CoopHance: Cooperative Enhancement for Robustness of Deep Learning Systems

Quan Zhang
BNRist, School of Software, Tsinghua University
China

Yongqiang Tian*
University of Waterloo
Canada

Yifeng Ding
University of Illinois at Urbana-Champaign
United States

Shanshan Li
National University of Defense Technology
China

Chengnian Sun
University of Waterloo
Canada

Yu Jiang*
BNRist, School of Software, Tsinghua University
China

Jiaguang Sun
BNRist, School of Software, Tsinghua University
China

ABSTRACT

Adversarial attacks have been a threat to Deep Learning (DL) systems to be reckoned with. By adding human-imperceptible perturbation to benign inputs, adversarial attacks can cause the incorrect behavior of DL systems. Considering the popularity of DL systems in the industry, it is critical and urgent for developers to enhance the robustness of DL systems against adversarial attacks.

In this study, we propose a novel enhancement technique for DL systems, namely CoopHance. CoopHance leverages two specifically customized components, *Regulator* and *Inspector*, to cooperatively enhance the DL systems' robustness against adversarial examples with different distortions. Regulator can purify adversarial examples with low or moderate distortions, while Inspector is responsible for detecting these adversarial examples with high distortion by capturing the abnormal status of DL systems. Our evaluation using various attacks shows that, on average, CoopHance can successfully resist 90.62% and 96.56% of the adversarial examples that are generated for the unprotected systems on CIFAR-10 and SVHN datasets separately, which is 188.14% more effective than five state-of-the-art enhancement techniques, including Feature Squeeze, LID, SOAP, Adversarial Training, and MagNet. Meanwhile, when attackers generate new adversarial examples on the enhanced systems, CoopHance can reject 78.06% of attacks, which outperforms the best of five enhancement techniques by 82.71% on average.

CCS CONCEPTS

• **Security and privacy → Domain-specific security and privacy architectures**; • **Computing methodologies → Neural networks**.

KEYWORDS

Robustness Enhancement, Deep Learning System

ACM Reference Format:
Quan Zhang, Yongqiang Tian, Yifeng Ding, Shanshan Li, Chengnian Sun, Yu Jiang, and Jiaguang Sun. 2023. CoopHance: Cooperative Enhancement for Robustness of Deep Learning Systems. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '23), July 17–21, 2023, Seattle, WA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3597926.3598093

1 INTRODUCTION

Deep learning (DL) systems have demonstrated their superior performance after being deployed into various domains, including many security-sensitive domains, such as medical image processing, self-driving, and face recognition [5, 23, 34]. However, DL systems are prone to vulnerabilities, as their key component, deep neural networks (DNN), can be easily misled by attackers with adversarial examples. By adding imperceptible adversarial perturbations to inputs [10, 14, 19, 27, 39, 40], attackers can cause abnormal behavior of DL systems and produce serious damage. Therefore, during the development and maintenance of DL systems, developers urgently need an effective and efficient method to enhance their robustness.

The researchers have explored various methods to improve the robustness of DL systems. Among them, adversarial training (AdvTrain) is known as the most effective technique [10, 25], but it causes degradation of accuracy and requires extensive computation resources and a bag of training tricks during training [30]. However, in some situations, such accuracy degradation and development efforts are unaffordable for developers, such as maintaining a frequently upgraded DL system [30, 47]. Thus, we aim to find a DL system enhancement technique that owns competitive effectiveness with AdvTrain and is more convenient to develop and deploy.

In addition to AdvTrain, there are two categories of techniques that are intended to enhance DL systems with lightweight or independent modules. Techniques of the first category are designed to enhance DL systems with detection-based modules, which are usually lightweight and easy to build and deploy [24, 26, 43]. These detection modules rely on some specific properties of adversarial examples to distinguish them. Nevertheless, they are prone to adaptive adversarial attacks, in which attackers craft special adversarial examples to hide corresponding properties and bypass

*Yu Jiang and Yongqiang Tian are corresponding authors

ISSTA '23, July 17–21, 2023, Seattle, WA, USA

ACM ISBN 979-8-4007-0221-1/23/07.
https://doi.org/10.1145/3597926.3598093

detection [7, 8]. Techniques of the second category purify the adversarial examples with preprocessing-based modules [12, 36, 44], which are usually built as independent modules and can be deployed to different DL systems with the same training data. However, they are demonstrated as gradient obfuscation that cannot truly fix the vulnerabilities of DNN models but can only complex the attack process [3]. When facing adversarial perturbations with higher distortion, they may lose their effectiveness.

Recently, researchers start to propose enhancements by combining different existing techniques [1, 26]. However, just as He et al. demonstrated, simple ensembles of weak methods are not robustness [17]. These enhancement techniques are soon found to be fragile in some situations [8], as they combine several existing techniques that own similar weaknesses and cannot cooperate effectively. Therefore, the cooperation of different techniques is crucial to a valid enhancement, which is a challenging problem. The first challenge is to find a proper cooperation mechanism to combine different techniques effectively. Moreover, since existing techniques are designed without considering interaction with others, they are not capable of effective cooperation.

In this paper, we propose CoopHance, a novel cooperative enhancement framework. To address the first challenge, in CoopHance, we design a cooperation mechanism based on the distortion of adversarial examples. It is found that for one type of adversarial attack, the effectiveness and stability of adversarial examples are proportional to their distortions. For example, as shown in Figure 1, by adding adversarial perturbation, attackers can confuse the DNN model to misclassify an input of class "dog" to "cat". When added to low-distortion perturbation, the crafted adversarial examples are weak and easy to purify, as they are very close to the decision boundary of two classes and are unstable. As the distortion of adversarial improves, they can go deeper in the "Cat"'s classification region, so they become stronger and are harder to purify. However, with high distortion, adversarial examples are very different from benign ones, resulting in anomalies in DNN models. Thus, CoopHance contains two modules: a preprocessing module called *Regulator* and a detection module called *Inspector*, which can cover the weaknesses of each other. *Regulator* should purify adversarial examples but may fail to handle very strong ones. In contrast, *Inspector* is responsible for detecting strong adversarial examples with high distortion, but it is insensitive to weak ones.

Then, to guarantee effective cooperation, we specially designed two modules, *Regulator* and *Inspector*. Specifically, *Regulator* is a UNet Neural Network with residual connections and confusion layers, which is trained in a simulated situation of adversarial attack and can purify very strong adversarial examples. As for *Inspector*, constructed as a per-category logistic regression model, it can accurately capture the inconsistency between the predicted category and internal activations to alert the developers. Moreover, to meet the demand for efficient deployment, the *Regulator* is designed to be independent of DL systems and can be deployed to the different DNNs trained from the same dataset, and *Inspector* can be built conveniently. With the above two components, CoopHance can effectively improve the robustness of DL systems against various adversarial attacks with low deployment costs.

To show the effectiveness of CoopHance, we evaluate it with six diverse attack methods [9, 10, 13, 25, 28, 32], including the

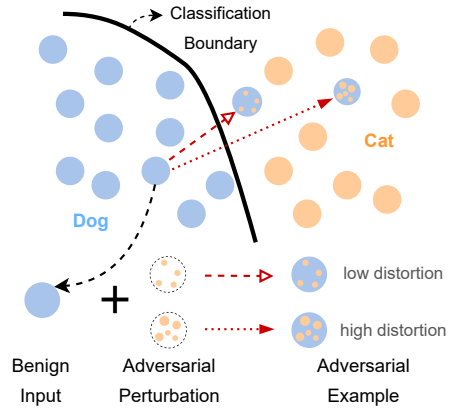


Figure 1: Adversarial perturbations can guide a normal input from the "Dog" to cross the classification boundary and go into the classification region of the "Cat". Adversarial examples with higher distortion usually go deeper into the classification region of the "Cat".

