Received: 18 April 2025 | Revised: 4 September 2025 | Accepted: 12 November 2025 | Published online: 31 December 2025

RESEARCH ARTICLE

Artificial Intelligence and Applications
2025, Vol. 00(00) 1-10
DOI: 10.47852/bonviewAIA52025957

BON VIEW PUBLISHING

Efficient Defense Against First Order Adversarial Attacks on Convolutional Neural Networks

Subah Karnine[1], **Sadia Afrose**[1], **and Hafiz Imtiaz**[1,]*

[1] *Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Bangladesh*

Abstract: Machine learning models, especially neural networks, are vulnerable to adversarial attacks, where inputs are purposefully altered to induce incorrect predictions. These adversarial inputs closely resemble benign (unaltered) inputs, making them difficult to detect, and pose significant security risks in critical applications, such as autonomous vehicles, medical diagnostics, and financial transactions. Several methods exist to improve the model's performance against these adversarial attacks, which typically modify the network architecture or training procedure. Often times, these adversarial training techniques only provide robustness against specific attack types and/or require substantial computational resources, making them impractical for real-world applications with limited resources. In this work, we propose a computationally-efficient adversarial fine-tuning approach to enhance the robustness of Convolutional Neural Networks (CNNs) against adversarial attacks and attain the same level of performance as the conventional adversarial training. More specifically, we propose to identify specific parts of the neural network model that are more vulnerable to adversarial attacks. Our analysis reveals that only a small portion of these vulnerable components accounts for a majority of the model's errors caused by adversarial attacks. As such, we propose to selectively fine-tune these vulnerable components using different adversarial training methods to develop an effective and resource-efficient approach to improve model robustness. We empirically validate our proposed approach with varying dataset and algorithm parameters. We demonstrate that our approach can achieve similar performance as the more resource-intensive conventional adversarial training method.

Keywords: adversarial attacks, machine learning model security, convolutional neural networks, fast gradient sign method (FGSM), projected gradient descent (PGD)

1. Introduction

Deep neural networks have proven to be highly effective in solving complex machine learning tasks, such as image recognition [1–3], speech recognition [4], natural language processing [4], and even computer games [5, 6]. These networks have achieved remarkable success in recognizing images with accuracy levels close to (and sometimes exceeding) that of humans. However, researchers have recently discovered that these networks are prone to adversarial attacks. More specifically, these attacks intention ally perturb samples to simulate worst-case scenarios, leading the network to output incorrect results with high confidence levels.

Adversarial examples were first discovered in the image classification domain Mądry et al. [7]. Their research showed that it is possible to transform the classification output corresponding to an image by making minimal alterations to it. This means that given an input $\mathbf{x}$ and any target classification t, it is possible to discover a new input $\tilde{\mathbf{x}}$ that is very similar to the original input $\mathbf{x}$, but classified as the target $t' \neq t$. The quantity of change required is often so small that it is difficult for humans to detect, making it a significant challenge to use neural networks in security-sensitive areas. In other words, adversarial examples pose a significant concern, since they can limit the domains in which neural networks can be safely used. For example, using neural networks in self-driving vehicles can be risky because an attacker could exploit adversarial examples to cause the car to take actions that it is not supposed to take [8]. As a result, constructing robust neural networks, that are resistant to such attacks, is a top priority for researchers in the field.

Consequently, robust defense mechanisms against such attacks on modern machine learning (ML) models have been the topic of extensive research. Schölkopf et al. [9] has started the journey towards adversarial resistant models. Since then, numerous techniques have been proposed to enhance the robustness of neural networks against adversarial threats. These include approaches similar to adversarial training – where the model is trained on adversarial examples, defensive distillation methods — which aim to smooth the model's decision boundaries [10], and certified defenses – which provide theoretical guarantees against specific attack types. However, existing solutions often have practical limitations, such as being tailored to specific attacks, requiring substantial computational resources, and offering incomplete protection against the wide range of possible adversarial attacks.

Our Contributions. In this work, we propose a novel and computationally efficient method for ensuring adversarial robustness of CNNs. We achieve this by proposing a selective adversarial fine-tuning approach. More specifically, our proposed approach identifies the model components that are most vulnerable to adversarial perturbations, and then ensures the model's robustness against adversarial attacks by selectively finetuning those components. To

***Corresponding author:** Hafiz Imtiaz, Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Bangladesh. Email: hafizimtiaz@eee.buet.ac.bd

this end, we identify the convolutional filters of a CNN model that are highly susceptible to adversarial attacks. We show that our proposed adversarial fine-tuning of these filters enables the resulting model to maintain a high accuracy on benign inputs, while exhibiting similar, if not better, resilience against adversarial inputs. Our contributions are summarized below:

1) We show that the effect of first order adversarial attacks on a CNN model is neither uniform nor random. In fact, certain parts of the model are more susceptible to an attack, regardless of data class. We empirically show this by identifying the filters in the convolutional layers across different datasets.
2) Since certain specific components of the model are more vulnerable to adversarial attacks, we argue that focusing on those components are crucial for ensuring model robustness against those attacks. To this end, we propose to split the model into trainable and non-trainable sections. We empirically demonstrate that performing adversarial fine-tuning of the vulnerable components in this way provides a model that performs just as well as existing adversarial training methods. Additionally, this enables a much simpler and computationally light adversarial training.
3) We demonstrate that increasing the fraction of trainable parts of the model does not significantly improve the model's robustness. This re-enforces our claim that the whole model does not need to be trained for appropriate adversarial security

Notations. For vector, matrix, and scalar, we used bold lower-case letter ($\mathbf{v}$), bold capital letter ($\mathbf{V}$), and unbolded letter (M), respectively. We used the symbol $\mathbf{v}_\mathbf{n}$ for the n-th column of the matrix $\mathbf{V}$; and v_{ij} denotes the (i, j)-th entry of matrix $\mathbf{V}$. We sometimes denote the set $\{1, 2, \ldots, N\}$ as $[N]$. Inequality $\mathrm{V} \geq 0$ apply entry-wise. We denoted L_2 norm (Euclidean norm) with $\|\cdot\|_2$, the L_∞ norm with the $\|\cdot\|_\infty$, and the Frobenius norm with $\|\cdot\|_F$.

2. Background and Related Works

2.1. Adversarial attacks on neural networks

Neural networks that are commonly used in practice, such as computer vision and speech recognition applications, are susceptible to adversarial attacks that manipulate the model into predicting the wrong output. Protection against such attacks has garnered particular research interest [11–14]. In computer vision, adversarial attacks are of particular interest as very small and undetectable perturbations can be added to an image to fool a model with high probability [15–17].

Ideally, for datasets with well-separated classes, it is expected for the classifier model to assign the same class to both the original input $\mathbf{x}$ and the adversarial input $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$, as long as $\|\boldsymbol{\eta}\|_\infty \leq \epsilon$, where ϵ is small enough to be discarded by the sensor or data storage device, which considers it to be noise or measurement error and not impact the classification decision [15, 18]. If w represents the weights of the linear model, then the dot product between $\mathbf{w}$ and adversarial example $\tilde{\mathbf{x}}$ is $\mathbf{w}^\top\tilde{\mathbf{x}} = \mathbf{w}^\top\mathbf{x} + \mathbf{w}^\top\boldsymbol{\eta}$. After going through the model, the previously imperceptible noise marker $\boldsymbol{\eta}$ causes the activation to grow by $\mathbf{w}^\top\boldsymbol{\eta}$. This increased activation can be maximized by assigning $\boldsymbol{\eta} = \mathrm{sign}(\mathbf{w})$, while making sure the constraint on $\boldsymbol{\eta}$ still holds. For a weight vector containing elements with an average magnitude of m and having n dimension, this perturbation results in an activation increase of ϵmn. While the norm of the perturbation $\boldsymbol{\eta}$ does not grow with the dimensionality of the problem, the change in activation caused by the perturbation ϵ can grow linearly with the dimensionality. As a result, in high-dimensional problems, it is possible to make many infinitesimal changes to the input that add up to one large change in the output [7, 15, 18].

2.2. Theoretical framework of adversarial vulnerability

Adversarial vulnerability is not an incidental flaw but a direct consequence of a model's reliance on "non-robust features" — patterns in the data distribution that are highly predictive for standard classification but are inherently brittle and unintelligible to human perception. Gradient-based attacks, such as FGSM and PGD (described below), are designed to exploit these very features by taking a step in the direction of the greatest change in the model's loss function [15, 18]. The high gradient sensitivity of the model to these features means that a tiny, imperceptible perturbation to the input can cause a significant change in the output, which is what we empirically observe in the "dominant filters" in this work.

