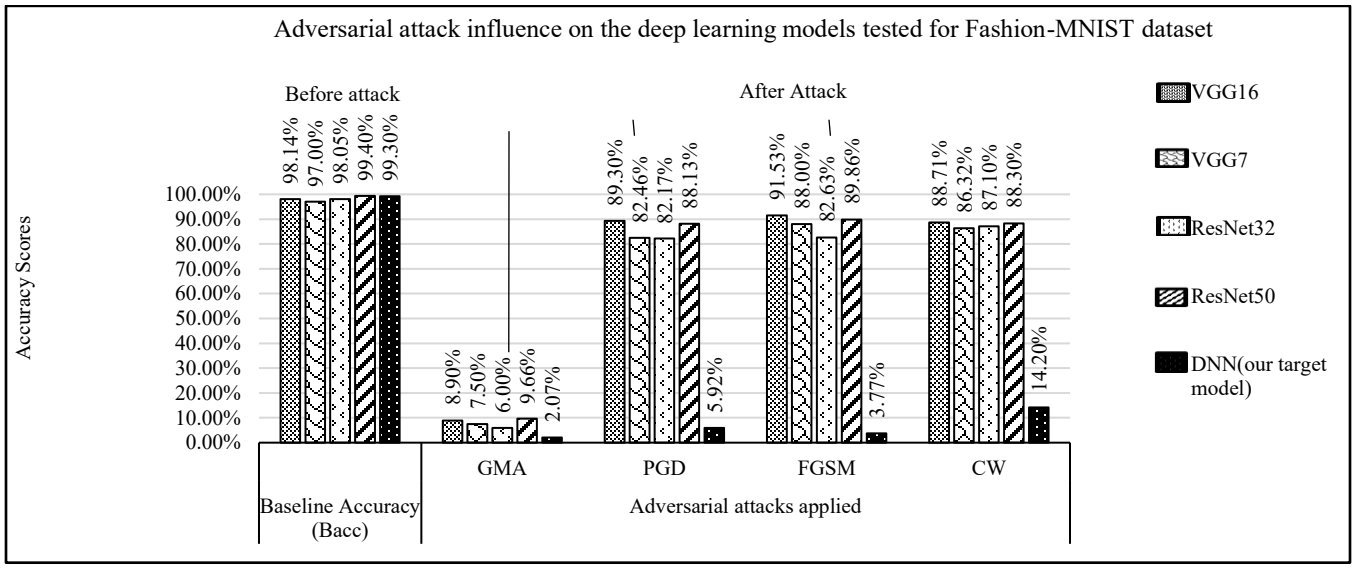
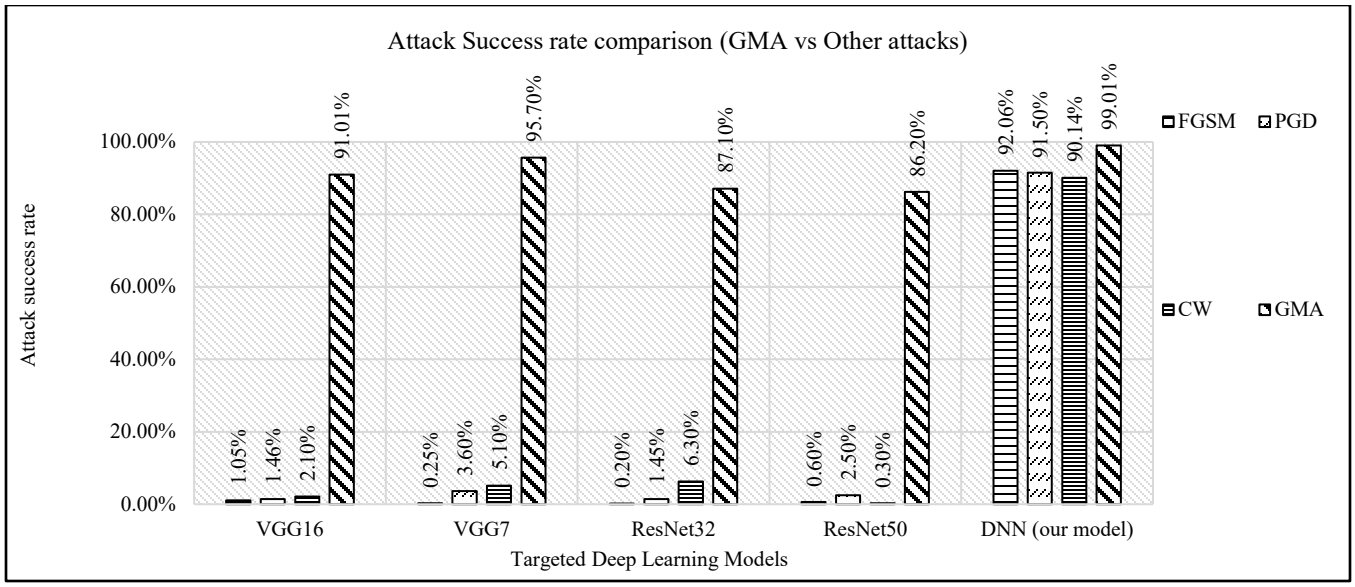
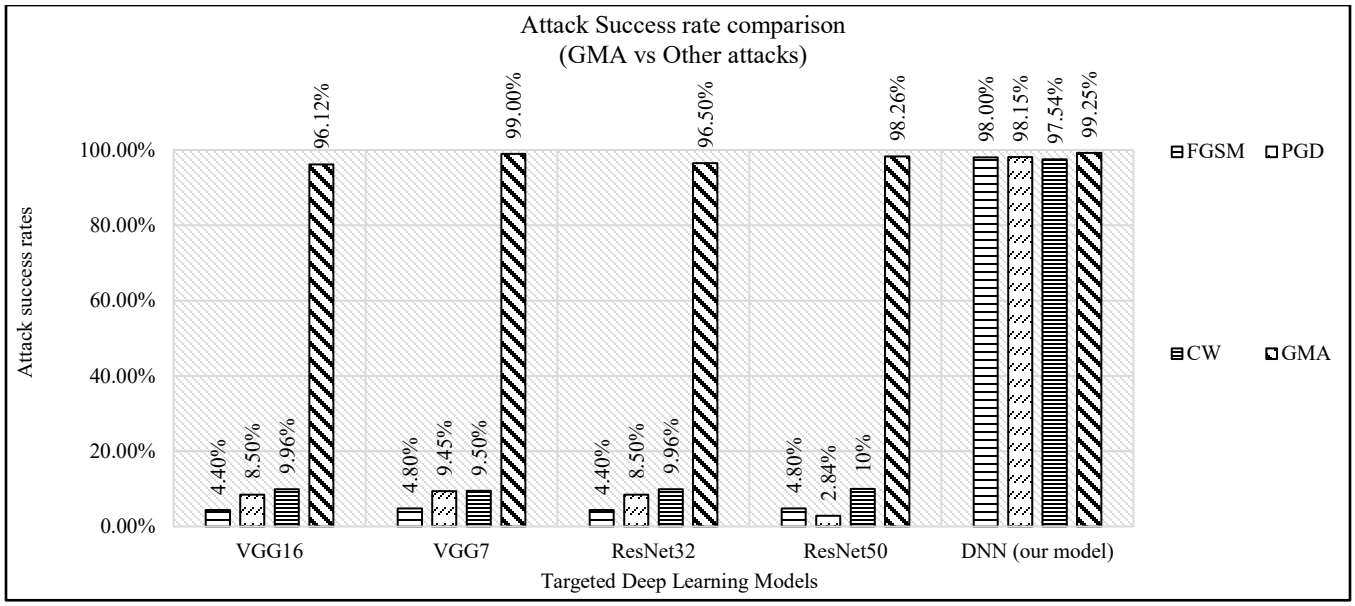


Fig. 8 The performance comparison of various DLMs before (B_{acc}) and after adversarial attacks (GMA, PGD, FGSM and CW) using the Fashion-MNIST dataset



(a) MNIST



(b) Fashion-MNIST

Fig. 9 Attack success rates of various adversarial attacks on tested DLMs using, (a) MNIST, and (b) Fashion-MNIST datasets.

Figure 10 presents a sample of classification results produced by the DLMs under both normal and GMA attack conditions.

The figure is divided into two sections: (1) the upper section, which displays classification outcomes for the MNIST dataset under (a) normal conditions and (b) GMA attack conditions, and (2) the lower section, which illustrates classification outcomes for the Fashion-MNIST dataset under (a) normal and (b) attack conditions. Under normal conditions, the DNNs accurately classified MNIST and Fashion-MNIST images, with predicted labels matching their corresponding ground truth labels. However, after exposure to the GMA attack, the models produced erroneous classifications. For instance, a handwritten digit '5' was misclassified as '7,' and a Fashion-MNIST image labelled 'Top' was misclassified as 'Pullover.' Despite its high success rate, the GMA attack is not entirely manipulative, as specific images, such as 'Ankle Boot' and 'Digit 1,' were still correctly classified. Nevertheless, GMA remains more sophisticated than standard adversarial attacks, achieving the highest attack success rates. The results presented in this section highlight that the vulnerability of models to GMA attacks has been proven via visual and attack analysis results. Also, this phenomenon underscores the necessity of robust defense models to defend against many sophisticated adversarial attacks.