AutoAttack [10], which is widely used for DL system robustness testing. For a comprehensive evaluation, we include two datasets, i.e., SVHN [29] and CIFAR-10 [22], and three DNN architectures, i.e., VGGNet [37], ResNet [16], and DenseNet [18]. We compare CoopHance with five state-of-the-art enhancement methods [24–26, 36, 43]. CoopHance achieves competitive performance in both two following scenarios. First, for adversarial examples that can confuse DL systems without enhancement, on CIFAR-10 and SVHN, 82.80~98.25% and 92.85~99.75% of such adversarial examples are disabled by CoopHance correspondingly. Compared with existing methods, CoopHance averagely outperforms them by 142.29% and 233.99% on two datasets. Second, enhanced with CoopHance, DL systems should have high robustness to incoming adversarial attacks. In this situation, CoopHance averagely improves 78.17% on resistance rate and 12.08% on AUC score compared with the best of existing methods, blocking 72.78% and 78.10% of adversarial examples with 91.30% and 94.78% of AUC on CIFAR-10 and SVHN, To investigate the cooperation of *Regulator* and *Inspector*, we analyze how they contribute to the overall enhancement performance separately and demonstrate that these two components can cooperate closely. Even in the adaptive attack, attackers cannot break two modules in CoopHance at the same time. It is worth mentioning that though CoopHance does not improve the robustness of DNNs directly, evaluation results show that it can enhance the robustness of whole DL systems as a systematic method.

Our contributions are summarized as follows:

- We proposed CoopHance that systematically combines *Regulator* and *Inspector* to enhance the robustness of DL systems.
- According to the cooperation mechanism, we implemented CoopHance with a well-designed *Regulator* and *Inspector*.
- We conducted extensive experiments on six attacks to demonstrate that CoopHance outperforms the other five baselines in terms of effectiveness.

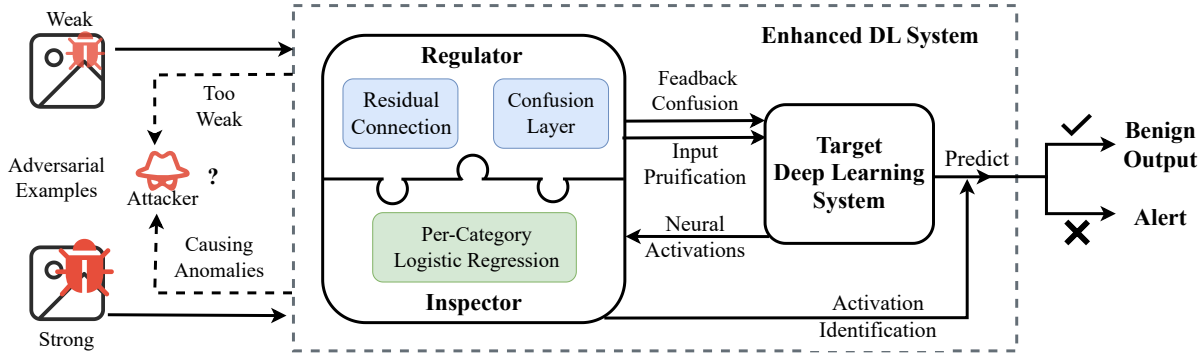


Figure 2: Overview of CoopHance. *Regulator* regulates the input image to purify adversarial examples and confuses the feedback gradient to hinder adversarial example generation. *Inspector* can identify whether the input image is malicious based on the neuron activations of the target system. With them, CoopHance can either help the target system give the benign output or alert developers to potential attacks., leaving attackers two attack objectives in contradictory directions.

2 PRELIMINARIES

2.1 Adversarial Attack

Recently, many attack and testing methods are proposed to generate adversarial examples to cause incorrect behaviors of DL systems [9, 13–15, 19, 25, 28, 32]. As shown in Figure 1, by adding human-imperceptible **adversarial perturbations**, attackers craft **adversarial examples** that can cross the classification of boundary and go into the classification region of another class [9, 13, 46]. Normally, attackers tend to find the perturbation with minimal distraction to craft adversarial examples, since such perturbations are imperceptible to humans. With low distortion, **weak adversarial examples** can just cross the classification boundary but cannot go deeper, so some interrupts on them can make them go back to their original classification regions and defeat attacks. To improve attacks' stability, attackers improve the distortion to build the **strong adversarial examples** that can go deeper into other classification regions. However, high distortion leads to obvious differences between adversarial examples and benign inputs, causing anomalies in DNNs' internal activation. Considering that the distortion of adversarial examples is controlled by attackers depending on their demands, developers must keep DL systems robust in all situations.

In this paper, we mainly concentrated on white-box adversarial attacks, which utilize DNN models' gradients as feedback to guide the generation of adversarial examples. Specifically, **FGSM** [13] modifies the image with a large fixed step length of ϵ in the direction of the gradient, and it usually induces large distortion. **PGD** [25] iteratively executes the process of FGSM attack until attack successes. In each iteration, it uses a smaller step length. Therefore, PGD has relatively moderate distortion with high effectiveness. **AutoAttack** [10] is an adaptive attack. With the automatic attack parameters selection, it is widely used for DL system robustness evaluation. **JSMA** [32] is a targeted attack, and it edits several pixels that have the highest probability to achieve the attack. Although the number of modified pixels is limited, large changes on several pixels still cause high distortion. **Deepfool** [28] calculates the minimal distance between input and its classification decision boundary. Then, it modifies the input with that distance to generate the adversarial examples with minimal distortion. **C&W** also aims to get the minimum distortion. It designs a special loss function and uses the optimizer to optimize the adversarial perturbation directly.

2.2 Robustness Enhancement

To address the threat of adversarial attacks, researchers make great efforts to enhance the robustness of DL systems. **Adversarial training (AT)** [25] is thought to be the best technique, as it improves the robustness of DNN models directly. It generates the adversarial examples continuously and uses them as the training data to train the DNN. However, training on adversarial examples requires extensive computation resources and causes significant performance drops, which is unacceptable for developers in some situations. In this paper, we aim to design an enhancement technique that is competitive with AT but is more convenient to develop and deploy.

For efficient deployment, researchers are exploring preprocessing-based and detection-based enhancements. Normally, these two types of enhancements are either lightweight or independent of DL systems, so they can be easily built and deployed to multiple DL systems correspondingly. For example, **Feature Squeeze (FS)** [43] and **LID** [24] are two detection-based modules. FS utilizes a variety of squeezers to compress the input and then observe the distance of the prediction confidence between the original and squeezed inputs. The ones that change a lot are identified as adversarial examples. LID catches the adversarial examples with the Local Intrinsic Dimensionality as the metric. They both detect by monitoring certain metrics without altering or retraining original DNNs. **SOAP** [36] purifies adversarial examples to benign inputs to guarantee the normal execution of DL systems. During purification, it leverages the label-independent nature of self-supervised signals to remove adversarial perturbation. **MagNet (MN)** [26] first uses several autoencoders to purify the inputs. Meanwhile, it utilizes the reconstruction error as a metric to detect adversarial examples. Though combining two enhancement modules, it has low robustness against strong adversarial examples, as its two modules both rely on the reconstruction of autoencoders.

2.3 Enhancement Scenario

Normally, attackers may implement the attack in the following two scenarios. For clear illustration, DL systems without any enhancement modules are called **target systems**, and the enhanced DL systems are referred to as **enhanced systems**.

Scenario A: Resist Existing Adversarial Examples. It is possible that a DL system was deployed without enhancement, and attackers implemented the attack and gained a set of dangerous adversarial examples. Then, developers enhance and re-deploy the DL system, but attackers are not aware of the enhancement modules and attack the DL system with previously generated adversarial examples. Thus, enhanced system should own high robustness against these existing adversarial examples so that they cannot cause illegal behaviors of DL systems anymore.

Scenario B: Resist New Adversarial Examples. The malicious users directly leverage diverse attack techniques to attack the enhanced DL systems to find new adversarial examples ranter than using examples collected from the unprotected target system. Enhanced systems should resist these adversarial attacks, so attackers will fail to generate effective adversarial examples.

3 METHODOLOGY

Figure 2 shows the overview of CoopHance. Specifically, CoopHance consists of two components, *Regulator* and *Inspector*. *Regulator* is a special-designed Unet-like Neural Network with Residual Connection and Confusion Layer. It purifies the inputs (e.g., adversarial examples) and interrupts the feedback gradients, so if attackers do not enlarge the distortion of adversarial examples, they cannot influence the enhanced system. However, once attackers attack the enhanced system with strong adversarial examples, these strong adversarial examples will cause abnormal activations in DNNs, which can be caught by the *Inspector*. Therefore, requiring attackers to adjust their adversarial examples to two opposite attack objectives, CoopHance can effectively enhance the robustness of DL systems.