We note that this sensitivity is systematically amplified through the CNN's hierarchical structure, transforming a minute perturbation into a catastrophic error [19]. Filters in early convolutional layers, which are responsible for extracting foundational, low-level features like edges and textures, are directly susceptible to the high-frequency adversarial noise, which can corrupt these features at the base of the network's representational hierarchy [20]. This initial error is then compounded as it propagates through subsequent layers, amplified by non-linear activation functions. The cumulative effect ensures that a minor perturbation at the input can lead to a significant distortion by the time it reaches the deeper layers, ultimately leading to misclassification.

2.3. Common adversarial attack methods

Fast Gradient Sign Method (FGSM). For a neural network model, let $\boldsymbol{\theta}$ be the model parameters, y be the target associated with input sample $\mathbf{x}$, and $J(\boldsymbol{\theta}, \mathbf{x}, y)$ be the cost function. Ilyas et al. [15] showed that the cost function can be linearized around the current $\boldsymbol{\theta}$ value, obtaining an optimal max-norm constrained perturbation of $\boldsymbol{\eta} = \epsilon\mathrm{sign}(\Delta * J(\boldsymbol{\theta}, \mathbf{x}, y))$. Goodfellow referred this approach as the Fast Gradient Sign Method (FGSM) of generating adversarial perturbations, and thereby, adversarial samples. Primarily designed to be fast, instead of a close adversarial estimate, FGSM is optimized for L_∞ distance metrics to ensure the amount of perturbation remains within the fixed bound of ϵ. Here, the L_∞ norm measures the largest absolute difference between every element of an input and its perturbed counterpart. The adversarial example can be calculated as $\tilde{\mathbf{x}} = \mathbf{x} + \epsilon\mathrm{sign}(\Delta * J(\boldsymbol{\theta}, \mathbf{x}, y))$, where ϵ is often chosen to be small enough to be imperceptible to humans. FGSM is a simple one-step algorithm for maximizing the inner part of the saddle point formulation of the loss function [7, 21].

Projected Gradient Descent (PGD). Mądry et al. [7] proposed a multi-step variant of FGSM. The Projected Gradient Descent (PGD) scheme for generating adversarial examples is a more powerful iterative attack that performs multiple gradient descent steps to find the perturbation with maximum loss while ensuring that the adversarial input stays within the constraint typically imposed by the $L\infty$ norm. It is shown to produce more effective attacks compared to FGSM. Instead of taking a single step of size ϵ in the direction of the gradient-sign, multiple smaller steps are taken. More specifically, at t-th iteration, the adversarial sample is given by $x^{t+1} = \Pi_x + S(x^t + \alpha\mathrm{sign}(\Delta_x J(\boldsymbol{\theta}, \mathbf{x}, y)))$. Here, $\boldsymbol{S}$ is the set of allowed perturbations chosen such that it maintains perceptual similarities between an input and its perturbed counterpart, and α is the step size.

2.4. Adversarial defense mechanisms

Several defense mechanisms have been proposed to address the vulnerabilities, such as adversarial training [7, 21], feature squeezing [22], defensive distillation [10], and other detection methods [23]. Issaoui et al. [24] presented an approach of combining custom activation functions with adversarial training. However, the existing methods often involve significant computational burden, measured by the number of trainable parameters, and practical limitations. For example, adversarial training improves robustness by retraining the entire model with adversarial examples, requiring a large number of trainable parameters. Defensive distillation aims to smooth decision boundaries by training a secondary model on softened class probabilities, but it has been shown to be less effective against stronger attacks [22]. Feature squeezing reduces adversarial effects by preprocessing inputs but often degrades benign accuracy and may not perform well against adaptive attacks.

In recent years, some works have highlighted the need for efficient and robust defenses against adversarial attacks on a diverse range of neural networks. Research on audio perturbations demonstrates that speech recognition systems are highly vulnerable to selective and multi-targeted manipulations, which shows that attackers can force systems to misrecognize specific phrases [25, 26]. Similarly, works on text-based backdoor attacks reveal that adversaries can exploit trigger positions and word choices to induce targeted misrecognition, further stressing the fragility of NLP models [25]. In the graph domain, works on dual-targeted and discrepancy-based attacks expose the susceptibility of graph neural networks (GNNs) to adversarial perturbations that selectively manipulate predictions [25, 26]. To mitigate such threats, detection-focused efforts have been proposed, such as score-based anomaly detection for audio and text modification techniques for NLP [25, 26]. These works show promise in identifying adversarial inputs, and demonstrate that adversarial attacks are becoming increasingly sophisticated, multi-targeted, and domain-specific, encompassing speech, text, and graph data. The works of Luo et al. [27] and Peng et al. [28] share the idea of targeted adaptation — dynamically identifying the most influential parameters (or layers/tokens) and updating only those. These approaches yield strong task performance with far fewer trainable weights and lower runtime/energy use. However, pre- and mid-network filtering often degrade benign-sample accuracy and can be defeated by adaptive or learned attacks that reintroduce adversarial signal within the filter's passband or exploit gradient obfuscation. Empirical studies show such filters may help against specific perturbations but provide limited, non-universal robustness across diverse, stronger threat models [29].

Advanced adversarial defense mechanisms have been proposed recently to enhance model robustness, while improving efficiency. Compressed Optimized Neural Networks integrate weight compression and multi-expert training to streamline deep neural networks — this improves both storage efficiency and adversarial robustness by introducing complexity that makes it more difficult for adversaries to exploit vulnerabilities while maintaining high accuracy and efficiency [30]. Similarly, Adversarial Feature-Level Fusion (or AFLF) strengthens model resilience by leveraging attention-based feature selection and adversarial feature learning, where robust feature extraction combined with a model-agnostic adversarial learning process improves classification robustness against a wide range of adversarial attacks [31]. Another approach, Robustness via-Synthesis, employs generative adversarial perturbations to enhance adversarial training, offering a significant advantage over traditional gradient-based adversarial training by synthesizing diverse perturbations using a generator network, thereby enabling more robust defenses against different types of attacks [32].

Our proposed selective adversarial fine-tuning compliments these recent advancements by focusing on optimizing the most vulnerable components of a neural network rather than retraining the entire model. By identifying and selectively fine-tuning specific convolutional filters that are highly susceptible to adversarial attacks, our method significantly reduces computational overhead, while maintaining robustness comparable to conventional adversarial training. This targeted approach provides a practical balance between efficiency and adversarial defense, making it a viable alternative for real-world deployment. We would like to emphasize that while prior works on parameter-efficient robustness (e.g., Luo et al. [27] and Peng et al. [28]) have shown benefits by tuning selected layers or subsets of parameters, these methods generally operate at a coarse level, updating entire layers or large parameter groups. In contrast, our approach identifies and fine-tunes only the most adversarially vulnerable filters, a finer-grained strategy that avoids retraining redundant parameters and prevents loss of benign accuracy. Additionally, unlike pruning-based sparsification, which discards vulnerable filters entirely, our method retains these filters and strengthens them through targeted adversarial fine-tuning.

3. Proposed Approach Against Adversarial Attacks

Adversarial training is crucial for any neural network model deployed in applications, where security is a priority. Nevertheless, computation cost, model complexity and memory usage, and inference times are crucial factors that need to be considered during adversarial training. Moreover, the adversarial training of the model must be robust enough to minimize the effect of different adversarial attacks. Our work is motivated by such need for efficient and robust adversarial training scheme. More specifically, we focus on a CNN trained on the MNIST handwritten digits dataset [3]. To that end, we are interested in investigating:

1) Does an adversarial attack affect different parts of a network equally?
2) If not, how can we take advantage of this while performing adversarial training of the network?

We utilized the Cleverhans open-source library to generate the adversarial examples, and demonstrate the vulnerability of neural network models. The library offers a selection of attacks and countermeasures for testing how susceptible machine learning models are to adversarial examples.

3.1. Model

As mentioned before, we consider a CNN for image classification and trained the model on the MNIST dataset. After training, the CNN captures the spatial relationships present in an image fairly accurately. Because fewer parameters are needed and weights can be reused, this architecture yields superior results than fully-connected neural networks. We present the details of the model we used in Table 1. Except for the output layer, we have used the ReLU activation. The structure of the base model is shown in Figure 1. This model has a total of 120,042 trainable parameters. Note that, the MNIST dataset's training partition consists of 60,000 gray scale images of size 28 × 28 pixels, and the test partition consists of 10,000 gray scale images of the same size.