(a) Normal **(b) Under GMA attack**

Fig. 10 A sample classification results produced by the targeted DNNs to GMA adversarial images of MNIST and Fashion-MNIST images

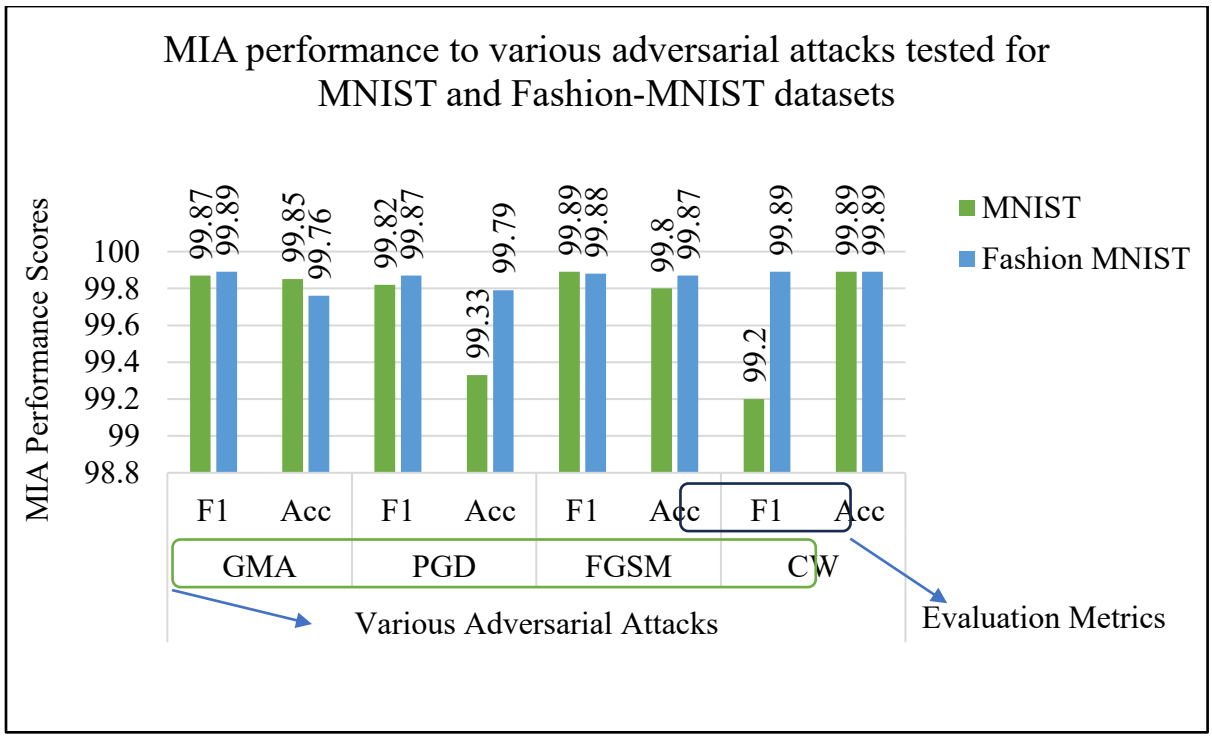


Fig. 11 The evaluation metrics f1 and recall scores plotted to visualize the MIA performance in detecting GMA, PGD, FGSM and CW attacks using MNIST and Fashion-MNIST datasets

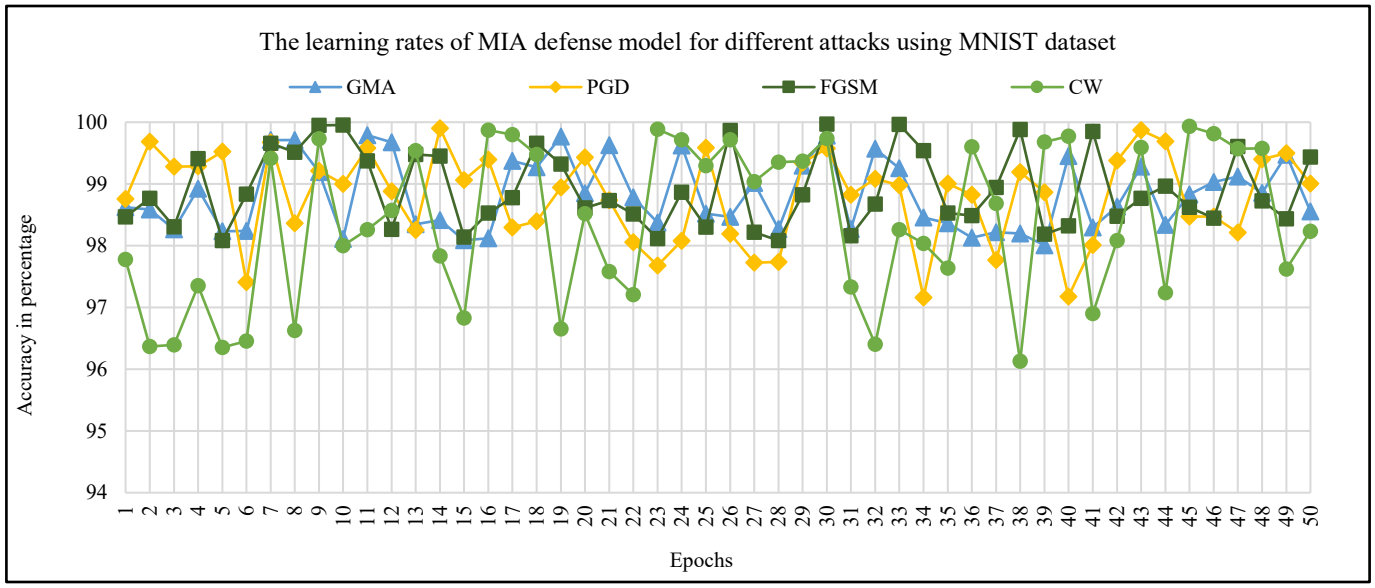


Fig. 12 The learning curves of the MIA defense model captured for 50 epochs while training different adversarial attacks GMA, PGD, FGSM and CW using MNIST data

6.3. MIA Performance Results

Table 3 summarizes the attack detection efficiency of the MIA method. For the MNIST dataset, the model achieves detection accuracies of 99.85%, 99.33%, 99.80%, and 99.89% against GMA, PGD, FGSM, and CW attacks, respectively. Similarly, the Fashion-MNIST dataset attains detection accuracies of 99.76%, 99.79%, 99.87%, and 99.89% for the same attacks. Figure 11 represents the MIA performance scores presented in Table 3, tested for various adversarial attacks. The learning and loss curves of the MIA defense model over 50 training epochs to train for GMA, PGD, FGSM, and CW attacks using MNIST and Fashion-MNIST datasets are shown from Figure 12 to Figure 15. Figure 12 and Figure 14 show the learning curves of the MIA defense model trained for GMA, PGD, FGSM and CW attacks using MNIST and Fashion-MNIST datasets, respectively.

Figure 13 and Figure 15 show the loss curves of the MIA defense model for training GMA, PGD, FGSM and CW attack using MNIST and Fashion-MNIST, respectively. The MIA defense model converges at 50 epochs, achieving optimal performance with minimal loss. The MIA method's attack detection capability and parameter efficiency are compared with several state-of-the-art approaches in Table 4. Unlike other methods that require an additional model for detection, the MIA method independently detects adversarial attacks without needing supplementary models. This advantage is attributed to MIA's unique training methodology.

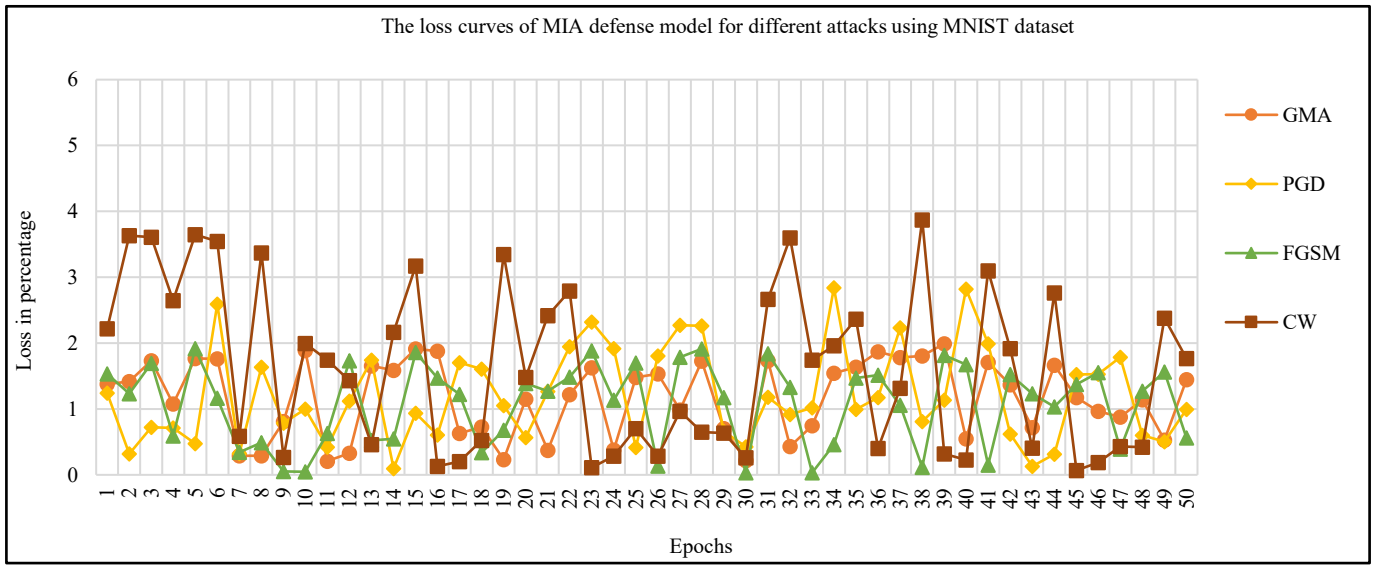


Fig. 13 The MIA defense model loss curves for GMA, PGD, FGSM and CW attacks using MNIST data

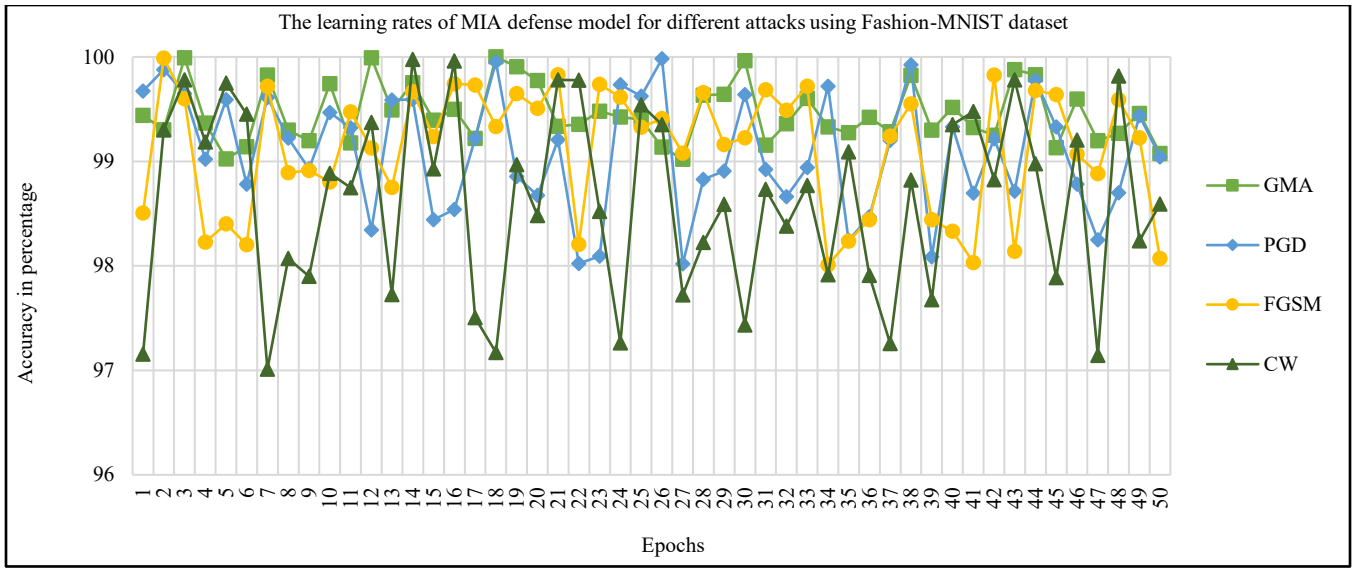


Fig. 14 The learning curves of the MIA defense model captured for 50 epochs while training different adversarial attacks GMA, PGD, FGSM and CW using Fashion-MNIST data