CoopHance is designed based on the close cooperation of two components, whose responsibilities in different enhancement scenarios are different. In ***Scenario A***, when CoopHance needs to resist an existing adversarial example, *Regulator* first processes this example with the objective to purify the adversarial perturbation, which is the responsibility of the residual connections and confusion layers in *Regulator* (see Section 3.1). Then, the processed input is fed into the target DL system for inference. Since the perturbation on the input has been removed, the target system executes normally. However, it is possible that the input regulated by *Regulator* is still an effective adversarial example, because the original adversarial example goes deeper into other categories' classification regions and is hard to purify. In this case, *Inspector* leverages the model's internal status, i.e., neuron activations, to catch the abnormal behaviors of the target system and further alert the developers.

Moreover, as illustrated in ***Scenario B***, attackers also generate new adversarial examples on enhanced system directly. Normally, with the guidance of DL system's gradient, an attacker can find the direction where they can lead a benign input to cross the classification boundary with minimal perturbation. Thus, to complex the generation of the adversarial example, utilizing the confusion layers, *Regulator* adds noise on the feedback gradient to confuse attackers. Misled by the *Regulator*, attackers will add the perturbations in the wrong direction. As a result, they may fail to craft the adversarial example, or they need to make more efforts and craft an adversarial example with higher distortion. However, similar to the situation in *Scenario A*, although that stronger adversarial example can cause the misclassification on the target system, they also cause more internal abnormal activations. These anomalies will trigger the *Inspector*, so they still cannot damage the enhanced system. Please note that CoopHance does not require the *Regulator* to completely prevent the generation of adversarial examples, which is a non-trivial task [3]. Instead, even if *Regulator* fails to do so, it can still enlarge the distortion of adversarial examples, so that the *Inspector* can effectively catch the abnormal status of DNNs.

3.1 Regulator

According to the cooperation mechanism, the *Regulator* is responsible for purifying the adversarial examples and hindering the generation of adversarial examples. To meet the responsibility, *Regulator* should accomplish the following three goals: First, *Regulator* must remove the malicious perturbation without influencing the benign inputs. For this goal, we adopt a UNet-like structure with **residual connections**, which helps *Regulator* concentrate on detailed features of inputs to distinguish benign features and malicious perturbations and remove malicious ones. Second, *Regulator* must complex the generation process of adversarial examples. It is a hard problem since we find attackers can utilize vulnerabilities of preprocessing modules to enlarge adversarial perturbations during preprocessing, so crafting adversarial examples becomes easier for them. To avoid facilitating attackers, *Regulator* utilizes the **confusion layers** and **adaptive noising training** to improve its robustness and confuse attackers with interrupted feedback. Last, *Regulator* should be easy to deploy. To meet this end, *Regulator* is designed to be independent of DL systems, and thus it can be deployed to different systems trained on the same dataset without complex adaptation. With the above design, *Regulator* is capable of meeting its responsibility and enhancing the DL system via purification.

3.1.1 Structure of Regulator. Figure 3 shows the structure of *Regulator*. Intuitively, *Regulator* is an UNet Neural Network [33]. As a variant of autoencoder, it aims to compress the input image with random noise into a hidden vector and generally reconstruct the vector back to the original clean image. During this process, *Regulator* must learn to avoid encoding noise and adversarial perturbations and only encode the feature of the original input into the hidden vector. Then, based on such a hidden vector, only benign features can be reconstructed, and therefore the input is purified. As a result, the DL system enhanced with *Regulator* can work on inputs without adversarial perturbation and make correct inferences.

To better purify adversarial examples, inspired by ResNet [16], we add **residual connections** on the autoencoder to help it better handle the detailed features of inputs. Without residual connections, the encoder of a simple autoencoder needs to encode the principal features first, followed by the detailed features, leading to insufficient learning of benign feature encoding. As a consequence, the autoencoder cannot precisely identify the benign features and adversarial perturbations, so part of adversarial perturbations is

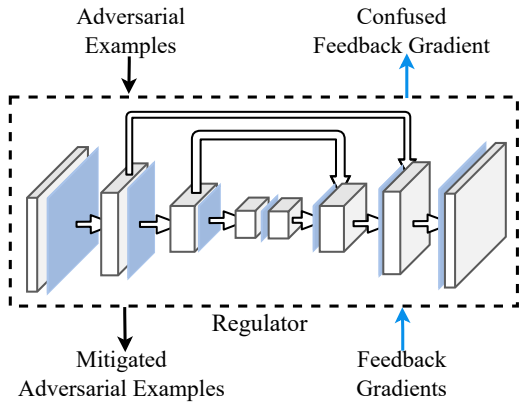


Figure 3: Structure of *Regulator*. Its main structure is an Unet Neural Network. Arrows in the inverted U-shape are Residual Connections. Blue rectangles are Confusion Layers.

also encoded by mistake. To mitigate this problem, inspired by existing works [16, 33], we add residual connections as a shortcut to pass some principal features directly to the decoder. In this way, the autoencoder only needs to learn the residual information, which represents the detailed features of the inputs. In the end, it can effectively encode details of benign inputs and reconstruct them without adversarial perturbations.

Moreover, as we mentioned before, attackers can craft new adversarial examples on enhanced system, during which they may confuse the preprocessing modules to avoid purification or even enlarge perturbation, so the **confusion layer** is proposed to improve *Regulator*. In detail, since each input is first processed by *Regulator* and then fed to the target system, *Regulator* needs to take part in backpropagation when calculating the gradients. In this case, the feedback gradients contain the information of *Regulator*, which may tell attackers how to confuse *Regulator*. To address this issue, the confusion layers are attached to some layers of the *Regulator* and add adaptive noise on activations of attached layers. As the blue rectangles in Figure 3 show, a confusion layer is attached after each down-sample layer and before each up-sample layer.

The confusion layers can improve the *Regulator* in the following two aspects: First, confusion layers can simulate the fluctuation caused by adversarial examples to improve *Regulator*'s ability during the training. As adversarial examples are different from benign inputs, they usually cause some fluctuations in DNNs' internal activations, finally leading to incorrect behaviors of DNNs. By simulating such fluctuations during the training of *Regulator*, confusion layers improve *Regulator*'s resistance to adversarial examples. Thus, when inferring on an adversarial example, *Regulator* can avoid being utilized by attackers. To better simulate the fluctuations, confusion layers add the Gaussian noises to DNNs' internal activations, as these activations are usually assumed to be the Gaussian distribution [20]. Second, these noises take part in the computation of feedback gradients and can confuse the feedback gradients. Without the guidance of precise gradients, attackers cannot achieve their attack objective with minimal perturbations, so they may fail to find adversarial examples, or they have to craft adversarial examples with larger distortion.

Algorithm 1: Adaptive Noising Training Strategy

Input: X: Training data set
Input: X_e: Adversarial examples
Input: *layers*: layers that need to measure fluctuations
Output: G: Well-trained *Regulator*

1 $i := 0$ // loop for training sample model
2 **while** $i{+}{+} \leq e$ **do**
3 $G' := update(G', X)$
4 **if** $i\text{-}1 = e \times 0.6$ **then**
  // save middle weights
5 $w := dump\_weights(G')$
// calculate $\mu_{i,j}$ and $\sigma_{i,j}$.
6 **foreach** i *in layers* **do**
7 **foreach** j *in channels* **do**
8 **foreach** x^k, x_e^k *in* X, X_e **do**
9 $distance[k] := G'_{i,j}(x^k) - G'_{i,j}(x_e^k)$
10 $\mu_{i,j} = cal\_average(distance)$
11 $\sigma_{i,j} = cal\_variance(distance)$
// load middle weights to untrained Regulator
12 $G = load\_weight(G, w)$
13 $G := init\_confusion\_layers(G, \mu, \sigma)$
14 $i = 0$
15 **while** $i{+}{+} \leq e \times 0.4$ **do**
16 $G := update(G, d)$

3.1.2 Adaptive Noising Training. Since confusion layers need to simulate the fluctuations caused by adversarial examples, for which a proper degree of simulated fluctuations is extremely important, we further design a novel **adaptive noising training** for *Regulator*. Specifically, when adding the Gaussian noise, each confusion layer has two hyper-parameters μ and σ, which should be set adaptively. To determine these hyper-parameters, a *Sample Model*, which owns the same structure of *Regulator* but does not have confusion layers, is trained to measure the fluctuation of activations and calculate these two hyper-parameters. By evaluating the reference model on a set of adversarial examples, we could get the precise μ and σ for each confusion layer. Then, during the training of *Regulator*, we add the fluctuation according to these two hyper-parameters, so *Regulator* can be robust to the fluctuation of activations. Only huge fluctuations caused by adversarial examples with large enough distortion can break *Regulator* to attack the target system.