The model is trained with RMSprop algorithm and 0.001 learning rate. Early stopping is employed to prevent over-fitting by monitoring the accuracy of the set over a patient of 2 epochs. A dropout layer after every convolutional layer was used further prevent overfitting. The accuracy and loss plots of the training and validation set, as shown in

Table 1
Details of the CNN model

Layer	Output shape	Param #
Conv2D	28 × 28 × 32	832
MaxPooling2D	14 × 14 × 32	0
Dropout	14 × 14 × 32	0
Conv2D	14 × 14 × 64	18,496
MaxPooling2D	7 × 7 × 64	0
Dropout	7 × 7 × 64	0
Flatten	3136	0
Dense	32	100,384
Dense	10	330

Figure 1
CNN model under consideration

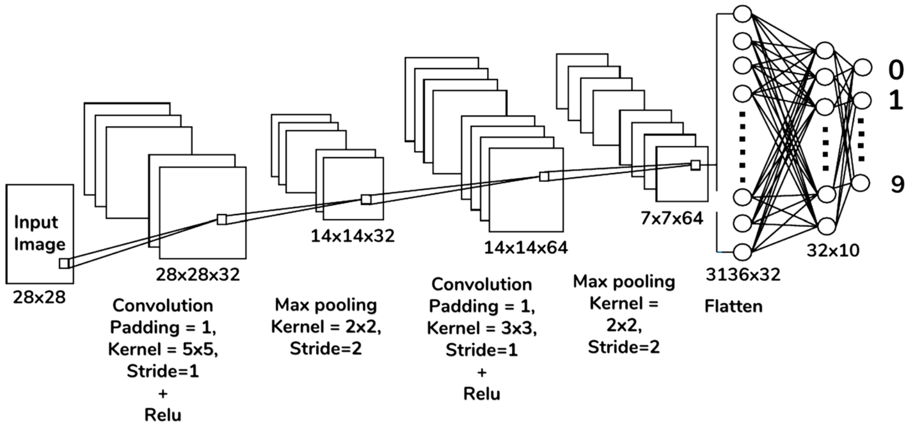

Input Image
28x28
28x28x32
Convolution Padding = 1, Kernel = 5x5, Stride=1 + Relu
14x14x32
Max pooling Kernel = 2x2, Stride=2
14x14x64
Convolution Padding = 1, Kernel = 3x3, Stride=1 + Relu
7x7x64
Max pooling Kernel = 2x2, Stride=2
3136x32
Flatten
32x10
0
1
9

Figure 2
Accuracy and loss plots of the base model.

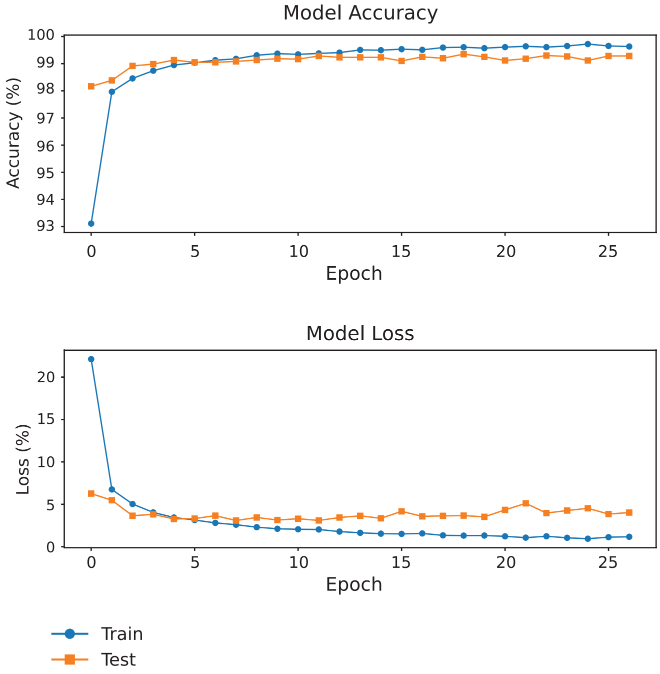


Figure 2, demonstrate that the model has managed to properly learn without overfitting to the dataset. Evaluating it against the test set gives us an accuracy of 99.33%.

3.2. Generating adversarial examples

To test the vulnerability of the model, adversarial examples are generated using both the FGSM and PGD approaches. Given a valid input data $\mathbf{x}$ and a target classification, $t = C(\mathbf{x})$, it is possible to find a similar input $\tilde{\mathbf{x}}$ such that $C(\tilde{\mathbf{x}}) = t$. Here $C(\mathbf{x}) = \arg\max F(\mathbf{x})$ is the classifier function, and $F(\mathbf{x})$ is the neural network loss. Additionally, $\mathbf{x}$ and $\tilde{\mathbf{x}}$ are close with respect to some distance metric. The adversarial example $\tilde{\mathbf{x}}$ with this property is known as a targeted adversarial example [7, 21]. A less powerful attack, or un-targeted attack, classifying $\mathbf{x}$ as a given target class searches only for a perturbed input $\tilde{\mathbf{x}}$ so that $C(\tilde{\mathbf{x}}) \neq C(\mathbf{x})$, and that $\mathbf{x}$ and $\tilde{\mathbf{x}}$ are close spatially. Carlini and Wagner [33] considered three different approaches to choosing the target class in a targeted attack:

1) Average Case — target class selected uniformly at random among the incorrect labels
2) Best Case — targeting the class least difficult to attack
3) Worst Case — targeting the class most difficult to attack

3.3. Model behavior under FGSM and PGD attacks

As mentioned before, a dataset of adversarial examples using FGSM and PGD approaches is generated for the corresponding

Table 2
Targeted FGSM attack

Source Class	Target Class 0	Target Class 1	Target Class 2	Target Class 3	Target Class 4	Target Class 5	Target Class 6	Target Class 7	Target Class 8	Target Class 9
0	X	X	X	X	X	✓	X	X	X	X
1	X	X	X	X	✓	X	X	X	X	X
2	X	X	X	X	X	X	X	X	X	X
3	X	X	X	X	X	✓	X	X	X	X
4	X	X	X	X	X	X	X	✓	X	✓
5	X	X	X	X	X	X	X	X	X	X
6	X	X	X	X	X	X	X	X	X	X
7	X	X	X	X	✓	X	X	X	X	X
8	X	X	X	X	X	X	X	X	X	X
9	X	X	X	X	✓	X	X	X	X	X

Table 3
Targeted PGD attack

Source Class	Target Class 0	Target Class 1	Target Class 2	Target Class 3	Target Class 4	Target Class 5	Target Class 6	Target Class 7	Target Class 8	Target Class 9
0	X	✓	✓	✓	✓	✓	✓	✓	✓	✓
1	✓	X	✓	✓	✓	✓	✓	✓	✓	✓
2	✓	✓	X	✓	✓	✓	✓	✓	✓	✓
3	✓	✓	✓	X	✓	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	X	✓	✓	✓	✓	✓
5	✓	✓	✓	✓	✓	X	✓	✓	✓	✓
6	✓	✓	✓	✓	✓	✓	X	✓	✓	✓
7	✓	✓	✓	✓	✓	✓	✓	X	✓	✓
8	✓	✓	✓	✓	✓	✓	✓	✓	X	✓
9	✓	✓	✓	✓	✓	✓	✓	✓	✓	X

Table 4
Adversarial accuracy before adversarial training vs magnitude of perturbation of the FGSM attack

Model type	Value of $\epsilon = 0.1$	Value of $\epsilon = 0.2$	Value of $\epsilon = 0.3$	Value of $\epsilon = 0.5$
Shallow model	62.10%	21.10%	10.61%	6.71%
Base model	70.36%	34.01%	21.26%	16.75%
Deeper model	74.15%	25.88%	11.64%	6.98%

Table 5
Adversarial accuracy before adversarial training vs magnitude of perturbation of the PGD attack