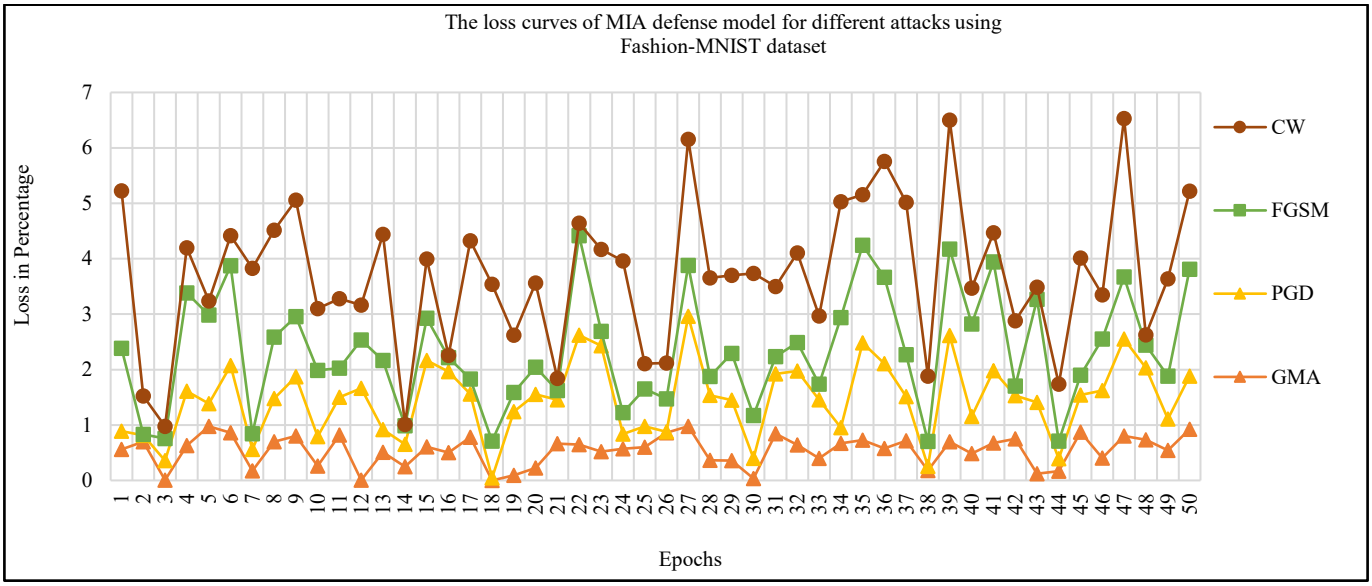


Fig. 15 The MIA defense model loss curves for GMA, PGD, FGSM and CW attacks using Fashion-MNIST data

Furthermore, the MIA method reduces the number of attack samples required for training to 40%, whereas other methods typically require 50%. This 10% reduction is significant, as generating adversarial samples is time-consuming, taking approximately 24 to 48 hours, depending on the attack type and perturbations applied [29]. The MIA achieved this 10% reduction because of the decision to choose the appropriate proportion of adversarial examples added to the dataset, where the EAT training model also exhibits optimal performance. According to EAT-training results shown in Table 5 and Table 6, the defense models exhibit >80% of best-case accuracy after adding 40% (training cycle-3) and 50% (training cycle-4) proportion of adversarial examples in the augmented dataset. Note that there were no significant differences in the detection accuracy between training cycle 3 and training cycle 4, but there is a significant difference in the requirement of generating adversarial samples. Hence, the MIA defense model is trained with 40% of adversarial examples. Thus, MIA offers an efficient training strategy by minimizing the need for extensive attack sample generation by 10%. The MIA defense model also employs a simple five-layer architecture, unlike the more complex architectures used by other methods. Despite its simplicity, it effectively detects GMA attacks alongside PGD, FGSM, and CW attacks. Notably, while existing methods fail to detect GMA and instead fall victim to it, as previously discussed, the MIA defense model successfully detects all tested attacks, achieving an average detection accuracy of 99.71%. The

performance scores (f1 and accuracy) of EAT Model 1, EAT Model 2, and EAT Model 3 are presented in Table 5 for the augmented MNIST dataset and Table 6 for the augmented Fashion-MNIST dataset. Each EAT model undergoes four training cycles:

- Train Cycle 1: 15% of attack samples in the training set
- Train Cycle 2: 25% of attack samples in the training set
- Train Cycle 3: 40% of attack samples in the training set
- Train Cycle 4: 50% of attack samples in the training set

Table 3. The detection accuracy of our MIA defense model was tested for Various adversarial attacks

Dataset Name	Adversarial Attacks							
	GMA		PGD		FGSM		CW	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc
MNIST	99.87	99.85	99.82	99.33	99.89	99.80	99.20	99.89
Fashion MNIST	99.89	99.76	99.87	99.79	99.88	99.87	99.89	99.89

Table 4. Performance comparison of state-of-art methods and proposed MIA method

Training Resources	Defense Methods				
Method Name	Bilateral adversarial Training [44]	Fast Adversarial Training [36]	Convolutional Filter statistics [34]	GAN based training [42]	MIA (The Proposed Metod)
Model Name	VGG16	VGG7	ResNet32	ResNet50	MIA defense model
Architecture	16 layered	7 layered	32 layered	50 layered	5 layered
Minimum Attack samples required to train	50%	50%	50%	50%	**40%**
Additional Model Required?	✓	✓	✓	✓	**X**
FGSM attack detected?	✓	✓	✓	✓	✓
PGD attack detected?	✓	✓	✓	✓	✓
CW attack detected?	✓	✓	✓	✓	✓
GMA attack detected?	**X**	**X**	**X**	**X**	✓
Model accuracy achieved on GMA attack	3.87%	5%	5.25%	4.11%	**99.71%**

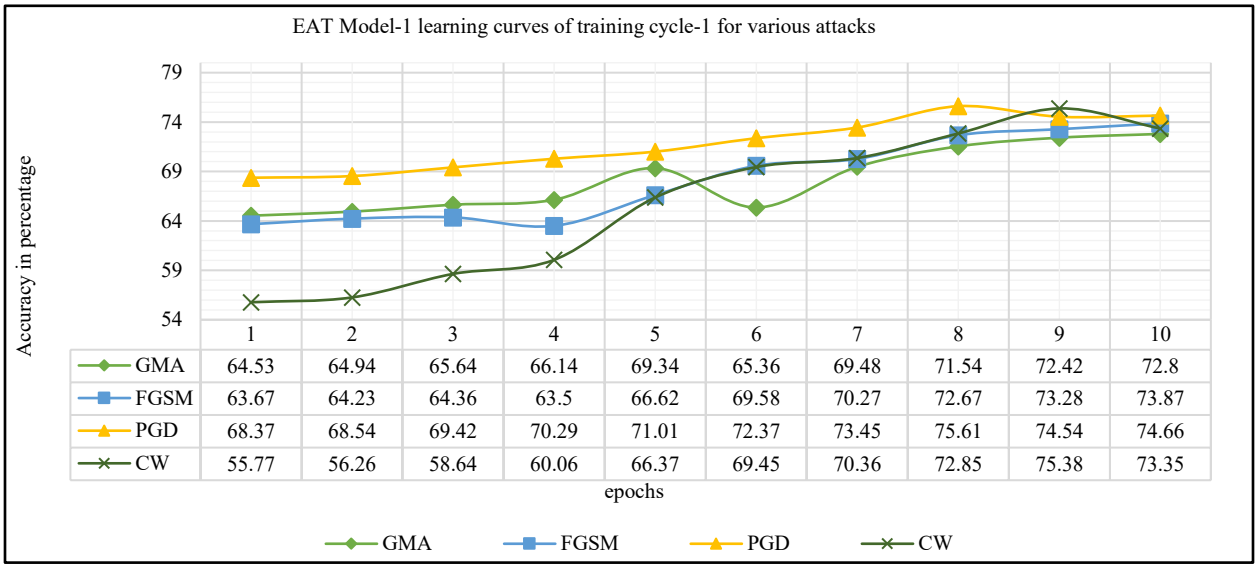
Table 5 and Table 6 show that model performance improves as the proportion of attack samples in the training set increases. All EAT models performed best during the third and fourth training cycles. However, the accuracy gain between these cycles is only 1%, despite the additional 10% attack samples required for training. Given this marginal improvement, we select the augmented dataset with 40% attack samples and a 10% reduction in adversarial example generation. The augmented dataset with a 40% proportion is chosen for EAT-integrated ensemble training. The learning and loss curves of each EAT model for training cycles 1 and 3 are illustrated in Figure 16 to Figure 23, providing further insights into model performance across different training stages.

Table 5. The accuracy and f1 scores of our defense models (in %) on different adversarial attacks for augmented MNIST dataset