The detail of the training strategy is shown in Algorithm 1. With four inputs, the benign inputs X, adversarial examples X_e, and the layers in *Regulator* that need to measure the fluctuation *layers*, the algorithm outputs a well-trained *Regulator* G. To train an effective *Regulator*, a well-trained *Sample Model* G' is first required to measure fluctuations (Line 2~5). With a well-trained G', we observe how each layer reacts to adversarial examples. Specifically, after feeding adversarial example $x_e^k \in X_e$ and corresponding benign input $x^k \in X$ into G' separately, we measure the distance between activations of two inputs on each channel of a layer as the fluctuation, just as shown in Line 6~9. $G'_{i,j}$ in Line 9 returns the activation

of channel j in layer i. Then, in Line 10~11, the algorithm calculates the μ and σ of distance for each channel. In addition, the μ and σ are measured based on various kinds of adversarial examples to reflect the universal fluctuation, so *Regulator* owns great generalization and is not specific for any one type of attack. In the end, confusion layers of G can be set with adaptive hyper-parameters.

With proper hyper-parameters, we could start to train *Regulator*. However, due to the random initialization, two models of the same structure are usually different. Hence, the statistic collected from one model may not fit another one. To address this issue, as presented in Line 12, the G is not trained from scratch but from the weights w that are dumped during the training of G' (Line 4~5). Weights w is dumped when G' is trained for 60% of total epochs, because, at this time, G' has converged to a relatively stable state and is going to converge to the final G'. Now, fluctuations collected on well-trained G' can represent the fluctuations on G with weights w. Finally, the algorithm loads the weights w, initializes confusion layers with proper μ and σ, and trains the *Regulator* (Line 12~16).

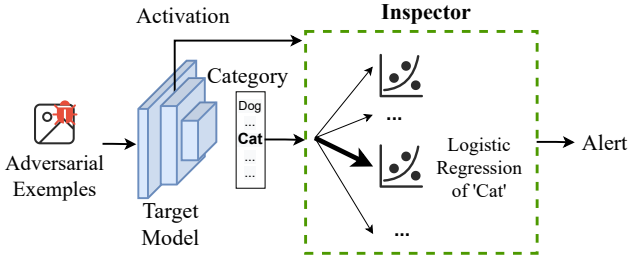


Figure 4: Structure of *Inspector*. It leverages per-category logistic regression models to catch adversarial examples.

3.2 Inspector

Another crucial component of CoopHance is *Inspector*, which is responsible to monitor the DNNs' internal activations and catch the adversarial examples that cause anomalies. Intuitively, each neuron in a DNN is activated when seeing specific features [35, 45]. Ideally, the DNN should classify based on both fragile features, including texture and color, and robust features like the border of an object. However, current DNNs rely more on fragile features, which can be easily interrupted by attackers to implement adversarial attacks [21]. In the contrast, it is hard to interrupt robust features, which keep activating corresponding neurons even though the DNN has been confused. Therefore, when DNN processes adversarial examples, only the neurons that reflect the fragile features are confused, leading to an inconsistency in DNN's activations. This inconsistency serves as the foundation for the detection capabilities of *Inspector*. More specifically, *Inspector* constructs a binary classifier, utilizing a logistic regression model, for each category within the classification task. As shown in Figure 4, given an input that is classified as "Cat" by target system, *Inspector* leverages the logistic regression model trained for category "Cat" to determine whether this input is adversarial or not. These logistic regression models are fed with DNN's internal activations and check if the abnormal inconsistency exists. If attackers utilize stronger adversarial examples, such anomalies will be more obvious and easier caught. Thus, *Regulator* can meet its responsibility to enhance DL systems against strong adversarial examples.

Besides the design of *Inspector*, using activations from which layer of DNN is also critical to the enhancement performance. It is found that activations of shallow layers that are near the inputs usually reflect more realistic features, like corners, borders, and colors, and activations of deep layers that are near the output layer contain more abstract features that are highly related to the final prediction [45]. Therefore, activations from deep layers are not suitable for our *Inspector*, as they are highly influenced by the adversarial examples and lead to misclassification. In contrast, shallow layers' activations contain the robust features of the original class, which is better for *Inspector*'s detection. Hence, we select the layers that are down-sampled only one time, which exists in most of the widely used DNN models. Moreover, the linear regression model is simple and very easy to train, so *Regulator* can be deployed to various DL systems with few adaption costs.

4 EVALUATION

We evaluate CoopHance from the following perspectives:

RQ1 What is the effectiveness of CoopHance in *Scenario A*, i.e., resisting the existing adversarial examples?
RQ2 What is the effectiveness of CoopHance in *Scenario B*, i.e., resisting new adversarial examples?
RQ3 What is the contribution of two components, *Regulator* and *Inspector*, towards the enhancement effectiveness?
RQ4 Can CoopHance resist specially designed adaptive attacks?

4.1 Experiment Setup

4.1.1 Metrics. We use two metrics to compare CoopHance with baselines. The first one is the **resistance rate** that measures the ratio of successfully resisted attacks with respect to the number of total attacks. As some enhancements have false positive examples, we need to set a specific False Positive Rate (FPR) as the threshold to calculate the resistance rate, which is set as 0.05 by following the FS [43]. The second metric is Area Under Curve (**AUC**) value, which evaluates the overall performance with different FPR. Thus, the enhancement methods that have no FPR cannot calculate the AUC, including AdvTrain [25] and SOAP [36]. These two metrics are widely used in other methods' evaluation [24, 26, 43].

4.1.2 Attack Setup. To evaluate the enhancement ability, we use six different adversarial attacks, including FGSM [13], JSMA [32], Deepfool [28], C&W [9], PGD [25], and AutoAttack [10]. Among all attacks, JSMA is a targeted attack and others are untargeted. The implementations of all attacks follow that of the original authors except JSMA, which is achieved with Cleverhans library [31]. For Deepfool and JSMA, we follow their default settings [28, 32]. The settings of FGSM and PGD follow that of LID [24]. However, on FGSM, LID sets a too large ϵ as 0.05, on which CoopHance achieves 100% resistance rate (shown in Figure 5a), so we set it as 5/255. For AutoAttack [10], a default setting of ℓ_∞ attack with $\epsilon = 8/255$ are utilized. On each dataset and attack method, we generate 2,000 adversarial examples on training data to train enhancement methods and generate 2,000 on test data for evaluation.

Table 1: Effectiveness of CoopHance and other baselines in *Scenario A*. The results in bold are the best results. Rows of "DF" and "Auto" are the results of resisting Deepfool and AutoAttack correspondingly.