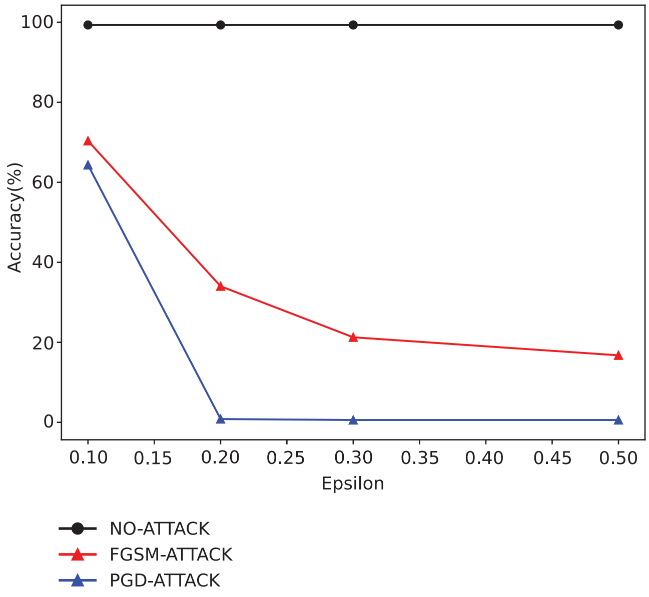
Model type	Value of $\epsilon = 0.1$	Value of $\epsilon = 0.2$	Value of $\epsilon = 0.3$	Value of $\epsilon = 0.5$
Shallow model	60.00%	1.44%	0.89%	0.89%
Base model	64.34%	0.82%	0.58%	0.58%
Deeper model	67.95%	5.51%	1.20%	1.14%

MNIST dataset with different ϵ values. We employ a white box targeted FGSM and PGD attack on the base model. Intuitively, higher ϵ values result in higher attack success rates, i.e., the model performance should have higher error rates. The model was evaluated using a test set of 10,000 adversarial examples for different values of ϵ.

For the FGSM and PGD attacks, each image from a class is used to produce adversarial examples targeting the remaining classes. In Tables 2 and 3, we show whether an adversarial attack is successful with its intention when the perturbation is induced to output a predetermined target class. As evident from Table 2, targeted FGSM attacks do not provide as much success as targeted PGD attacks. More specifically, adversarial accuracy of the model for the same amount of perturbation drops down to 10.32% for targeted FGSM attacks and 21.26% for non-targeted FGSM attacks on the test set. On the other hand, since PGD is a stronger attack, the targeted PGD attacks are more successful. We show the details of the targeted PGD attacks on our model in Table 3. For the same level of perturbation, the adversarial accuracy decreases to 9.78% for targeted PGD attacks and 0.58% for non-targeted PGD attacks on the test set.

To further test the effect of the magnitude of this perturbation of the model with a shallow and a deeper network, we built two more models trained on MNIST. The shallow model has two convolution layers with number of filters 16 and 32, respectively. The deeper network has one added convolutional layer, making it three convolutional layers with 32, 64, and 128 filters. The other parameters of the networks are kept consistent with that of our base model. The results are summarized in Tables 4 and 5. As expected, all the models perform worse with higher perturbation attack. In Figure 3, we show the model performance on adversarial examples for different ϵ values. It is evident from this figure that PGD is the stronger attack approach of the two.

Figure 3
Adversarial accuracy vs magnitude of perturbation



3.4 Filter identification

We hypothesize that the adversarial examples are generated by exploiting particular filters in the convolution layers. In this section, we investigate this hypothesis and identify the filters more susceptible to be exploited during the FGSM or PGD attacks. We extract the output features of the convolutional layers of the model and observe their individual effects on different inputs. As such, the difference in output of the convolutional layer between a benign input image and the corresponding adversarial input image provides the relative effect of an adversarial attack on the layer.

To systematically identify these vulnerable filters, we follow a structured approach. The process consists of three main steps: (i) generating adversarial images, (ii) extracting convolutional features, and (iii) identifying the most affected filters. This method is applied to all convolutional layers of the model to ensure that vulnerabilities are captured across different levels of feature abstraction. The entire process is outlined in Algorithm 1, where adversarial examples are first generated for each input, followed by feature extraction across convolutional layers. The difference in activation values between benign and adversarial images is computed to quantify the effect of adversarial perturbations. Finally, the most frequently affected filters in each layer are identified as dominant filters, which are highly susceptible to adversarial attacks.

From the test set of the MNIST dataset, we selected 100 images from each class, and generated adversarial images corresponding to these benign images with a white box attack targeting the remaining nine classes with $\epsilon = 0.3$. As a result, 900 targeted adversarial examples from the 100 images of each class (9 adversarial examples for a single image) are generated. We extracted the outputs of the convolutional layers, and calculated the difference in the output between a benign

Algorithm 1: Filter identification process

Step 1: Generate Adversarial Examples

```
for each image x in the selected class (limit to num_images) do
    for each target class t in range(num_classes) do
        if t is not the original class of x then
            $x_{adv}$ ← GenerateAdversarialExample(x, t)
            Store benign-adversarial image pair (x,$x_{adv}$)
        end if
    end for
end for
```

Step 2: Extract Convolutional Features

```
for each convolutional layer l in CNN do
    for each (x,$x_{adv}$) in benign-adversarial pairs do
        $F_{benign,l}$ ← ForwardPassCNN(x, layer=l)
        $F_{adv,l}$ ← ForwardPassCNN($x_{adv}$, layer=l)
        $D_l$ ← ComputeRMSDifference($F_{benign,l}$,$F_{adv,l}$)
    end for
end for
```

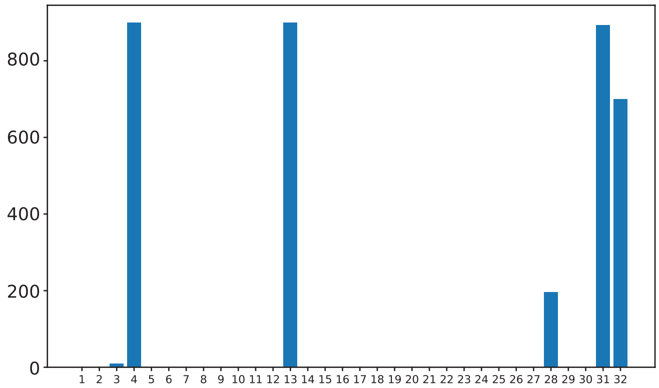
Step 3: Identify Most Affected Filters

```
for each convolutional layer l in CNN do
    $S_l$ ← SelectTopKFilters($D_l$, top_filters_per_layer)
end for
```

Step 4: Identify top 10% most frequently affected filters in layer l as dominant filters image and its adversarial counterpart. As hypothesized, we observe that some filters are affected more than others. We selected the top 10% of the affected filters for each of the 900 benign-adversarial pairs. Frequency of the filters appearing in the top 10% of all the convolutional filters for the first convolutional layer for input class 0 on the MNIST dataset is plotted on a histogram shown in Figure 4. The x-axis labels are filter IDs, and y-axis labels are frequency of occurrence. As is clear from the results, some filters cause more difference in convolutional layer output. We repeat this investigation for all the other nine classes, and observed that the top 10% filters remain mostly the same. That is, the same filters in a convolutional layer are affected the most in an adversarial attack regardless of the input class, as can be seen from Figure 5. The second convolutional layer also exhibited a similar behavior. The identified filters for the second layer are shown in Figure 6. It is important to note that our approach differs fundamentally from pruning or conventional adversarial training. While pruning would discard these dominant filters, risking the loss of critical feature representations, our method instead fine-tunes them adversarially to strengthen their resilience. This selective correction leverages the empirical finding that a small subset of filters is consistently exploited across attack scenarios, thus enabling a more efficient yet equally robust alternative to full adversarial retraining.

Figure 4

The frequency of the affected filters appearing in the top 10% of all the convolutional filters for the first layer



Characteristics of Dominant Filters. It is evident from the aforementioned figures that the identified filters exhibit higher difference in activation for specific classes, and play a critical role in feature extraction. In other words, their sensitivity to small changes in the input makes them particularly susceptible to adversarial perturbations. Even when the degree of perturbation ϵ is changed, the identified vulnerable filters remain almost the same. In other words, first-order adversarial attacks target these filters, and exploit their high activation values (due to subliminal changes in the input image) to amplify the errors that propagate through the network. This evidently impacts the classification performance of the model under attack significantly.