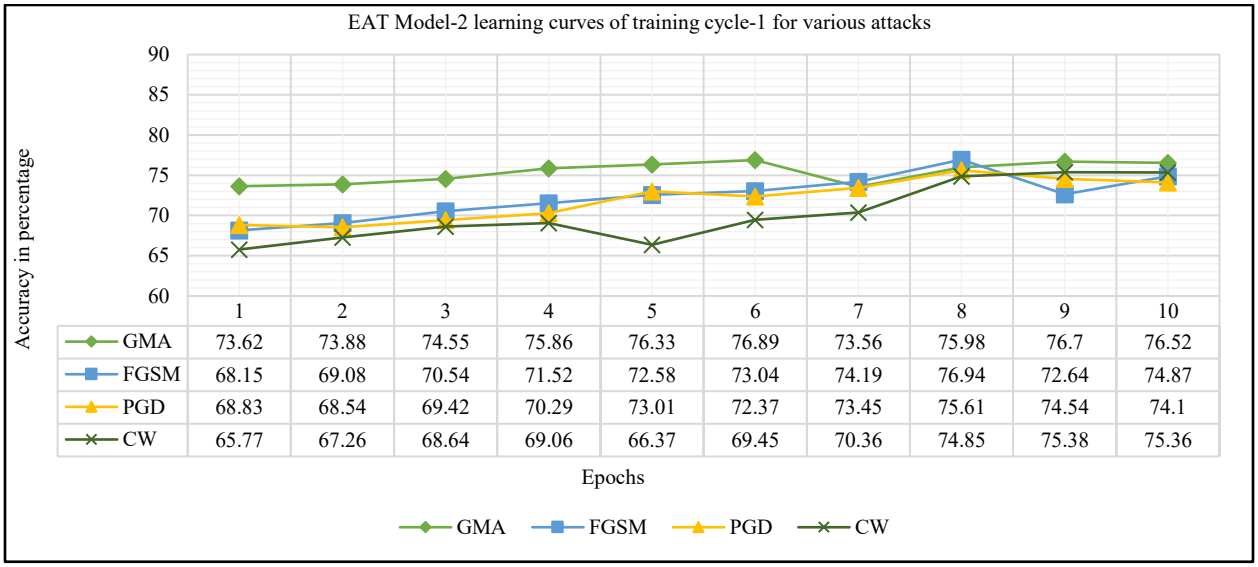
EAT Model - Training Cycle		GMA		PGD		FGSM		CW	
		F1	Acc	F1	Acc	F1	Acc	F1	Acc
EAT model 1	Train Cycle -1 (15%)	72.80	72.00	74.66	74.63	73.87	73.54	72.55	73.35
	Train Cycle-2 (25%)	74.62	73.89	73.77	73.67	74.26	74.68	74.27	74.17
	Train Cycle-3 (40%)	81.57	81.66	80.71	80.65	82.60	82.56	82.10	81.61
	Train Cycle-4 (50%)	82.68	81.67	81.24	80.54	83.24	82.64	83.42	82.12
EAT model 2	Train Cycle -1 (15%)	75.60	76.52	74.16	74.10	74.26	74.87	75.13	75.36
	Train Cycle-2 (25%)	77.15	78.75	75.66	75.26	74.17	74.77	78.63	78.10
	Train Cycle-3 (40%)	87.25	87.87	85.78	85.68	86.00	87.20	87.65	86.87
	Train Cycle-4 (50%)	87.51	87.88	88.27	88.11	88.43	88.30	87.85	87.87
EAT model 3	Train Cycle -1 (15%)	76.10	77.56	77.45	77.17	77.15	77.00	76.25	76.15
	Train Cycle-2 (25%)	76.66	78.10	79.42	76.57	78.72	78.52	77.62	76.92
	Train Cycle-3 (40%)	95.82	95.80	94.67	94.87	94.23	94.30	95.53	95.66
	Train Cycle-4 (50%)	98.69	98.55	96.38	96.17	96.00	95.26	95.64	96.27

Table 6. The accuracy and f1 scores of eat-trained models at four training cycles on different adversarial attacks using an augmented Fashion-MNIST dataset

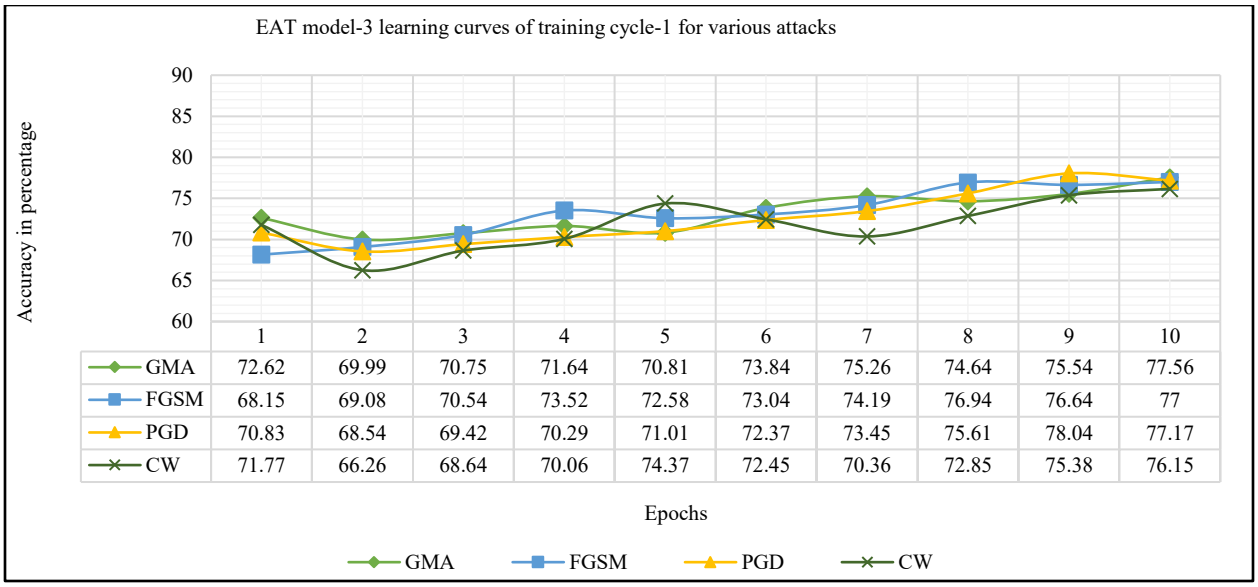
EAT model - Training Cycle		GMA		PGD		FGSM		CW	
		F1	Acc	F1	Acc	F1	Acc	F1	Acc
EAT model 1	Train Cycle -1 (15%)	71.30	71.21	74.66	74.63	73.87	73.54	72.55	73.35
	Train Cycle-2 (25%)	73.67	73.64	73.71	73.65	73.25	73.45	73.25	73.15
	Train Cycle-3 (40%)	83.28	83.26	83.27	83.25	83.65	83.53	83.29	83.26
	Train Cycle-4 (50%)	84.68	84.56	84.42	84.12	84.32	84.20	84.50	84.35
EAT model 2	Train Cycle -1 (15%)	76.86	76.26	76.53	76.12	76.47	76.27	76.39	76.35
	Train Cycle-2 (25%)	77.65	77.35	76.50	76.45	76.78	76.77	76.37	76.17
	Train Cycle-3 (40%)	88.56	88.18	88.78	85.68	86.00	87.20	87.65	86.87
	Train Cycle-4 (50%)	89.35	89.28	89.15	89.10	89.73	89.50	89.68	89.53
EAT model 3	Train Cycle -1 (15%)	76.60	76.52	76.74	76.62	76.57	76.34	76.33	76.28
	Train Cycle-2 (25%)	77.83	77.64	77.71	77.39	77.58	77.36	77.95	77.87
	Train Cycle-3 (40%)	96.54	96.32	96.66	96.34	96.30	96.56	96.23	96.29
	Train Cycle-4 (50%)	97.93	97.38	97.74	97.67	97.99	97.67	97.86	97.63



(a) Eat model-1

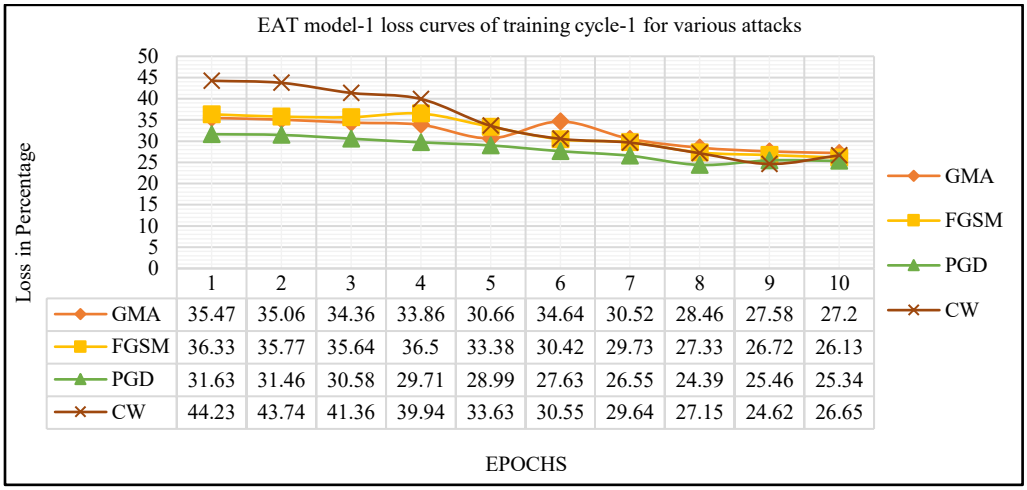


(b) Eat model-2

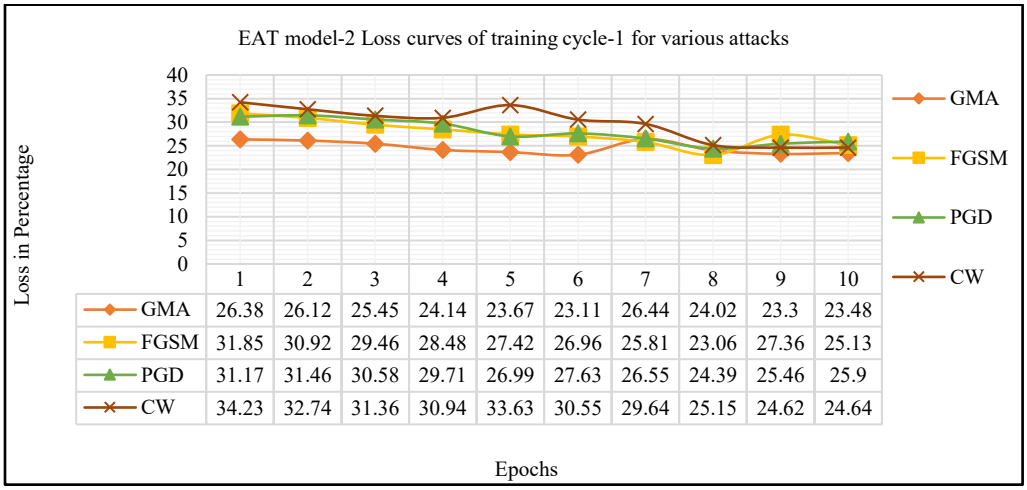


(c) Eat model-3

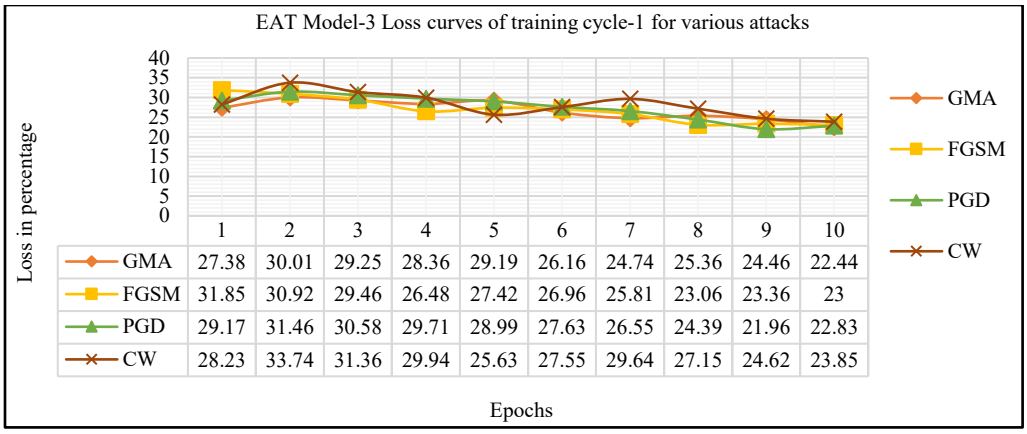
Fig. 16 The learning curves of the EAT-trained models for the first cycle (i.e., 15% of adversarial examples augmented to trainset) of the EAT process using MNIST image data