Attack	Resistance Rate (FPR = 0.05)						AUC				Resistance Rate (FPR = 0.05)						AUC			
	FS	LID	SOAP	MN	AT	CoopHance	FS	LID	MN	CoopHance	FS	LID	SOAP	MN	AT	CoopHance	FS	LID	MN	CoopHance
CIFAR-10																				
Model	ResNet										VGGNet									
PGD	75.00	21.20	3.60	81.00	82.95	**89.90**	95.95	77.87	90.06	**96.50**	31.85	28.40	9.20	68.15	75.30	**87.90**	93.04	82.52	81.24	**96.01**
FGSM	20.20	15.25	29.65	61.75	82.70	**92.35**	76.73	67.65	81.81	**98.52**	25.85	17.40	21.65	41.15	74.50	**83.15**	74.70	71.89	65.98	**96.34**
JSMA	73.00	35.25	54.00	83.85	71.15	**98.25**	96.21	84.11	93.40	**99.55**	69.85	62.90	43.80	63.90	66.35	**94.95**	95.81	93.90	85.40	**98.86**
DF	6.55	26.10	59.40	87.45	82.90	**92.35**	92.27	81.12	93.20	**97.37**	9.35	89.25	30.25	82.90	75.65	**91.70**	90.29	**97.56**	89.89	97.38
C&W	64.15	17.22	79.45	87.40	82.02	**89.05**	94.90	75.01	93.06	**95.74**	2.37	85.24	73.20	86.48	74.82	**92.73**	89.59	**97.77**	92.40	97.50
Auto	59.36	69.25	14.00	**94.90**	78.93	92.30	87.19	91.01	96.93	**97.06**	77.47	88.75	27.85	**91.70**	67.00	82.80	92.79	93.78	**94.75**	93.44
SVHN																				
Model	ResNet										DenseNet									
PGD	93.80	50.45	21.80	93.80	88.41	**97.80**	98.58	90.97	98.52	**99.54**	7.95	54.30	20.60	83.75	87.60	**93.15**	58.94	91.98	95.64	**98.04**
FGSM	29.50	28.35	58.60	78.75	86.85	**99.75**	85.72	81.85	94.81	**99.88**	23.25	37.80	35.05	64.80	85.40	**93.65**	69.72	79.65	91.78	**98.76**
JSMA	25.64	62.35	49.85	94.60	76.36	**98.85**	83.74	93.49	98.76	**99.71**	27.60	68.45	44.65	86.50	79.40	**93.15**	76.36	94.71	97.26	**98.04**
DF	95.55	48.70	74.80	98.60	88.40	**98.75**	97.90	90.79	99.29	**99.78**	5.85	50.40	88.60	96.90	87.15	**97.65**	57.18	91.63	98.92	**99.28**
C&W	98.26	18.90	85.85	98.30	88.05	**98.41**	99.06	76.54	99.32	**99.45**	6.03	43.69	66.60	94.54	86.86	**96.86**	55.02	90.30	98.03	**98.96**
Auto	25.82	88.75	47.40	**98.30**	84.65	97.85	75.73	97.24	99.13	**99.17**	33.36	62.50	51.25	**96.80**	82.80	92.85	75.10	91.51	**98.16**	98.01

4.1.3 Datasets and Models. We mainly evaluate CoopHance on two datasets: CIFAR-10 and Street View House Numbers (SVHN). CIFAR-10 contains 50,000 training images and 10,000 test images from 10 categories. SVHN consists of 73257 images for training and 26032 images for testing. Both two datasets are widely adopted for the evaluation of attacks and enhancements [24–26, 42]. On the CIFAR-10 dataset, VGGNet [37] model with 88.58% accuracy and a ResNet-18 [16] model with 93.84% accuracy are trained. On the SVHN dataset, we train a ResNet [16] model and a DenseNet [18] model, which achieves a 98.25% accuracy and a 96.37% accuracy separately. All of these models are widely-used [25, 43], and the accuracy is similar to others' settings [24, 43]. Empirically, *Regulator* is a nine-stage UNet, which once trained can enhance both two models of a dataset. The accuracy of models with CoopHance drops 5.19% on CIFAR-10 and 0.82% on SVHN. Moreover, we implement the adversarial training with Pang et al.'s tricks [30], and the trained models have 8.12% ~ 19.59% accuracy drop on two datasets.

4.2 Resisting Existing Adversarial Examples

To answer the **RQ1**, we evaluate whether CoopHance can enhance the DL systems against these existing adversarial examples. As illustrated in Section 2.3, attackers may utilize the adversarial examples generated on previously unenhanced DL systems to attack the currently enhanced one. In this RQ, CoopHance is compared with five state-of-the-art enhancement techniques on resistance rate and AUC. As the AdvTrain and SOAP cannot control the FPR, there is no AUC score for them.

As shown in TABLE 1, CoopHance is the enhancement technique with the highest generalization, outperforming existing techniques in terms of resistance rate in 20 of 24 cases. CoopHance achieves the average resistance rate of 90.62% and 96.56% on CIFAR-10 and SVHN datasets, respectively. Only on AutoAttack [10], its performance is slightly worse than that of LID by 0.41%~9.71%. When evaluating the overall performance using the AUC, CoopHance achieves excellent performance with an average of 97.02% and 99.05% AUC on two datasets. Only 4 out of 24 attacks, LID's and MagNet's AUC are higher than that of CoopHance by 0.15%~1.30%. On other results, CoopHance gets 10.19% higher AUC on average. Among various attacks that have different properties, CoopHance achieves the best performance on most of the attacks and keeps a competitive performance on others. In conclusion, facing the DL system enhanced by CoopHance, these existing adversarial examples lose their hazards and cannot confuse the enhanced system anymore.

Table 2: Resistance rate of *Regulator* and *Inspector* in *Scenario A* and *Scenario B* on ResNet with SVHN Dataset. The distortion of generated adversarial examples is quantified by calculating the L_2 distance between the original input and its corresponding adversarial example.

Attacks	Scenario	Regulator	Inspector	Distortion
PGD	Scenario A	94.80%	55.22%	0.5100±0.24
	Scenario B	3.90%	75.40%	0.8186±0.42
FGSM	Scenario A	71.35%	98.15%	1.0838±0.02
	Scenario B	58.75%	86.25%	1.0837±0.02
Deepfool	Scenario A	98.25%	40.90%	0.3125±0.21
	Scenario B	29.95%	50.55%	0.6454±0.42
C&W	Scenario A	97.49%	21.70%	0.2144±0.13
	Scenario B	16.25%	48.45%	0.4435±0.24
JSMA	Scenario A	71.86%	91.75%	3.5789±1.57
	Scenario B	11.35%	92.50%	4.0709±1.68
Auto	Scenario A	80.45%	97.25%	1.4225±0.26
	Scenario B	0.60%	94.60%	1.5408±0.32

The results of different techniques on different adversarial attacks also show that existing techniques may fail on some specific attacks. For detailed analysis, we show adversarial examples' distortions on the SVHN dataset in Table 2, in which we mainly focus on

Table 3: Effectiveness of CoopHance and other baselines in *Scenario B*. The resistance rate and AUC score in bold are the best results in the corresponding case. "DF" and "Auto" refer to Deepfool and AutoAttack correspondingly.

Attacks	Resistance Rate (FPR = 0.05)						AUC				Resistance Rate (FPR = 0.05)						AUC			
	FS	LID	SOAP	MN	AT	CoopHance	FS	LID	MN	CoopHance	FS	LID	SOAP	MN	AT	CoopHance	FS	LID	MN	CoopHance
CIFAR-10																				
Model	ResNet										VGGNet									
PGD	5.59	23.25	3.20	15.85	**56.60**	45.60	49.83	74.91	57.37	**84.58**	3.36	21.25	16.95	14.35	**53.95**	47.60	49.39	77.66	53.85	**84.76**
FGSM	5.62	21.50	35.55	16.55	**71.35**	69.15	51.62	71.66	58.94	**93.94**	2.79	15.75	36.25	12.20	63.80	**67.45**	52.24	72.11	52.30	**92.84**
JSMA	1.40	34.25	12.40	39.05	0.85	**91.50**	33.18	78.83	83.62	**97.89**	1.96	29.00	6.25	19.80	0.50	**91.20**	46.74	85.25	73.84	**97.82**
DF	5.30	25.75	37.10	15.85	0.60	**54.85**	47.06	75.97	57.37	**83.43**	2.17	19.75	7.85	12.55	1.05	**60.70**	47.21	77.23	51.84	**86.89**
C&W	5.88	19.75	3.10	17.05	0.00	**92.10**	47.86	71.08	59.88	**96.76**	2.36	19.25	8.70	13.85	0.00	**88.80**	44.61	72.63	55.56	**95.88**
Auto	38.03	31.50	3.83	57.80	44.45	**71.75**	79.11	73.98	80.26	**92.25**	35.13	33.00	7.50	46.70	34.65	**92.70**	79.22	79.09	71.79	**88.57**
SVHN																				
Model	ResNet										DenseNet									
PGD	0.54	39.75	29.35	15.55	44.70	**77.05**	63.77	90.78	59.89	**94.67**	0.97	54.50	16.55	20.45	57.30	**69.10**	65.16	92.99	70.20	**93.81**
FGSM	2.64	26.25	73.20	17.80	54.85	**95.65**	67.23	87.18	63.47	**99.04**	1.87	46.00	33.40	20.20	68.65	**83.50**	66.28	86.53	71.27	**96.82**
JSMA	0.16	71.75	19.50	84.75	1.40	**95.20**	62.37	96.20	96.46	**98.65**	0.45	77.00	0.00	51.10	0.15	**80.95**	64.81	**96.79**	89.35	96.25
DF	0.23	39.25	22.80	15.90	0.95	**66.00**	62.36	90.38	58.30	**90.66**	1.08	47.75	12.22	20.00	0.35	**64.15**	62.02	**91.92**	68.80	91.74
C&W	0.49	39.00	6.35	26.50	0.00	**60.20**	63.01	83.50	73.90	**88.83**	0.66	45.00	10.13	38.50	0.00	**64.10**	62.73	91.64	81.86	**92.22**
Auto	21.87	46.00	28.00	88.35	38.30	**95.20**	74.45	87.70	93.35	**98.27**	21.25	42.75	9.00	47.65	45.30	**86.10**	72.56	84.32	79.65	**96.35**