Source of Vulnerability. The heightened vulnerability of a few particular convolutional filters is closely tied to the hierarchical structure of the network. In general, convolutional filters in the initial layers, which are responsible for extracting foundational features (such as edges and textures), are affected as adversarial perturbations disrupt low-level feature extraction, and propagate errors to subsequent layers. On the other hand, convolutional filters in the intermediate layers aggregate and refine outputs from earlier layers. Therefore, adversarial perturbations distort mid-level feature representations in these filters, which further propagate the errors deeper into the network. Filters in the deeper layers rely heavily on the preceding layers to generate accurate outputs and to form high-level abstractions. Errors introduced in earlier layers are therefore amplified in these layers, making their filters particularly vulnerable to adversarial perturbations. We have empirically validated that these observations do not change with a change in the level of perturbation ϵ, and as a result, does not affect the identified dominant filters for a given model. Therefore, employing defensive measures on these particular filters should provide strong defense against adversarial attacks, while not sacrificing the model performance on benign inputs. Additionally, our proposed approach

Figure 5

The frequency of the affected filters appearing in the top 10% for the first convolutional layer for all input classes

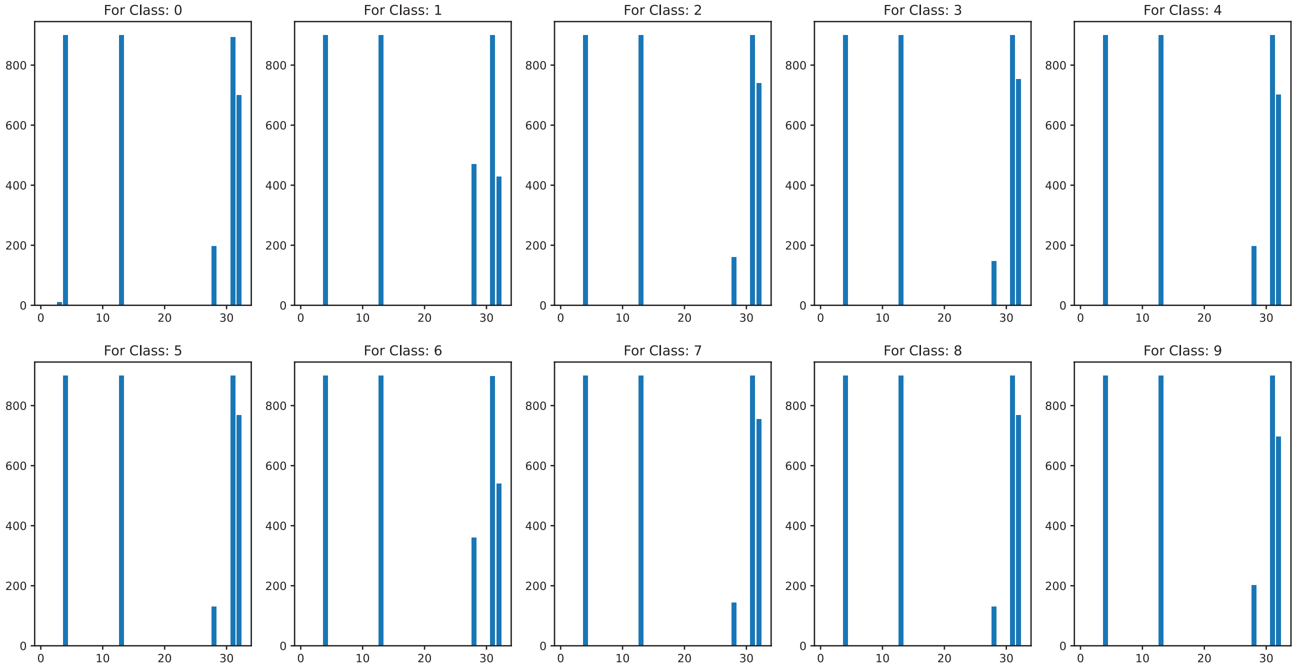
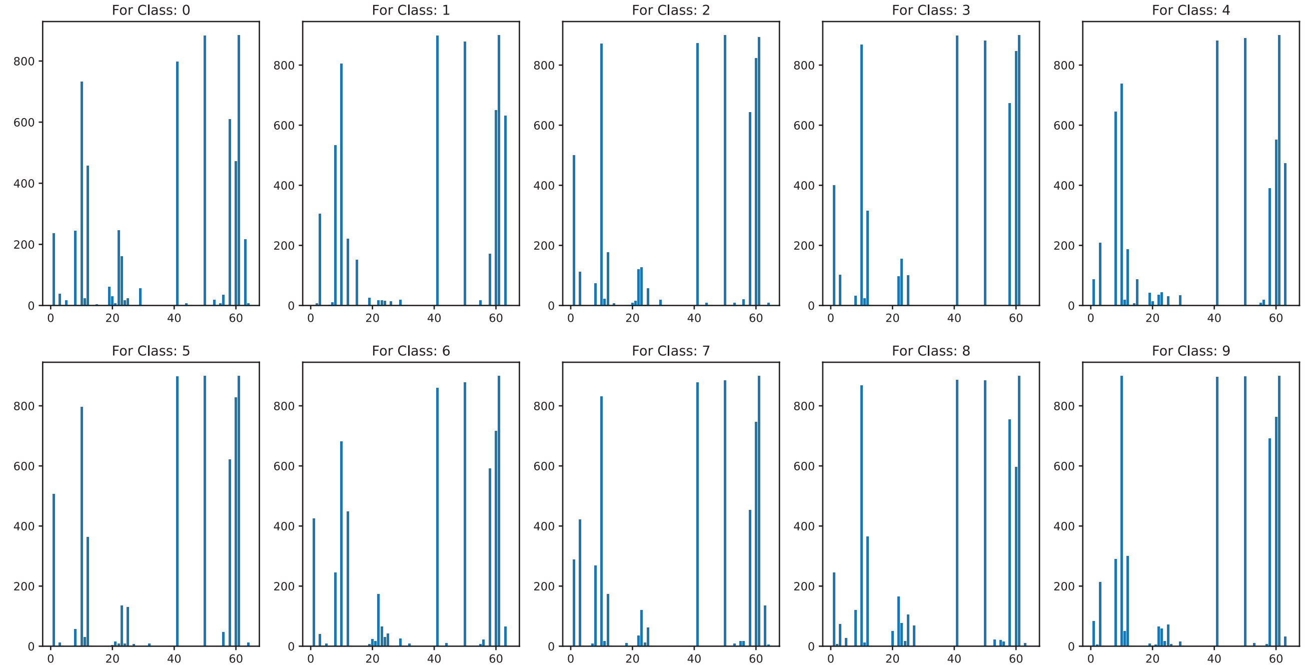


Figure 6
The frequency of the affected filters appearing in the top 10% for the second convolutional layer for all input classes

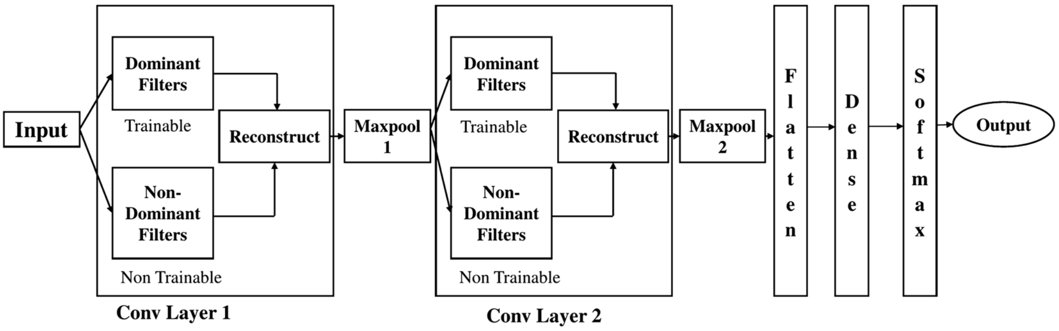


has the added advantage of achieving the same level of adversarial defense as the conventional approach [7], while re-training much less parameters, as will be shown in Section 4.