(a) EAT model-1



(b) EAT model-2



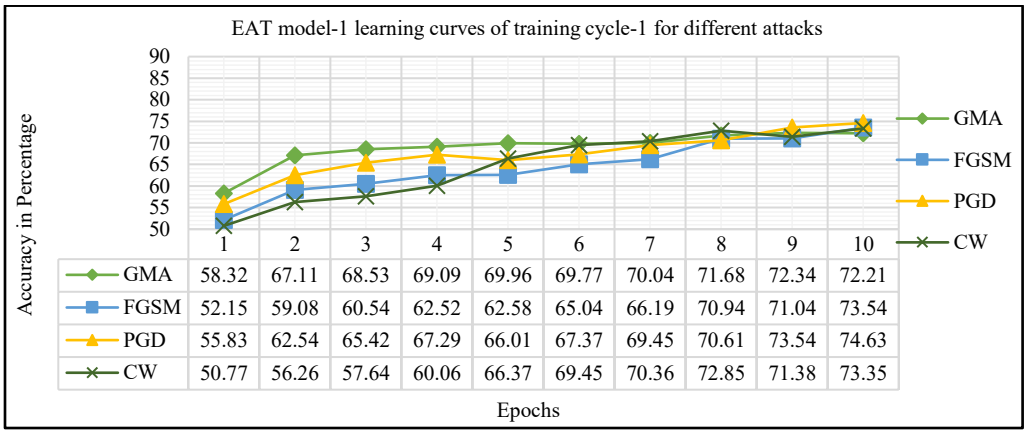
	1	2	3	4	5	6	7	8	9	10
GMA	27.38	30.01	29.25	28.36	29.19	26.16	24.74	25.36	24.46	22.44
FGSM	31.85	30.92	29.46	26.48	27.42	26.96	25.81	23.06	23.36	23
PGD	29.17	31.46	30.58	29.71	28.99	27.63	26.55	24.39	21.96	22.83
CW	28.23	33.74	31.36	29.94	25.63	27.55	29.64	27.15	24.62	23.85

(c) EAT model-3

Fig. 17 The loss curves of EAT-trained models for the first cycle using MNIST image data

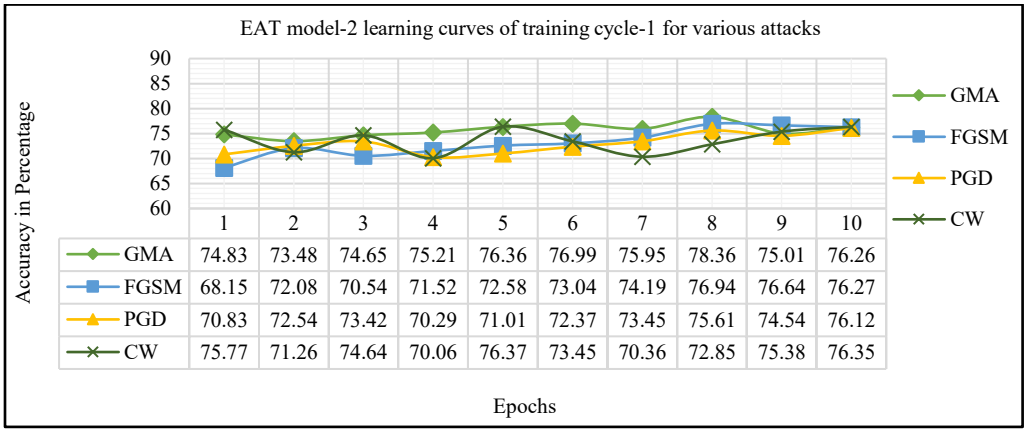
The performance of EAT Model 1, EAT Model 2, and EAT Model 3 during Training Cycle 1 for MNIST and Fashion-MNIST is presented in Figure 16 and Figure 18, respectively. Similarly, their performance during Training Cycle 3 for MNIST and Fashion-MNIST is shown in Figure 20 and Figure 22. The corresponding loss plots for EAT Model 1, EAT Model 2, and EAT Model 3 during Training Cycle 1 for MNIST and Fashion-MNIST are illustrated in Figure 17 and Figure 19, respectively.

At the same time, those for Training Cycle 3 are depicted in Figure 21 and Figure 23. These plots demonstrate a gradual improvement in classification accuracy for each EAT model, with performance peaking at the convergence point. All three EAT models converge after 10 epochs, with EAT Model 3 achieving the best performance distinguishing between benign and perturbed images. MIA is a unique training method that builds a robust defense model against GMA, PGD, FGSM, and CW attacks.



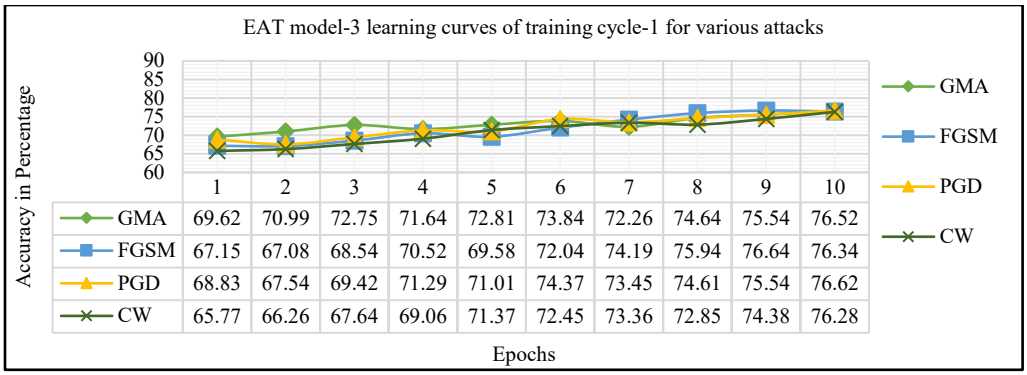
	1	2	3	4	5	6	7	8	9	10
GMA	58.32	67.11	68.53	69.09	69.96	69.77	70.04	71.68	72.34	72.21
FGSM	52.15	59.08	60.54	62.52	62.58	65.04	66.19	70.94	71.04	73.54
PGD	55.83	62.54	65.42	67.29	66.01	67.37	69.45	70.61	73.54	74.63
CW	50.77	56.26	57.64	60.06	66.37	69.45	70.36	72.85	71.38	73.35

(a) EAT model-1



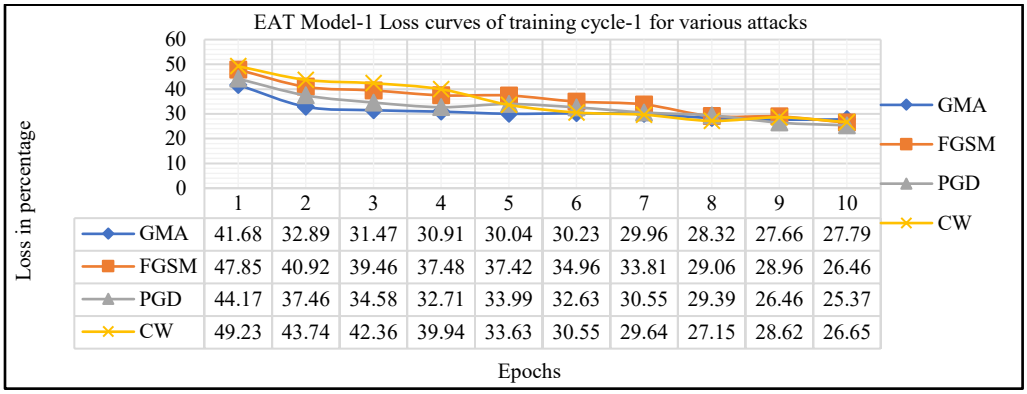
	1	2	3	4	5	6	7	8	9	10
GMA	74.83	73.48	74.65	75.21	76.36	76.99	75.95	78.36	75.01	76.26
FGSM	68.15	72.08	70.54	71.52	72.58	73.04	74.19	76.94	76.64	76.27
PGD	70.83	72.54	73.42	70.29	71.01	72.37	73.45	75.61	74.54	76.12
CW	75.77	71.26	74.64	70.06	76.37	73.45	70.36	72.85	75.38	76.35

(b) EAT model-2

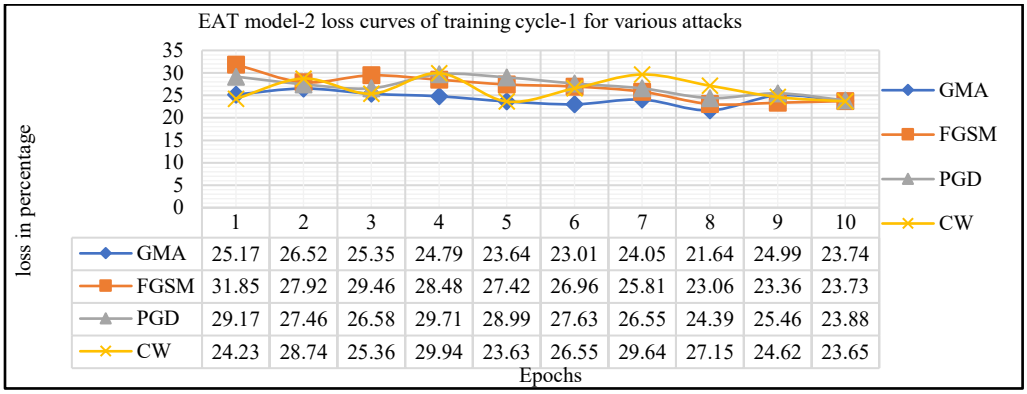


(c) EAT model-3

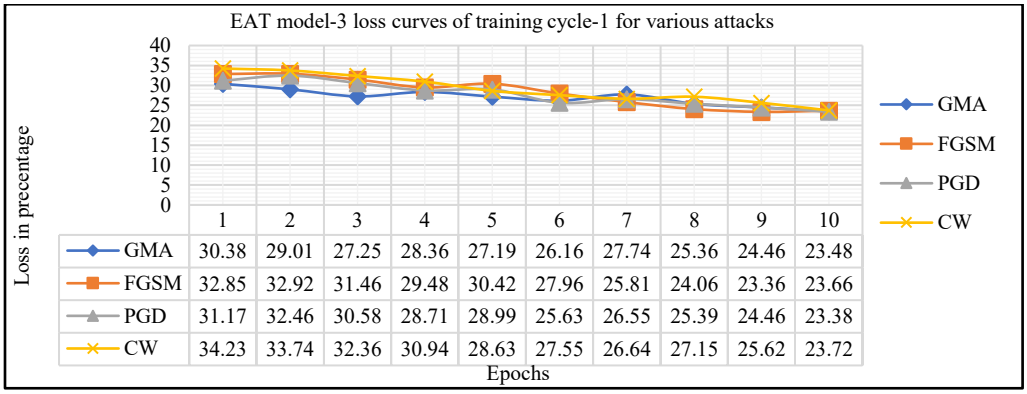
Fig. 18 The learning curves of the EAT models for the first cycle using Fashion-MNIST image data