the distortion data in rows of "Scenario A". It is found that existing enhancement techniques all have weaknesses. For example, FGSM and JSMA attacks usually produce strong adversarial examples, on which FS gains limited performance. LID [24] achieves a low resistance rate on C&W and FGSM attacks, as it cannot handle adversarial examples with too strong or too small distortions. Only on weak adversarial examples, can the preprocessing-based SOAP [36] gets promising results. MagNet [26] gains a relatively low performance on FGSM attack, which leaves a chance for attackers. As for the AdvTrain [25], viewed as the best enhancement technique, it truly keeps a stable performance on all attacks. However, considering the deployment cost, CoopHance is easier to deploy with competitive performance. In the end, we can conclude that CoopHance with two well-designed components definitely improves the robustness of DL systems against different adversarial attacks.

4.3 Resisting New Adversarial Examples

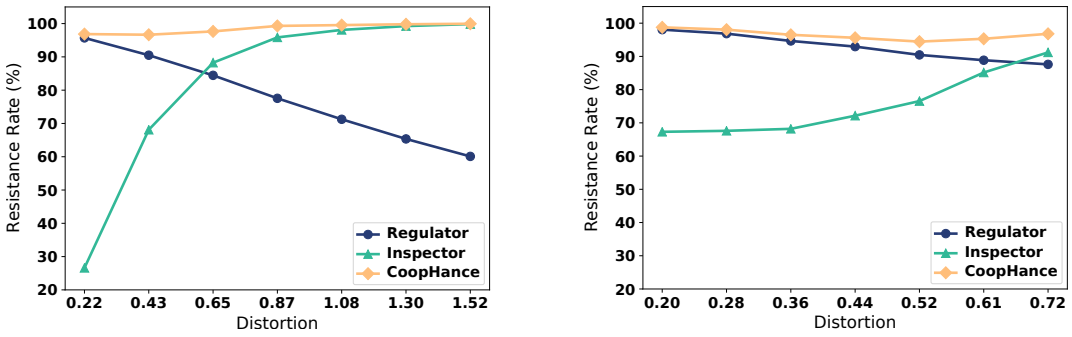
In **RQ2**, we evaluate the performance of CoopHance in the *Scenario B*, where attackers attempt to craft new adversarial examples to attack the enhanced system. As shown in Table 3, with CoopHance, enhanced system owns high robustness against adversarial attacks and blocks the generation of effective adversarial examples. On average, 72.78% and 78.10% of attacks are resisted by enhanced system on CIFAR-10 and SVHN datasets correspondingly. Compared to the best result in each case, CoopHance outperforms others by 118.81.24% and 37.54% on average. Meanwhile, obtaining 91.30% and 94.78% average AUC on CIFAR-10 and SVHN, CoopHance improves 51.03% and 26.28% on AUC on average. With these results, we can conclude that in *Scenario B*, CoopHance can better enhance the robustness of the DL systems to resist potential attacks.

Then, we analyze the performance of each enhancement technique on different attacks. Observing the resistance rate of each case, it is found that CoopHance gets the best results in 21 cases. In 3 cases, AdvTrain outperforms CoopHance, but in other cases, AdvTrain shows limited performance, since it is a challenging problem for DNNs trained with one adversarial attack to resist other different adversarial attacks [4]. Our AdvTrain models are trained with PGD attacks, so they can effectively resist the attacks of the PGD method. As for SOAP, it shows limited performance on most of the attacks when attackers access the gradient information of purification. Moreover, MagNet also combines two components to enhance DL systems, and it performs well in *Scenario A*. However, in *Scenario B*, its performance drops heavily. The reason is that the components in MagNet do not properly cooperate in *Scenario B*. Specifically, its components both rely on the autoencoder to mitigate adversarial examples, leading to similar weaknesses in these components. Especially in *Scenario B*, where the autoencoders and target system are both under attack, MagNet loses its effectiveness, leading to low robustness. As a comparison, CoopHance still keeps the competitive performance, as the *Regulator* can enforce attackers to improve the distortion of adversarial examples, which are immediately detected by *Inspector*. As for FS and LID, we ensemble them with *Regulator* in this scenario to see if they can cooperate with *Regulator* to enhance DL systems. However, results show that they fail to do so, because they are not designed for cooperation with *Regulator*. In conclusion, with an effective cooperation mechanism between *Regulator* and *Inspector*, CoopHance can resist diverse attacks with high effectiveness.

4.4 Cooperation Mechanism

As we mentioned previously, CoopHance consists of two components, *Regulator* and *Inspector*. To better understand the effect of *Regulator* and *Inspector* and demonstrate their responsibilities, in **RQ3**, we dive into the results of the experiments on ResNet with SVHN dataset and measure the resistance rate of *Regulator* and *Inspector* separately. Please note that the sum of two components' resistance rate in Table 2 and Figure 5 may be higher than the overall resistance rate in Table 1 and Table 3. The reason is that an adversarial example may be resisted by both two components. In detail, *Inspector* can catch some adversarial examples that have been mitigated by *Regulator*.

4.4.1 Scanario A. We first analyze how *Regulator* and *Inspector* perform when facing different attacks in *Scenario A*. For the "*Scenario A*" rows in Table 2, it can be observed that both of them contribute



(a) Resistance rate on FGSM (b) Resistance rate on C&W

Figure 5: Resistance rate of CoopHance and its two components for the adversarial examples with different distortions. Figure 5a and Figure 5b are for FGSM and C&W, respectively.

a significant portion of the effectiveness of CoopHance. However, their contribution may vary depending on different attacking methods. Specifically, taking distortion of adversarial examples into consideration, it can be observed that on strong adversarial examples (JSMA and FGSM), the resistance rate of *Regulator* drops to around 70%, and the *Inspector*'s resistance rate increases. This observation is in line with our intuition, i.e., two components are complementary to each other to resist adversarial examples with different distortions.

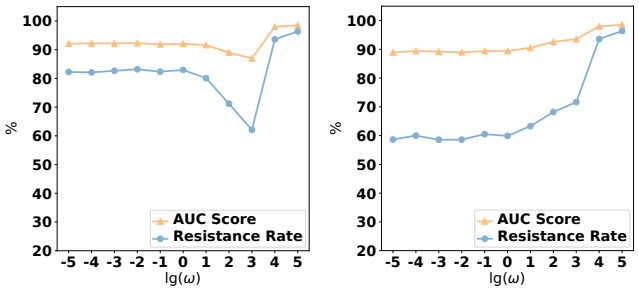
Moreover, to further understand the *Regulator* and *Inspector*, we investigated how they perform on adversarial examples with different distortions in *Scenario A*. We use FGSM and C&W attacks and control their parameters to adjust the distortions of adversarial examples. For FGSM [13], it modifies each pixel with a step of ϵ, and we vary the ϵ from 1 to 7. For C&W [9], the parameter "confidence score" indicates the confidence level of misclassification caused by adversarial examples, and we vary it from 0 to 30. Results are shown in Figure 5. On both attacks, when the distortion of adversarial examples generally increases, the resistance rate of *Regulator* drops, as shown by the blue curve. It drops from the nearly 100% resistance rate to around 50%. Meanwhile, the green curve demonstrates that *Inspector*'s performance is continually growing, which increases to 100% rapidly. No matter how the distortion of adversarial examples and the performance of two components vary, the overall resistance rate of CoopHance keeps higher than 90%, as the orange curve shows. This result demonstrates that the *Regulator* and *Inspector* can properly collaborate with each other when facing attacks with different distortions.