3.5. Model splitting and adversarial training

The existing approach for defense against adversarial attacks is to re-train the model with adversarial samples accompanied by correct class labels. Training the model with a certain form of attack gives it the necessary defense for that attack and weaker (to some extent) attacks [7], but the training also makes the model lose some of its initial performance capabilities. We hypothesize that re-training the parts of the model that are exploited the most during adversarial attacks should provide a balanced outcome on both requirements. We term the most susceptible filters, as described in the previous section, as dominant filters. A potential approach to address the vulnerability of these filters is to apply pruning techniques, where these filters could be dropped entirely. However, pruning risks the loss of critical feature representations, which could degrade the model's performance on benign data. Instead, we intend to re-train or fine-tune these filters, while keeping other parts of the model frozen, for defense against adversarial attacks. This can be accomplished with a split model, as shown in Figure 7. Note that, we are proposing an adversarial fine-tuning method. In the proposed split model, each convolutional layer is divided into two parallel layers — one containing the dominant filters (which are identified as the most vulnerable filters to adversarial perturbations), and the other containing the non-dominant filters, as shown in Figure 7. The weights of the trained base model are transferred to this new split model, allowing it to leverage prior knowledge about benign data, while targeting specific vulnerabilities. During the fine-tuning process, only the weights of the dominant filters are updated, while the non-dominant filters are kept frozen to preserve their original functionality. Afterwards, these parallel layers are reconstructed back into their original structure before proceeding to the pooling layer, ensuring the model's architecture is restored without disrupting its baseline performance. By targeting only the most vulnerable filters, this method enhances adversarial robustness, while avoiding unnecessary retraining of non-critical parts of the network. Additionally, it retains the model's pre-trained performance on benign data, as demonstrated in our experiments.

Figure 7
Split model for adversarial fine-tuning of dominant filters



We outline the structured process of model splitting and adversarial fine-tuning in Algorithm 2. The algorithm describes the initialization of the split model, separation of layers, adversarial fine-tuning, and final reconstruction of the model. Unlike conventional adversarial training, which modifies all model parameters, our method selectively updates only the most vulnerable filters. This leads to a more efficient defense mechanism with lower computational overhead. Furthermore, we ensure that the base model's adversarial training follows the method outlined in ref. [7], reinforcing robustness against specific attack types. Our experimental results validate the effectiveness of this selective fine-tuning strategy, striking a balance between adversarial defense and performance retention on benign inputs.

Algorithm 2: Model splitting and adversarial training

Step 1: Initialize Split Model

```
Initialize Split_CNN_Model identical to original CNN
```

Step 2: Split Convolutional Layers

```
for each convolutional layer l in Split_CNN_Model do
    Split into two parallel layers:
    One with dominant_filters (trainable)
    One with remaining filters (frozen)
    Merge outputs to reconstruct original feature map
end for
```

Step 3: Adversarial Fine-Tuning

```
Load pre-trained model weights
Compile model with RMSprop optimizer (ϵ = 1e–08) and categorical cross-entropy loss
```

Step 3.1: Prepare Training Dataset

```
Generate adversarial images
Concatenate adversarial images with benign images to create a new dataset
```

Step 3.2: Fine-Tuning Process

```
for each training batch (x,y) do
    Compute loss on benign and adversarial samples
    Back-propagate gradients only through dominant filters
    Update dominant filter weights
 end for
```

Step 3.3: Model Evaluation

```
Evaluate model on adversarial test images ($x_{adv}$)
Evaluate model on benign test images
```

3.6. Computational complexity

As discussed in Sections 3.4 and 3.5, our proposed selective adversarial fine-tuning approach involves two primary steps: i) identifying dominant filters, and ii) freezing non-dominant filters and selectively fine-tuning dominant filters. This methodology is designed to optimize computational efficiency, while maintaining robustness against adversarial attacks.

Identification of Dominant Filters. For identifying dominant filters, we generate benign-adversarial image pairs, extract feature maps from the convolutional layers, compute feature level differences, rank filters according to the differences, and finally average across image pairs. The computational cost of this step is therefore $O(LN + Lf\ log\ f + Lf\ d^2 + Lf\ d^2k^2 + \theta d^2N) \approx O(Lf\ d^2k^2 + \theta d^2N)$ for an L layer neural network with θ total parameters. Here, we assumed the input image dimensions to be $d \times d$, number of benign-adversarial image pairs N, number of filters f, and filter kernel size $k \times k$.

Adversarial Fine-tuning. Now, the CNN model training has a computational complexity of $O(Lf\ d^2k^2)$ [34]. Since we propose to fine-tune only α fraction of the filters, our selective adversarial fine-tuning approach has a computational complexity of $O(Lf\ d^2k^2 + \theta d^2N + \alpha Lf\ d^2k^2)$. It is evident that there is some overhead, i.e., $O(Lf\ d^2k^2 + \theta d^2N)$, for identifying the dominant filters. However, one can choose α based on their performance requirement and computational capability.

4. Experimental Results

As mentioned before, we generated adversarial images for the MNIST training set with $\epsilon = 0.3$ using both FGSM and PGD. After the proposed adversarial fine-tuning of the split model, and conventional adversarial training of the base model, we evaluate the models for both benign accuracy and adversarial accuracy. We recall that the accuracy on adversarial images with $\epsilon = 0.3$ for the base model was 21.26% for non-targeted FGSM and 0.58% for non-targeted PGD.

For comparison, we consider three models: a shallow model consisting of two convolution layers with 16 and 32 filters, respectively; a base model featuring two convolution layers with 32 and 64 filters, respectively; and a dense model comprising three convolution layers with 32, 64, and 128 filters, respectively. In Tables 6 and 7, we show the performance of the three models under consideration for conventional adversarial training against FGSM and PGD attacks respectively. As mentioned before, we evaluate the models for both benign accuracy and adversarial accuracy. The adversarial accuracy increases, as expected, with adversarial training for both PGD and FGSM. But there is a drop in the benign accuracy for both cases, as can be seen in the fourth column for both adversarial training against FGSM and PGD attacks. To address this, Ilyas et al. [15] proposed using a training set containing a mixture of benign and adversarial images. We follow this approach as well — we take 42,000 benign images and their adversarial counterparts, and perform adversarial training of the models. After adversarial training, the adversarial accuracy of the models over an average of five training's remains essentially the same for both FGSM and PGD attacks. For the base model, benign accuracy drops from 99.33% to 96.00% for FGSM training and from 99.33% to 87.79% for PGD training. If adversarial training is performed with a mixture of benign and adversarial training data, we can bring up the benign accuracy to satisfactory levels without having to sacrifice adversarial accuracy (see the fifth column). That is, for all models under consideration, the accuracy on benign images for both FGSM and PGD adversarial training can be attained near 99.00%.

In Tables 8 and 9, we show the performance of the models under consideration for our proposed adversarial fine-tuning training against FGSM and PGD attacks respectively. As we can observe from the table, our adversarial fine-tuning approach by splitting the model provides similar results as the conventional adversarial training, even though a smaller number of parameters were fine-tuned. As the conventional approach, performance of the proposed approach is better when the adversarial finetuning is done with a mixture of benign and adversarial

Table 6
Performance of the conventional adversarial training of the entire model (FGSM)

Model	Test image type	Before adversarial training	Adversarial training with FGSM	Adversarial training with benign + FGSM
Shallow	Benign	98.94%	93.00%	98.86%
	Adv (FGSM)	10.61%	98.60%	98.60%
Base	Benign	99.33%	96.00%	98.83%
	Adv (FGSM)	21.26%	98.95%	98.98%
Dense	Benign	99.36%	90.11%	98.67%
	Adv (FGSM)	11.64%	98.91%	98.75%

Table 7
Performance of the conventional adversarial training of the entire model (PGD)

Model	Test image type	Before adversarial training	Adversarial training with PGD	Adversarial training with benign + PGD
Shallow	Benign	98.94%	95.94%	99.04%
	Adv (PGD)	0.89%	98.78%	98.63%
Base	Benign	99.33%	87.79%	99.13%
	Adv (PGD)	0.58%	98.89%	98.86%
Dense	Benign	99.36%	95.68%	99.05%
	Adv (PGD)	1.20%	98.72%	98.59%

Table 8
Performance of the proposed adversarial fine-tuning of the split model (FGSM)

Model	Test image type	Adversarial fine-tuning with FGSM	Percent difference with conventional	Adversarial fine-tuning with benign + FGSM	Percent difference with conventional
Shallow	Benign	94.03%	+1.11%	98.77%	−0.09%
	Adv (FGSM)	98.77%	+0.17%	98.30%	−0.30%
Base	Benign	95.45%	−0.57%	98.91%	+0.08%
	Adv (FGSM)	98.60%	−0.35%	98.61%	−0.37%
Dense	Benign	64.67%	−28.26%	98.89%	+0.22%
	Adv (FGSM)	98.20%	−0.72%	98.00%	−0.76%

Table 9
Performance of the proposed adversarial fine-tuning of the split model (PGD)

Model	Test image type	Adversarial fine-tuning with PGD	Percent difference with conventional	Adversarial fine-tuning with benign + PGD	Percent difference with conventional
Shallow	Benign	93.93%	−2.10%	98.86%	−0.18%
	Adv (PGD)	98.15%	−0.64%	97.75%	−0.89%
Base	Benign	94.11%	+7.20%	99.01%	−0.12%
	Adv (PGD)	98.76%	−0.13%	98.56%	−0.30%
Dense	Benign	76.97%	−19.56%	98.01%	−1.05%
	Adv (PGD)	95.67%	−3.09%	95.05%	−3.59%

training data. For the base model, benign accuracy for both FGSM and PGD training is approximately 99.00%, and the adversarial accuracy is approximately 98.50% — this is essentially the same as the conventional adversarial training results on the base model.