(a) EAT model-1

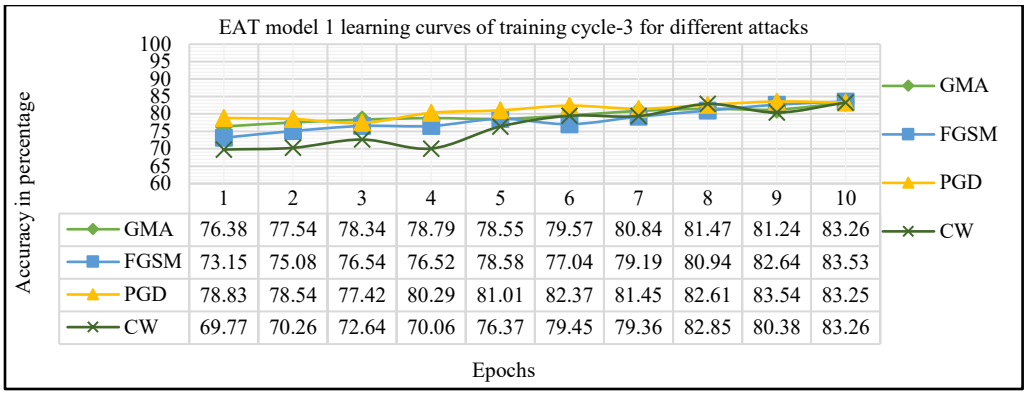


(b) EAT model-2



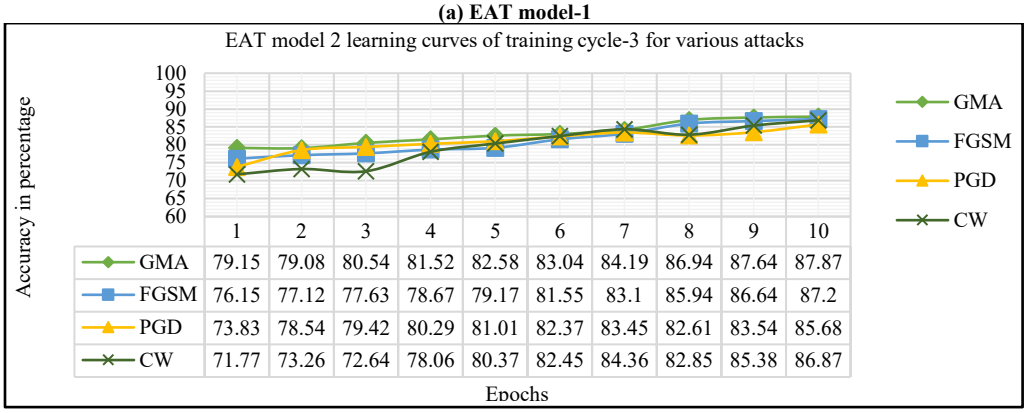
(c) EAT model-3

Fig. 19 The loss curves of EAT models for the first cycle using Fashion-MNIST image data



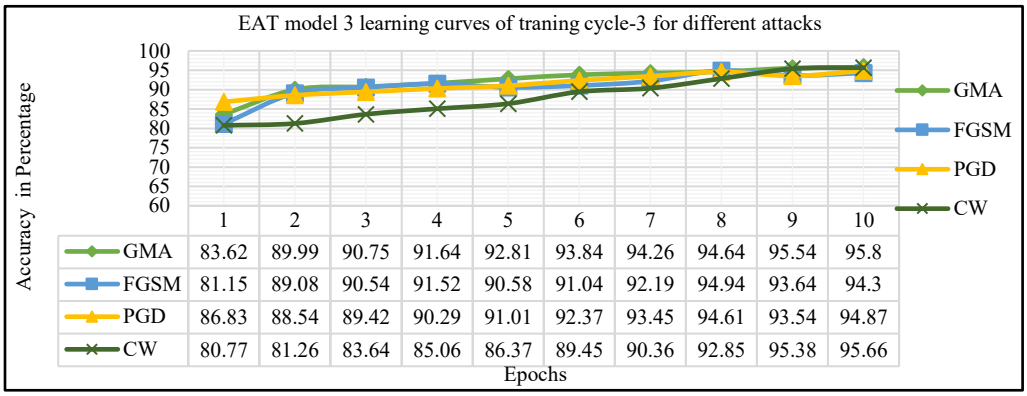
	1	2	3	4	5	6	7	8	9	10
GMA	76.38	77.54	78.34	78.79	78.55	79.57	80.84	81.47	81.24	83.26
FGSM	73.15	75.08	76.54	76.52	78.58	77.04	79.19	80.94	82.64	83.53
PGD	78.83	78.54	77.42	80.29	81.01	82.37	81.45	82.61	83.54	83.25
CW	69.77	70.26	72.64	70.06	76.37	79.45	79.36	82.85	80.38	83.26

(a) EAT model-1



	1	2	3	4	5	6	7	8	9	10
GMA	79.15	79.08	80.54	81.52	82.58	83.04	84.19	86.94	87.64	87.87
FGSM	76.15	77.12	77.63	78.67	79.17	81.55	83.1	85.94	86.64	87.2
PGD	73.83	78.54	79.42	80.29	81.01	82.37	83.45	82.61	83.54	85.68
CW	71.77	73.26	72.64	78.06	80.37	82.45	84.36	82.85	85.38	86.87

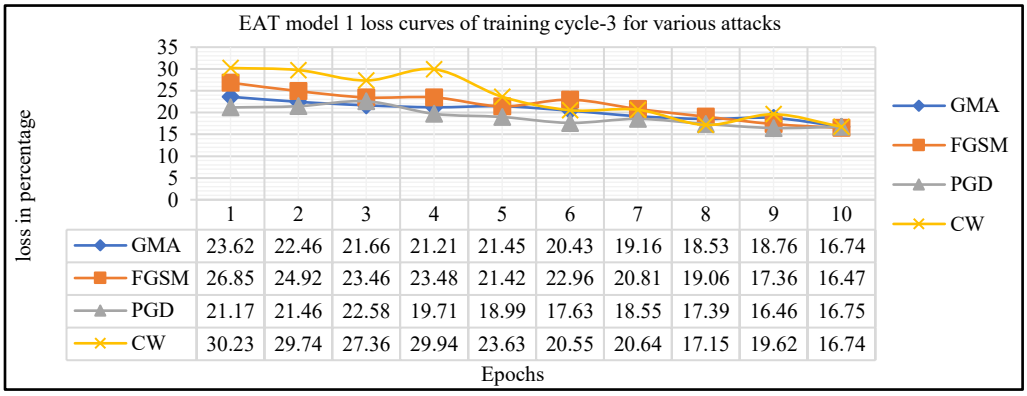
(b) EAT model-2



	1	2	3	4	5	6	7	8	9	10
GMA	83.62	89.99	90.75	91.64	92.81	93.84	94.26	94.64	95.54	95.8
FGSM	81.15	89.08	90.54	91.52	90.58	91.04	92.19	94.94	93.64	94.3
PGD	86.83	88.54	89.42	90.29	91.01	92.37	93.45	94.61	93.54	94.87
CW	80.77	81.26	83.64	85.06	86.37	89.45	90.36	92.85	95.38	95.66

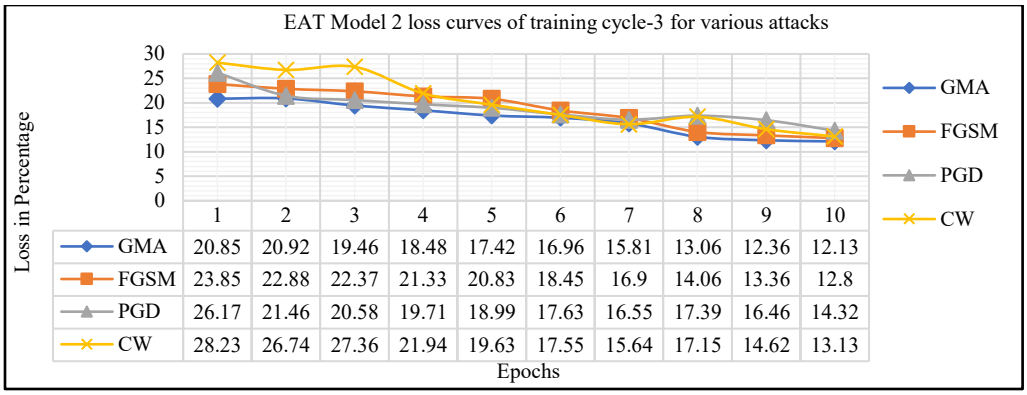
(c) EAT model-3

Fig. 20 The learning curves of the EAT models for the third cycle, i.e., 40% of adversarial examples augmented to trainset using MNIST data



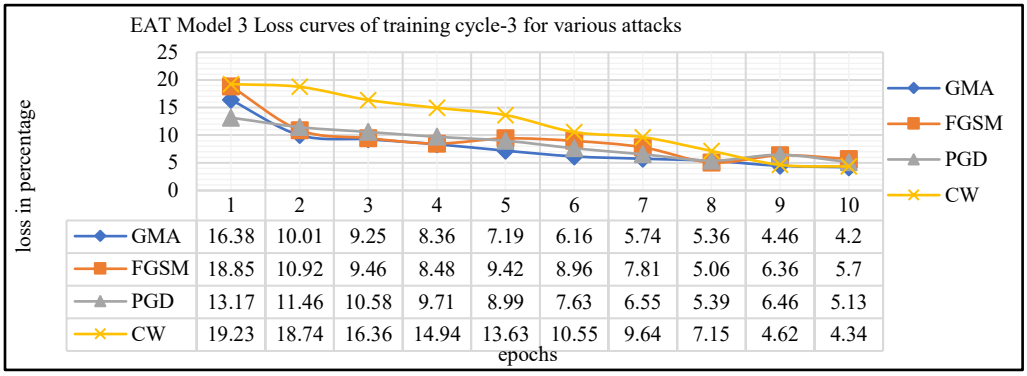
	1	2	3	4	5	6	7	8	9	10
GMA	23.62	22.46	21.66	21.21	21.45	20.43	19.16	18.53	18.76	16.74
FGSM	26.85	24.92	23.46	23.48	21.42	22.96	20.81	19.06	17.36	16.47
PGD	21.17	21.46	22.58	19.71	18.99	17.63	18.55	17.39	16.46	16.75
CW	30.23	29.74	27.36	29.94	23.63	20.55	20.64	17.15	19.62	16.74