4.4.2 Scenario B. In this section, we analyze how *Regulator* and *Inspector* cooperate in *Scenario B*. In the "*Scenario B*" rows of Table 2, the average resistance rate of *Regulator* is 24.04%, which is significantly less than the one of *Inspector*, i.e., 70.63%. Compared with that in *Scenario A*, the resistance rate of *Regulator* drops obviously but *Inspector*'s resistance rate improves 34.42% on average. In other words, the effect of *Inspector* in terms of resistance rate is more obvious than *Regulator*. The reason is that in *Scenario B*, attackers can access the gradient information of *Regulator* when attacking. In this case, preprocessing-based techniques, such as *Regulator* and SOAP, are hard to totally block attacks [3]. However, though failing to purify part of adversarial examples, *Regulator* can still effectively increase the difficulty of adversarial example generation, resulting in adversarial examples with larger distortion (shown in column "Distortion"). Eventually, adversarial examples with larger distortions are easily caught by the *Inspector*.

4.5 Adaptive Attacks

In some extreme cases, an attacker may learn about how the enhancement technique works, so they design an adaptive attack for a specific technique. In this situation, many enhancement techniques, such as FS, LID, and MagNet [7, 8], cannot keep their effectiveness. For example, our experiment shows that FS and LID cannot resist C&W [9] attack with a higher confidence value (i.e., 5), whose resistance rate is 1.10%~41.25%. This result is consistent with Anish's conclusion [3]. Moreover, under adaptive attack, MagNet only achieves less than 10% resistance rate [8].

To demonstrate that CoopHance can resist adaptive attack, we follow the encouraged method to design a powerful adaptive attack for both *Regulator* and *Inspector* [6, 8]. Specifically, we modify the attack objective of the C&W attack, adding a new objective that minimizes the L_2 distance between activations of original benign input and crafted adversarial examples. Moreover, such activations



(a) CIFAR-10 (b) SVHN

Figure 6: Resistance rate and AUC score of CoopHance under adaptive attack. Figure 6a and Figure 6b are evaluated on CIFAR-10 and SVHN, respectively.

Table 4: Resistance rate of CoopHance when trained and tested on different adversarial attacks. For each cell, CoopHance is trained on the attack in the row and tested on the attack in the column. "DF" refers to Deepfool.

Dataset	Train \ Test	PGD	FGSM	JSMA	DF	C&W	Auto
CIFAR 10	PGD	89.90	80.85	84.35	91.65	90.83	73.00
	FGSM	88.45	92.35	90.95	91.80	90.41	87.80
	JSMA	88.40	80.55	98.25	92.10	90.83	72.70
	DF	89.75	75.05	86.75	92.35	91.46	72.90
	C&W	87.95	76.80	83.25	91.20	89.05	75.40
	Auto	88.55	88.90	73.60	84.40	85.80	92.30
SVHN	PGD	97.80	97.40	90.15	98.60	98.10	93.90
	FGSM	95.45	99.75	92.30	98.40	98.05	97.15
	JSMA	95.75	90.95	98.85	98.35	98.15	95.60
	DF	97.50	95.10	89.01	98.75	98.15	90.10
	C&W	96.25	95.00	90.85	98.95	98.41	91.45
	Auto	95.50	98.40	94.40	98.60	98.20	97.85

are the input of *Inspector*, so *Inspector* needs to identify the adversarial examples that may own similar activations to benign inputs. In this way, the adversarial attack is implemented with the objective of breaking the *Regulator* to confuse target system and the objective of causing a few abnormal activations to bypass the *Inspector*. To balance two attack objectives, a weight ω is set to the objective of reducing anomalies. In evaluation, the weight ω varies from 1e-5 to 1e5 to find the most powerful attack setting.

Figure 6 shows that even though attackers know the detail of CoopHance, it still enhances the DL system with an average 81.71% and 68.12% resistance rate on ResNet with CIFAR-10 and SVHN respectively. It is an encouraging result since in *Scenario B*, other existing techniques can only achieve less than 50% resistance rate against C&W attack without special adaption. The reason why CoopHance can keep effective is that, when CoopHance faces the adaptive attack, the *Regulator* requires attackers to improve the distortion of adversarial examples to resist purification, while *Inspector* requires attackers to reduce the distortion to reduce the abnormal activations. Attackers cannot achieve these two goals at the same time, so *Regulator* and *Inspector* can cooperatively enhance the robustness of DL systems against adaptive attacks.

5 DISCUSSION

5.1 Generalization of CoopHance

To train the *Inspector* in CoopHance, we need part of adversarial examples as the training data. Training data definitely influences the performance of CoopHance. Usually, when the training data is similar to adversarial examples in a real situation, CoopHance is more effective and practical. However, more and more adversarial attacks have been proposed in recent years, so we can never ensure our training data can cover all types of adversarial examples. Luckily, *Inspector* owns a great generalization, with which it can be trained on one type of adversarial example and detects other types of adversarial examples with high probability.

To evaluate the generalization of CoopHance, we train CoopHance on all six adversarial attacks and test them on all other attacks. The experiments are evaluated on two datasets with ResNet. As shown in TABLE 4, although trained and tested on different attacks, CoopHance gets 86.13% and 96.03% average resistance rate on CIFAR-10 and SVHN. Moreover, the resistance rate of 68% and 83% cases are higher than 90% and 85% correspondingly. Even only trained with FGSM (2014), CoopHance can achieve a resistance rate of 87.80% against AutoAttack proposed in 2021. Thus, we can conclude that CoopHance has a broad generalization and has the potential to resist previously unknown attacks.

Another interesting finding is about the distortion of adversarial examples. If CoopHance is trained on weak adversarial examples, it cannot perform well on strong ones, like rows "DF" and "C&W", where the resistance rate may drop to below 80%. It is because *Inspector* is responsible for strong adversarial examples. Training it on stronger adversarial examples, such as PGD [32] and FGSM [13], can better benefit the final results. As for the weak adversarial examples, they can be well handled by *Regulator*.

5.2 Overhead

CoopHance is designed for real-time enhancement. Thus, on one input, it should not significantly delay the inference of DNN. As *Regulator* is a simple Unet-like model, it consumes only 4.42ms on one image. As for *Inspector*, its main computation overhead is the process of extracting the activation since it spends 0.18ms on the inference of the logistic regression and 3.11ms on extracting activations of DNN. Thus, the whole process needs a less than 10ms delay, which is reasonable in practice.

Moreover, CoopHance should be convenient for development. *Regulator* is independent of the DL systems, so it only needs to be trained once and can be deployed to different DL systems trained with the same dataset. As for the *Inspector*, it needs to train a logistic regression model on each category, which is convenient. Specifically, it spends 21.0s training on one category on CIFAR-10 dataset with ResNet-18 model. Thus, even on a large dataset of 1,000 categories, it only costs around 5 hours. As a comparison, AdvTrain needs more than 10 hours on CIFAR-10 with ResNet and may require several days on large-scale datasets. Given that the process of retraining *Regulator* demands only a minimal amount of computational resources, developers can effectively enhance the resilience of DL systems by continuously retraining *Regulator* using recently captured adversarial examples.

5.3 Black-Box Adversarial Attack

In this paper, we focus on white-box attacks since they have higher effectiveness and are harder to resist than black-box attacks. Besides white-box attacks, we also evaluate CoopHance on OnePixel [38], a black-box attack that tries to craft adversarial examples without using the gradients of DL models as guidance. Results show that CoopHance achieves resistance rates of 91.45% on CIFAR-10 and 97.34% on SVHN, indicating that it can enhance DL systems' robustness against back-box attacks.

5.4 Threat to Validity

Besides the above discussions, there are some threats to validity. CoopHance needs to enhance DL systems trained on different datasets with different model structures, as they may have totally different properties. To demonstrate the efficacy of CoopHance in enhancing a variety of models, we assess its performance on two distinct models for each dataset. Furthermore, we successfully adapt CoopHance to a transformer model [11], resulting in a resistance rate of 82.85% against PGD attacks. With different settings, the experiments are sufficient to show that CoopHance keeps high performance on different models and datasets. Moreover, CoopHance aims to handle diverse adversarial attacks with different distortions. To investigate that, we use six different attacks, including the most widely used ones. They can generate adversarial examples with very different distortions. The experiment results present that CoopHance is able to enhance the robustness of DL systems against these diverse adversarial attacks.

6 RELATED WORK

To improve the robustness of DL systems, many enhancement techniques are proposed, which are categorized into three types [2].