In Tables 6–9, we observe that adversarial training with a mixture of benign and adversarial samples provides approximately 98% accuracy, regardless of the model type. This indicates that the dominant filters are indeed the most vulnerable portion of the model and securing just those filters provides as good a result as conventional adversarial training. However, our proposed adversarial fine-tuning is more computation-friendly, since it involves a smaller amount of trainable parameters. We argue that the advantage of the proposed approach becomes greater with denser and more complex networks containing many trainable parameters, as shown in the following.

Thus far, we chose the top 10% filters as dominant filters. We investigate the effect of choosing more dominant (and therefore, trainable) filters. As shown in Tables 10 and 11, performance does not change noticeably for either the PGD or the FGSM. The accuracy findings show the benign and adversarial accuracies for the three models when the percentage of trainable filters in the convolutional layers are increased. On the last column for both tables, first value corresponds to FGSM fine-tuning, second value corresponds to Benign + FGSM fine-tuning. We argue that with this combination of dataset and attack model, the vulnerability of the model lies mostly within the top 10% of the dominant filters, which once again proves that adversarial training of the entire network is somewhat wasteful.

In Table 12, we show the percentage of trainable parameters is reduced when using the proposed adversarial fine-tuning of the split model. For the dense model, more than half of the model does not require adversarial training for achieving a robust performance against adversarial attacks. More specifically, the proposed method reduces trainable parameters by 64.07% for the top 10% dominant filters, 56.93% for 20%, and 35.73% for 50%, significantly decreasing computational costs. This reduction translates into lower memory usage and faster training times, particularly for larger models, making the method scalable to real-world applications. Compared to conventional adversarial training, which involves retraining of all parameters, our approach achieves comparable robustness while requiring substantially fewer trainable parameters, as shown in Tables 8 and 9. These results remain consistent across varying levels of perturbation in the attacking

Table 10
Effect of dominant filter percentage for FGSM along with percent difference with 10% baseline

Dominant filters	Test image type	Adv. fine-tuning with FGSM	Adv. fine-tuning with benign + FGSM	Percent difference with 10% baseline
10%	Benign	95.99%	99.13%	–
	Adv (FGSM)	98.69%	98.53%	–
20%	Benign	96.33%	99.10%	+0.35%/−0.03%
	Adv (FGSM)	98.64%	98.52%	−0.05%/−0.01%
50%	Benign	95.92%	99.07%	−0.07%/−0.06%
	Adv (FGSM)	98.86%	98.77%	+0.17%/+0.24%

Table 11
Effect of dominant filter percentage for PGD along with percent difference with 10% baseline

Dominant filters	Test image type	Adv. fine-tuning with PGD	Adv. fine-tuning with benign + PGD	Percent difference with 10% baseline
10%	Benign	94.11%	99.01%	–
	Adv (PGD)	98.76%	98.56%	–
20%	Benign	93.40%	99.08%	−0.75%/+0.07%
	Adv (PGD)	98.70%	98.67%	−0.06%/+0.11%
50%	Benign	91.29%	99.17%	−3.00%/+0.16%
	Adv (PGD)	98.88%	99.00%	+0.12%/+0.45%

Table 12
Percentage of reduction in trainable parameters

Dominant filters	Shallow model	Base model	Dense model
10%	7.96%	14.33%	64.07%
20%	7.08%	12.81%	56.93%
50%	4.54%	8.05%	35.73%

images, highlighting the computational efficiency of the proposed method in contrast to traditional approaches.

4.1. Discussions

It is worth noting that our experimental findings can also be interpreted as an implicit ablation study of the proposed framework. In Section 3.4 (Figures 4–6), we showed that adversarial perturbations consistently exploit a small subset of convolutional filters across classes and perturbation magnitudes, effectively isolating the most critical vulnerable components. The subsequent experiments in Tables 10–12 further highlight that increasing the proportion of fine-tuned filters beyond this vulnerable subset does not meaningfully improve robustness, indicating that the identification mechanism itself, rather than the number of updated parameters, is the key contributor to performance gains. Moreover, by comparing FGSM-based and PGD-based fine-tuning in Tables 6–9, we effectively assess the impact of different adversarial objectives: PGD fine-tuning yields stronger robustness that also transfers to FGSM attacks, whereas FGSM fine-tuning provides weaker generalization. Together, these analyses clarify the contribution of each design choice: (i) filter-level vulnerability identification provides efficiency and robustness, and (ii) stronger adversarial objectives during fine-tuning enhance transferability. These results validate the central components of our framework without the need for additional experiments.

4.2. Remarks

Adversarial training may not be an one-shot approach for the model to be robust against future adversarial attacks. If the attackers get access to a model information; fully or partially; adversarial examples can be generated even if a model is trained against it. This is especially true for adversarial training against FGSM attacks, since FGSM is a one-step algorithm that computes the perturbation with a single step in the direction of the gradient of the loss function. It does not explore the gradient of the loss function in its entirety. Attackers can take advantage of this characteristic by computing new adversarial attacks the model is not trained against, if model information is revealed. Thus, a model trained against FGSM attacks may need further training if it is suspected that model information has been leaked. This underlines one advantage of our proposed split model fine-tuning, where we can achieve almost similar results using much lower computational burden. More specifically, for adaptive adversaries or transfer attacks in safety-critical settings, such as autonomous driving or healthcare, our proposed scheme would provide a computational advantage over conventional adversarial re-training. Additionally, PGD training provides better defense since it takes an iterative approach to the gradient descent in order to maximize the loss. A stronger adversarial training, such as, with PGD adversarial images, will not only secure the model against PGD attacks, but also provide protection against other weaker attacks (such as FGSM) to a certain extent.

4.3. Future works

Despite the promising results, our method has several limitations that warrant further exploration. First, the effectiveness of dominant filter identification may vary depending on the underlying network architecture. For example, convolutional networks, with their structured and localized receptive fields, lend themselves more naturally to filter-level vulnerability analysis, whereas attention-based models may not exhibit the same clear filter dominance patterns. Second, dataset characteristics such as inter-class similarity and noise levels can influence how consistently adversarial vulnerabilities manifest across filters, potentially affecting the reliability of selective fine-tuning. Finally, while our evaluation demonstrates robustness gains on the tested benchmarks, additional studies on larger-scale and more heterogeneous datasets are needed to assess the generalizability of the approach. Addressing these factors represents an important avenue for future research.

More specifically, implementing our proposed method with second order adversarial attacks (i.e., the Carlini-Wagner [33] approach) is deferred for future research. Additionally, interesting directions of future works could be i) to extend our work to larger and more complex datasets, such as CIFAR-10, CIFAR-100, and Tiny ImageNet, ii) applying the proposed approach to advanced architectures, including VGG and ResNet, and iii) benchmarking the proposed approach against state-of-the-art defense mechanisms. Last but not the least, exploring the transferability of identified dominant filters across models with similar structures could be of independent interest, such as transitioning from a model with layers of (16, 32) to one with (16, 32, 64). These directions will help establish the scalability and versatility of our approach in real-world scenarios.

5. Conclusion

Adversarial attacks take advantage of the excess capacity of the neural network models in such a way that makes subliminal adjustments to the inputs (imperceptible to humans), and thereby causes the model to make inaccurate predictions. Since neural network-based models are deployed in several critical applications, strong defense mechanisms against such adversarial attacks are rightfully warranted. However, existing approach for attaining model robustness against adversarial attacks is adversarial training, which typically provides defense against specific attack types and requires substantial computational resources. In this work, we showed that by algorithmically identifying specific vulnerable parts of the neural network model, and performing adversarial fine-tuning of those parts, we can attain the same level of performance as the conventional adversarial training. Our analysis reveals that only a small portion of the vulnerable components accounts for a majority of the model's errors caused by adversarial attacks. As such, we propose to selectively fine-tune these vulnerable components, which ensures significant computational-load savings. We empirically validate our proposed approach on the MNIST dataset, and demonstrate that our approach can achieve similar performance as the more resource-intensive conventional adversarial training method. Our results also demonstrate that robustness gains arise primarily from selectively fine-tuning adversarially vulnerable filters rather than retraining larger portions of the model, and from employing stronger adversarial objectives during fine-tuning. We note that a more efficient defense mechanism is crucial since neural network models are increasingly deployed in safety-critical and socially impactful applications — such as healthcare, finance, and autonomous systems, where adversarial attacks could lead to harmful consequences. Additionally, our proposed lightweight selective fine-tuning approach would certainly help reduce the computational and energy overhead of conventional adversarial training, contributing to more sustainable AI deployment.