(a) EAT model-1



	1	2	3	4	5	6	7	8	9	10
GMA	20.85	20.92	19.46	18.48	17.42	16.96	15.81	13.06	12.36	12.13
FGSM	23.85	22.88	22.37	21.33	20.83	18.45	16.9	14.06	13.36	12.8
PGD	26.17	21.46	20.58	19.71	18.99	17.63	16.55	17.39	16.46	14.32
CW	28.23	26.74	27.36	21.94	19.63	17.55	15.64	17.15	14.62	13.13

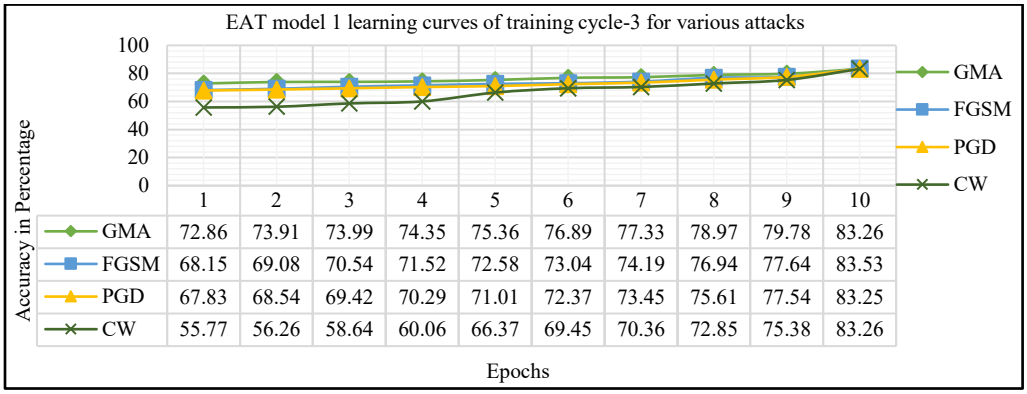
(b) EAT model-2



	1	2	3	4	5	6	7	8	9	10
GMA	16.38	10.01	9.25	8.36	7.19	6.16	5.74	5.36	4.46	4.2
FGSM	18.85	10.92	9.46	8.48	9.42	8.96	7.81	5.06	6.36	5.7
PGD	13.17	11.46	10.58	9.71	8.99	7.63	6.55	5.39	6.46	5.13
CW	19.23	18.74	16.36	14.94	13.63	10.55	9.64	7.15	4.62	4.34

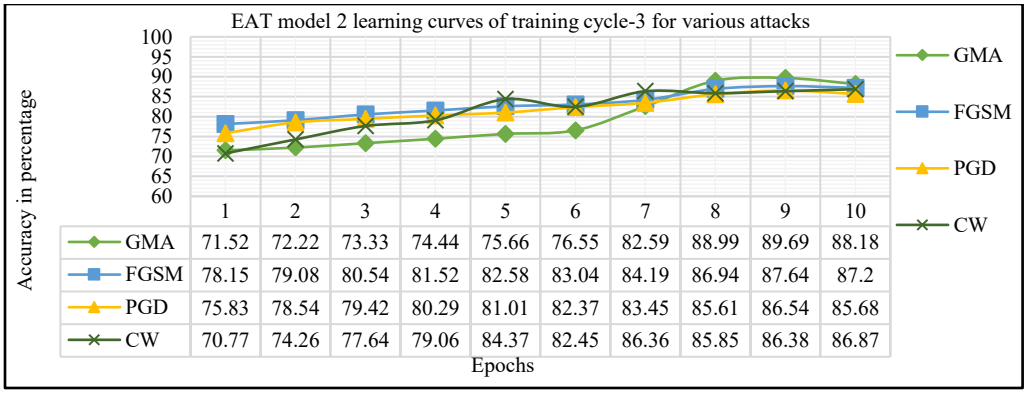
(c) EAT model-3

Fig. 21 The loss curves of EAT models for the third cycle MNIST data



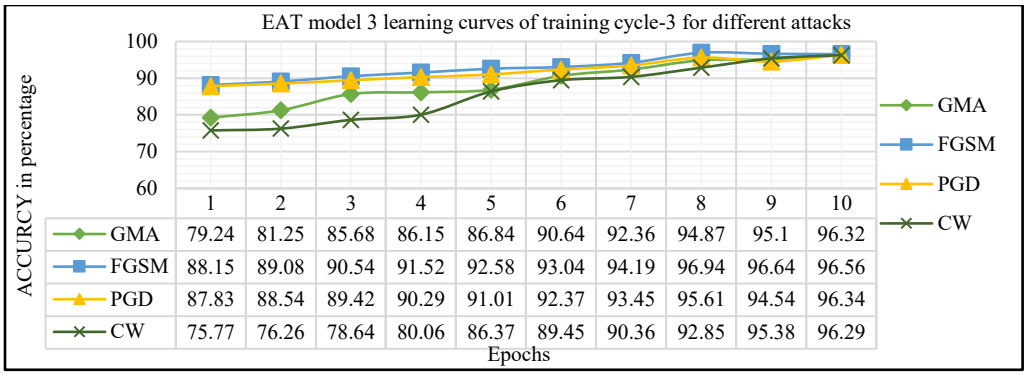
	1	2	3	4	5	6	7	8	9	10
GMA	72.86	73.91	73.99	74.35	75.36	76.89	77.33	78.97	79.78	83.26
FGSM	68.15	69.08	70.54	71.52	72.58	73.04	74.19	76.94	77.64	83.53
PGD	67.83	68.54	69.42	70.29	71.01	72.37	73.45	75.61	77.54	83.25
CW	55.77	56.26	58.64	60.06	66.37	69.45	70.36	72.85	75.38	83.26

(a) EAT model-1



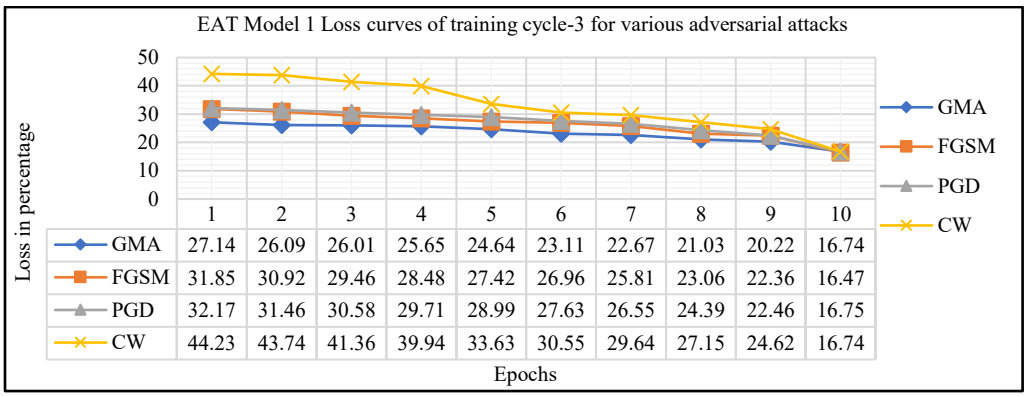
	1	2	3	4	5	6	7	8	9	10
GMA	71.52	72.22	73.33	74.44	75.66	76.55	82.59	88.99	89.69	88.18
FGSM	78.15	79.08	80.54	81.52	82.58	83.04	84.19	86.94	87.64	87.2
PGD	75.83	78.54	79.42	80.29	81.01	82.37	83.45	85.61	86.54	85.68
CW	70.77	74.26	77.64	79.06	84.37	82.45	86.36	85.85	86.38	86.87

(b) EAT model-2

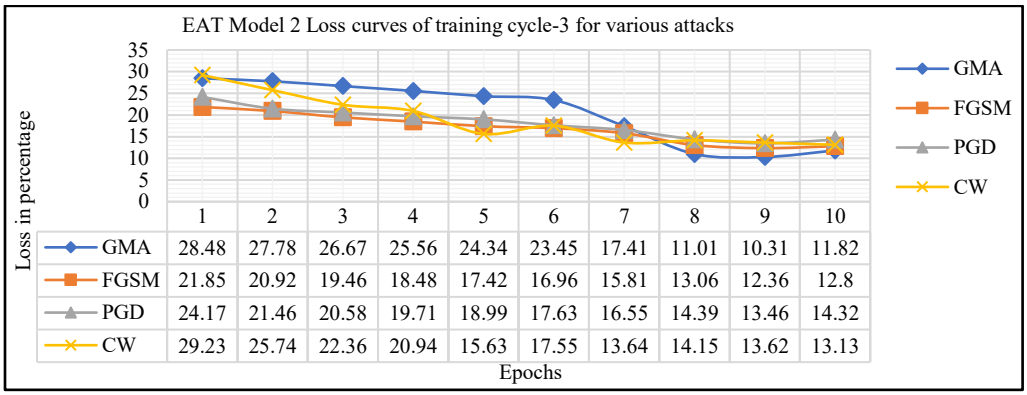


(c) EAT model-3

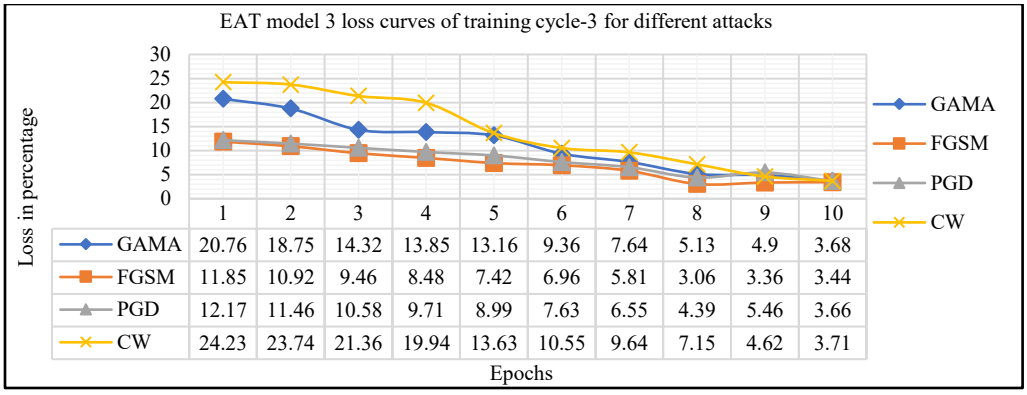
Fig. 22 The learning curves of the EAT models for the third cycle, i.e., 40% of adversarial examples augmented to trainset using Fashion-MNIST data



(a) EAT model-1



(b) EAT model-2



(c) EAT model-3

Fig. 23 The loss curves of EAT models for the third cycle Fashion-MNIST data

7. Conclusion

Securing deep image recognition systems against advanced gradient-based adversarial attacks is paramount. While adversarial training methods effectively counter these attacks, they fail to identify and defend against Gradient-based Miniature Attacks, often succumbing to their influence. In response to this challenge, MIA is a novel adversarial training framework engineered to detect gradient-based attacks such as GMA, FGSM, CW, and PGD. The lightweight MIA architecture ensures robust performance while dramatically reducing the need for extensive training attack samples to 40% compared to existing methods needed 50%. MIA demonstrates an impressive average detection accuracy of 99.71%. The simplicity and computational efficiency of the MIA's architecture make it both cost-effective and scalable, offering a clear advantage over previous AT defense models.