Acknowledgement

The authors would like to express their sincere gratitude towards the Department of Electrical and Electronic Engineering (EEE) of Bangladesh University of Engineering and Technology (BUET) for providing support for research.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The data that support the findings of this study are openly available in github.io at https://doi.org/10.1109/5.726791, reference number [3].

Author Contribution Statement

Subah Karnine: Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Sadia Afrose:** Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Hafiz Imtiaz:** Conceptualization, Methodology, Formal analysis, Resources, Writing – original draft, Writing – review & editing, Supervision, Project administration.

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

[2] Kornblith, S., Chen, T., Lee, H., & Norouzi, M. (2021). Why do better loss functions lead to less transferable features. *Advances in Neural Information Processing Systems*, *34*, 28648–28662.

[3] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (2002). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

[4] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97. https://doi.org/10.1109/MSP.2012.2205597

[5] Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, Z. D., & Blundell, C. (2020). Agent57: Outperforming the Atari human benchmark. In *International Conference on Machine Learning*, *19*, 507–517.

[6] Goldwaser, A., & Thielscher, M. (2020). Deep reinforcement learning for general game playing. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*, 1701–1708. https://doi.org/10.1609/aaai.v34i02.5533

[7] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv Preprint:1706.06083*. https://doi.org/10.48550/arXiv.1706.06083

[8] Wu, H., Yunas, S., Rowlands, S., Ruan, W., & Wahlström, J. (2023). Adversarial driving: Attacking end-to-end autonomous driving. In *2023 IEEE Intelligent Vehicles Symposium*, 1–7. https://doi.org/10.1109/IV55152.2023.10186386

[9] Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., & Bengio, Y. (2021). Toward causal representation learning. *Proceedings of the IEEE*, *109*(5), 612–634. https://doi.org/10.1109/JPROC.2021.3058954

[10] Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2016). Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, 582–597. https://doi.org/10.1109/SP.2016.41

[11] Kurakin, A., Goodfellow, I. J., & Bengio, S. (2017). Adversarial machine learning at scale. In *International Conference on Learning Representations*, 1–17.

[12] Rozsa, A., Gunther, M., & Boult, T. E. (2018). Towards robust deep neural networks with BANG. In *2018 IEEE Winter Conference on Applications of Computer Vision*, 803–811. https://doi.org/10.1109/WACV.2018.00093

[13] Fu, Y., Yu, Q., Zhang, Y., Wu, S., Ouyang, X., Cox, D., & Lin, Y. (2021). Drawing robust scratch tickets: Subnetworks with inborn robustness are found within randomly initialized networks. *Advances in Neural Information Processing Systems*, *34*, 13059–13072.

[14] Tramèr, F., Behrmann, J., Carlini, N., Papernot, N., & Jacobsen, J. H. (2020). Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations. In *International Conference on Machine Learning*, 9561–9571.

[15] Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., & Madry, A. (2019). Adversarial examples are not bugs, they are features. *Advances in Neural Information Processing Systems*, 32.

[16] Tian, Y., Zhong, Z., Ordonez, V., Kaiser, G., & Ray, B. (2020). Testing DNN image classifiers for confusion & bias errors. *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 1122–1134. https://doi.org/10.1145/3377811.3380400

[17] Shen, M., Yu, H., Zhu, L., Xu, K., Li, Q., & Hu, J. (2021). Effective and robust physical-world attacks on deep learning face recognition systems. *IEEE Transactions on Information Forensics and Security*, *16*, 4063–4077. https://doi.org/10.1109/TIFS.2021.3102492

[18] Buckner, C. (2020). Understanding adversarial examples requires a theory of artefacts for deep learning. *Nature Machine Intelligence*, *2*(12), 731–736. https://doi.org/10.1038/s42256-020-00266-y

[19] Block, A., Foster, D. J., Krishnamurthy, A., Simchowitz, M., & Zhang, C. (2023). Butterfly effects of SGD noise: Error amplification in behavior cloning and autoregression. *arXiv Preprint: 2310.11428*. https://doi.org/10.48550/arXiv.2310.11428

[20] Wang, R., Zeng, S., Wu, W., Jia, Y., Ng, W. W., & Wang, X. (2024). On the adversarial robustness of hierarchical classification. In *2024 IEEE International Conference on Systems, Man, and Cybernetics*, 4358–4364. https://doi.org/10.1109/SMC54092.2024.10831681

[21] Weng, C. H., Lee, Y. T., & Wu, S. H. B. (2020). On the trade-off between adversarial and backdoor robustness. *Advances in Neural Information Processing Systems*, *33*, 11973–11983.

[22] Meng, D., & Chen, H. (2017). Magnet: A two-pronged defense against adversarial examples. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 135–147. https://doi.org/10.1145/3133956.3134057

[23] Yang, P., Chen, J., Hsieh, C. J., Wang, J. L., & Jordan, M. (2020). Ml-loo: Detecting adversarial examples with feature attribution. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*, 6639–6647. https://doi.org/10.1609/aaai.v34i04.6140

[24] Issaoui, H., Eladel, A., Zouinkhi, A., Zaied, M., Khriji, L., & Nengroo, S. H. (2024). Defending CNN against FGSM attacks using beta-based personalized activation functions and adversarial training. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3432773

[25] Ko, K., Kim, S., & Kwon, H. (2025). Selective audio perturbations for targeting specific phrases in speech recognition systems. *International Journal of Computational Intelligence Systems*, *18*(1), 103. https://doi.org/10.1007/s44196-025-00844-1

[26] Ko, K., Kim, S., & Kwon, H. (2023). Multi-targeted audio adversarial example for use against speech recognition systems. *Computers & Security*, *128*, 103168. https://doi.org/10.1016/j.cose.2023.103168

[27] Luo, S., Yang, H., Xin, Y., Yi, M., Wu, G., Zhai, G., & Liu, X. (2025). Tr-pts: Task-relevant parameter and token selection for efficient tuning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4360–4369.

[28] Peng, G., Yang, Y., Zhang, D., Xie, X., Jiang, A., & Ding, F. (2025). Dynamic layer-wise strategy for parameter-efficient fine-tuning. *Proceedings of the 2025 8th International Conference on Software Engineering and Information Management*, 165–169. https://doi.org/10.1145/3725899.3725924

[29] Ziyadinov, V., & Tereshonok, M. (2023). Low-pass image filtering to achieve adversarial robustness. *Sensors*, *23*(22), 9032.

[30] Kraidia, I., Ghenai, A., & Belhaouari, S. B. (2024). Defense against adversarial attacks: robust and efficient compressed optimized neural networks. *Scientific Reports*, *14*(1), 6420. https://doi.org/10.1038/s41598-024-56259-z

[31] Zhao, J., Xie, L., Gu, S., Qin, Z., Zhang, Y., Wang, Z., & Hu, Y. (2025). Universal attention guided adversarial defense using feature pyramid and non-local mechanisms. *Scientific Reports*, *15*(1), 5237. https://doi.org/10.1038/s41598-025-89267-8

[32] Liao, W., Liu, Z., Shen, M., Chen, R., & Liu, X. (2024). Apr-net: Defense against adversarial examples based on universal adversarial perturbation removal network. *IEEE Transactions on Artificial Intelligence*. https://doi.org/10.1109/TAI.2024.3504478

[33] Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 39–57. https://doi.org/10.1109/SP.2017.49

[34] Shah, B., & Bhavsar, H. (2022). Time complexity in deep learning models. *Procedia Computer Science*, *215*, 202–210. https://doi.org/10.1016/j.procs.2022.12.023

How to Cite: Karnine, S., Afrose, S., & Imtiaz, H. (2025). Efficient Defense Against First Order Adversarial Attacks on Convolutional Neural Networks. *Artificial Intelligence and Applications*. https://doi.org/10.47852/bonviewAIA52025957