Nevertheless, the effectiveness of the model integration approach to complex models and datasets is minimally guaranteed because this MIA method depends on training methodology rather than a DLM or dataset. This study intentionally excluded more complex detectors, such as VGG16 or ResNet50, to prioritize lightweight design. MIA's unique technique might reduce the overall requirement of adversarial training samples, yet has the disadvantage of depending on integrating two DLMs. Even though the first phase of training used lightweight models, the proposed method requires multiple training cycles. Also, MIA is limited to less complex datasets and DLMs. Future research might focus on alternative or tweaked training strategies to reduce the training overhead and enhance the framework's adaptability, enabling the detection of new attack types across various application areas and further advancing the resilience of deep learning systems against adversarial threats.

References

[1] N.G. Girish Kumar, Ashish Kishore, and Aaditya J. Krishna, "Real-Time Traffic Sign Recognition and Autonomous Vehicle Control System Using Convolutional Neural Networks," *Multimedia Tools and Applications*, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[2] Mohamed Cheniti, Zahid Akhtar, and Praveen Kumar Chandaliya, "Dual-Model Synergy for Fingerprint Spoof Detection Using VGG16 and ResNet50," *Journal of Imaging*, vol. 11, no. 2, pp. 1-13, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[3] Ming Yan et al., "Cancer Type and Survival Prediction Based on Transcriptomic Feature Map," *Computers in Biology and Medicine*, vol. 192, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[4] Haoran Cheng, "Advancements in Image Classification: from Machine Learning to Deep Learning," *ITM Web of Conferences*, *2nd International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2024)*, vol. 70, pp. 1-8, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and Harnessing Adversarial Examples," *arXiv Preprint*, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[6] Nicholas Carlini, and David Wagner, "Adversarial Examples are not Easily Detected: Bypassing Ten Detection Methods," *AISec '17: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Dallas, Texas, USA, pp. 3-14, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[7] Mustafa Sinasi Ayas, Selen Ayas, and Seddik M. Djouadi, "Projected Gradient Descent Adversarial Attack and its Defense on a Fault Diagnosis System," *2022 45th International Conference on Telecommunications and Signal Processing (TSP)*, Prague, Czech Republic, pp. 36-39, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[8] Fahri Anıl Yerlikaya, and Serif Bahtiyar, "Data Poisoning Attacks Against Machine Learning Algorithms," *Expert Systems with Applications*, vol. 208, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Hai Huang et al., "Data Poisoning Attacks to Deep Learning Based Recommender Systems," *arXiv Preprint*, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[10] Ibrahim M. Ahmed, and Manar Younis Kashmoola, "Threats on Machine Learning Technique by Data Poisoning Attack: A Survey," *International Conference on Advances in Cyber Security*, Penang, Malaysia, pp. 586-600, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[11] Antonio Emanuele Cina et al., "AttackBench: Evaluating Gradient-based Attacks for Adversarial Examples," *arXiv Preprint*, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] Chuan Guo et al., "Gradient-Based Adversarial Attacks Against Text Transformers," *arXiv Preprint*, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[13] Zheng Yuan et al., "Meta Gradient Adversarial Attack," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, pp. 7748-7757, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[14] Battista Biggio, Blaine Nelson, and Pavel Laskov, "Poisoning Attacks Against Support Vector Machines," *arXiv Preprint*, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[15] Yingqi Liu et al., "Trojaning Attack on Neural Networks," *25th Annual Network and Distributed System Security Symposium (NDSS 2018)*, San Diego, CA, USA, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[16] Luis Muñoz-González et al., "Towards Poisoning of Deep Learning Algorithms with Back-Gradient Optimization," *AISec '17: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, Dallas, Texas, USA, pp. 27-38, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[17] Nicolas Papernot, "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks," *2016 IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, USA, pp. 582-597, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[18] Nicholas Carlini, and David Wagner, "Defensive Distillation is not Robust to Adversarial Examples," *arXiv Preprint*, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[19] Bakary Badjie, José Cecílio, and António Casimiro, "Denoising Autoencoder-Based Defensive Distillation as an Adversarial Robustness Algorithm Against Data Poisoning Attacks," *ACM SIGAda Ada Letters*, vol. 43, no. 2, pp. 30-35, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[20] Murat Kuzlu et al., "Adversarial Security Mitigations of Mmwave Beamforming Prediction Models Using Defensive Distillation and Adversarial Retraining," *International Journal of Information Security*, vol. 22, no. 2, pp. 319-332, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[21] Antonio Emanuele Cinà et al., "Wild Patterns Reloaded: A Survey of Machine Learning Security Against Training Data Poisoning," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1-39, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[22] Tommaso Zoppi, and Andrea Ceccarelli, "Detect Adversarial Attacks Against Deep Neural Networks with GPU Monitoring," *IEEE Access*, vol. 9, pp. 150579-150591, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[23] Rui Yang, Xiu-Qing Chen, and Tian-Jie Cao, "APE-GAN++: An Improved APE-GAN to Eliminate Adversarial Perturbations," *IAENG International Journal of Computer Science*, vol. 48, no. 3, pp. 827-44, 2021. [Google Scholar] [Publisher Link]

[24] Zeyu Wang et al., "Revisiting Adversarial Training at Scale," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24675-24685, 2024. [Google Scholar] [Publisher Link]

[25] Yuhao Mao et al., "Connecting Certified and Adversarial Training," *Advances in Neural Information Processing Systems*, vol. 36, 2024. [Google Scholar] [Publisher Link]

[26] Shu Hu et al., "Outlier Robust Adversarial Training," *Proceedings of the 15th Asian Conference on Machine Learning*, *PMLR*, pp. 454-469, 2024. [Google Scholar] [Publisher Link]

[27] Maksym Andriushchenko, and Nicolas Flammarion, "Understanding and Improving Fast Adversarial Training," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16048-16059, 2020. [Google Scholar] [Publisher Link]

[28] Yue Xing, Qifan Song, and Guang Cheng, "On the Algorithmic Stability of Adversarial Training," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26523-26535, 2021. [Google Scholar] [Publisher Link]

[29] Sravanti Addepalli, Samyak Jain, and Venkatesh Babu R., "Efficient and Effective Augmentation Strategy for Adversarial Training," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1488-1501, 2022. [Google Scholar] [Publisher Link]

[30] Ruslan Abdulkadirov, Pavel Lyakhov, and Nikolay Nagornov, "Survey of Optimization Algorithms in Modern Neural Networks," *Mathematics*, vol. 11, no. 11, pp. 1-37, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[31] Kevin P. Murphy, *Probabilistic Machine Learning: An Introduction*, MIT Press, 2022. [Google Scholar] [Publisher Link]

[32] Tianyu Pang et al., "Towards Robust Detection of Adversarial Examples," *Advances in Neural Information Processing Systems*, vol. 31, 2018. [Google Scholar] [Publisher Link]

[33] Xin Li, and Fuxin Li, "Attack Samples Detection in Deep Networks with Convolutional Filter Statistics," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, pp. 5764-5772, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[34] Francesco Crecchi et al., "Fader: Fast Adversarial Example Rejection," *Neurocomputing*, vol. 470, pp. 257-268, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[35] Eric Wong, Leslie Rice, and J. Zico Kolter, "Fast is Better than Free: Revisiting Adversarial Training," *arXiv Preprint*, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[36] Ali Shafahi et al., "Adversarial Training for Free!," *Advances in Neural Information Processing Systems*, vol. 32, 2019. [Google Scholar] [Publisher Link]

[37] Xiaojun Jia et al., "LAS-AT: Adversarial Training with Learnable Attack Strategy," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, pp. 13398-13408, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[38] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion, "On the Effectiveness of Adversarial Training Against Common Corruptions," *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence, PMLR*, pp. 1012-1021, 2022. [Google Scholar] [Publisher Link]

[39] Zhuang Qian et al., "A Survey of Robust Adversarial Training in Pattern Recognition: Fundamental, Theory, and Methodologies," *Pattern Recognition*, vol. 131, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[40] Afia Sajeeda, and B.M. Mainul Hossain, "Exploring Generative Adversarial Networks and Adversarial Training," *International Journal of Cognitive Computing in Engineering*, vol. 3, pp. 78-89, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[41] Jianyu Wang, and Haichao Zhang, "Bilateral Adversarial Training: Towards Fast Training of More Robust Models Against Adversarial Attacks," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), pp. 6629-6638, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[42] Kathrin Grosse et al., "On the (Statistical) Detection of Attack Samples," *arXiv Preprint*, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[43] Andrea Paudice et al., "Detection of Adversarial Training Examples in Poisoning Attacks through Anomaly Detection," *arXiv Preprint*, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[44] Florian Tramèr et al., "Ensemble Adversarial Training: Attacks and Defenses," *arXiv Preprint*, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[45] Lars Buitinck et al., "API Design for Machine Learning Software: Experiences from the Scikit-Learn Project*," European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases: Languages for Data Mining and Machine Learning*, 2013. [Google Scholar] [Publisher Link]

[46] Yann LeCun, Corinna Cortes and Chris Burges, MNIST Handwritten Digit Database, Github. [Online]. Available: https://github.com/unlucky-13/Level-4-Term-2/blob/master/CSE472%20Machine%20Learning%20Sessional/Assignment%203/MNIST%20handwritten%20digit%20database%2C%20Yann%20LeCun%2C%20Corinna%20Cortes%20and%20Chris%20Burges.html

[47] Jonas Rauber et al., "Foolbox Native: Fast Adversarial Attacks to Benchmark the Robustness of Machine Learning Models in Pytorch, Tensorflow, and JAX," *Journal of Open Source Software*, vol. 5, no. 53, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[48] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms," *Arxiv*, 2017. [CrossRef] [Google Scholar] [Publisher Link]