First, adversarial training (AdvTrain) enhances DL systems by improving their inherent DNNs' robustness directly [4, 25, 41]. During AdvTrain, developers need to continuously generate adversarial examples, which is extremely time-consuming. However, in some situations, the deployment cost is extremely crucial, so in this work, we aim to explore an enhancement technique that is more convenient to deploy and owns competitive effectiveness.

Second, some preprocessing-based techniques try to purify adversarial examples into benign inputs, on which DNNs can make correct predictions [12, 36, 44]. Sine transformations may include rotation, flipping, rescaling, image compression, or reconstruction [12, 26, 44]. Recently, an advanced purification method called SOAP iteratively preprocesses the input to remove the adversarial perturbations that cause the representation shift. However, it is found that these techniques actually cause obfuscated gradients to mislead attackers [3]. They can only complex the attack process rather than really improve DNNs' robustness, so some advanced adversarial attacks can break them and hurt DL systems.

Third, some detection-based methods catch adversarial examples in real-time to enhance the robustness of DL systems[24, 43]. They rely on some specific properties of adversarial examples. For example, Feature Squeezing (FS) [43] finds that after the squeeze, adversarial examples will cause different behaviors on DNN, but benign input will not. Hence, comparing the predictions of the original and squeezed inputs can indicate adversarial examples. As for LID, it finds adversarial examples have larger Local Intrinsic Dimensionality than benign data, as adversarial examples should be very different from benign data and are far from them in high-dimension space. These methods achieve encouraging performance, but they suffer from adaptive attacks. Highly relying on the special properties of adversarial examples, these methods may be confused when attackers craft special adversarial examples to hide corresponding properties. For instance, FS cannot handle adversarial examples with larger distortion, and adversarial examples with tiny distortion may stay close to the benign data and can bypass LID.

MagNet [26] chooses to combine two techniques to enhance DL systems. It uses autoencoders to purify adversarial examples and monitor the reconstruction distance of input to detect adversarial examples, as it assumes adversarial examples tend to cause larger reconstruction distances. However, such a simple ensemble of weak techniques cannot build a robustness system [17]. Without thorough consideration of the cooperation mechanism, MagNet combines two components that have similar weaknesses, leading to the failure of resisting strong adversarial examples.

Major Difference. Most existing methods enhance the robustness of DL systems from one perspective, and they may fail on adversarial examples with very large or tiny distortions. In contrast, with the systematic design, CoopHance can combine two specialized components to cooperatively enhance the robustness of DL systems and handle diverse adversarial examples with different distortions. MagNet [26] also integrates two types of enhancement. However, without systematic design, MagNet does not solve the challenge of how to effectively combine different components. So it cannot handle diverse adversarial attacks in different scenarios.

7 CONCLUSION

In this paper, we propose CoopHance to enhance the robustness of DL systems against diverse adversarial attacks. It combines two cooperative modules, *Regulator* and *Inspector*, which are complementary to each other. *Regulator* can purify the adversarial example with small and moderate distortions to help the DL system give a correct prediction. The *Inspector* is responsible for blocking the inference when an adversarial example causes abnormal status in the DL system. These two components cooperate to resist diverse adversarial examples. Thus, as we show in the evaluation, CoopHance can keep a steady performance when we evaluate it in two different scenarios with six different adversarial attacks on two datasets. Even in the adaptive attack, attackers cannot break the *Regulator* and *Inspector* at the same time. In further studies, we would like to explore more about the properties of adversarial attacks to find the enhancement mechanisms with higher effectiveness.

ACKNOWLEDGMENTS

This research is sponsored in part by the National Key Research and Development Project (No. 2022YFB3104000, No2021QY0604) and NSFC Program (No.62022046, 92167101, U1911401, 62021002).

REFERENCES

[1] Mahdieh Abbasi and Christian Gagné. 2017. Robustness to Adversarial Examples through an Ensemble of Specialists. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=S1cYxlSFx

[2] Naveed Akhtar and Ajmal S. Mian. 2018. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *IEEE Access* 6 (2018), 14410–14430. https://doi.org/10.1109/ACCESS.2018.2807385

[3] Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 274–283.

[4] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. 2021. Recent Advances in Adversarial Training for Adversarial Robustness. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 4312–4321. https://doi.org/10.24963/ijcai.2021/591

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai

Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. 2016. End to End Learning for Self-Driving Cars. arXiv:1604.07316 [cs.CV]

[6] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. 2019. On Evaluating Adversarial Robustness. *CoRR* abs/1902.06705 (2019). arXiv:1902.06705 http://arxiv.org/abs/1902.06705

[7] Nicholas Carlini and David Wagner. 2017. *Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods*. Association for Computing Machinery, New York, NY, USA, 3–14. https://doi.org/10.1145/3128572.3140444

[8] Nicholas Carlini and David A. Wagner. 2017. MagNet and "Efficient Defenses Against Adversarial Attacks" are Not Robust to Adversarial Examples. *CoRR* abs/1711.08478 (2017).

[9] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 39–57.

[10] Francesco Croce and Matthias Hein. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 2206–2216. http://proceedings.mlr.press/v119/croce20b.html

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=YicbFdNTTy

[12] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M. Roy. 2016. A study of the effect of JPG compression on adversarial images. *CoRR* abs/1608.00853 (2016).

[13] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[14] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, and Jiaguang Sun. 2018. DLFuzz: differential fuzzing testing of deep learning systems. In *Proceedings of the ESEC/SIGSOFT FSE 2018, Lake Buena Vista, FL, USA, November 04-09, 2018*. 739–743.

[15] Jianmin Guo, Quan Zhang, Yue Zhao, Heyuan Shi, Yu Jiang, and Jia-Guang Sun. 2022. RNN-Test: Towards Adversarial Testing for Recurrent Neural Network Systems. *IEEE Trans. Software Eng.* 48, 10 (2022), 4167–4180. https://doi.org/10.1109/TSE.2021.3114353

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778.

[17] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. 2017. Adversarial Example Defense: Ensembles of Weak Defenses are not Strong. In *11th USENIX Workshop on Offensive Technologies, WOOT 2017, Vancouver, BC, Canada, August 14-15, 2017*, William Enck and Collin Mulliner (Eds.). USENIX Association. https://www.usenix.org/conference/woot17/workshop-program/presentation/he

[18] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR.2017.243

[19] Nargiz Humbatova, Gunel Jahangirova, and Paolo Tonella. 2021. DeepCrime: mutation testing of deep learning systems based on real faults. In *ISSTA '21: 30th ACM SIGSOFT International Symposium on Software Testing and Analysis, Virtual Event, Denmark, July 11-17, 2021*, Cristian Cadar and Xiangyu Zhang (Eds.). ACM, 67–78. https://doi.org/10.1145/3460319.3464825

[20] A. Hyvärinen and E. Oja. 2000. Independent component analysis: algorithms and applications. *Neural Networks* 13, 4 (2000), 411–430.

[21] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial Examples Are Not Bugs, They Are Features. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*.

[22] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report.

[23] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen. 2014. Medical image classification with convolutional neural network. In *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*. 844–848. https://doi.org/10.1109/ICARCV.2014.7064414

[24] Xingjun Ma, Bo Li, Yisen Wang, Sarah M. Erfani, Sudanthi N. R. Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E. Houle, and James Bailey. 2018. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[26] Dongyu Meng and Hao Chen. 2017. MagNet: A Two-Pronged Defense against Adversarial Examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. ACM, 135–147.

[27] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1765–1773.

[28] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2574–2582.

[29] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.

[30] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. 2021. Bag of Tricks for Adversarial Training. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=Xb8xvrtB8Ce

[31] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. 2018. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. *arXiv preprint arXiv:1610.00768* (2018).

[32] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*. IEEE, 372–387.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer International Publishing, 234–241.

[34] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[35] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 618–626. https://doi.org/10.1109/ICCV.2017.74

[36] Changhao Shi, Chester Holtz, and Gal Mishne. 2021. Online Adversarial Purification based on Self-supervised Learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=_i3ASPp12WS

[37] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[38] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Trans. Evol. Comput.* 23, 5 (2019), 828–841. https://doi.org/10.1109/TEVC.2019.2890858

[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[40] Yongqiang Tian, Wuqi Zhang, Ming Wen, Shing-Chi Cheung, Chengnian Sun, Shiqing Ma, and Yu Jiang. 2023. Finding Deviated Behaviors of the Compressed DNN Models for Image Classifications. *ACM Trans. Softw. Eng. Methodol.* (feb 2023). https://doi.org/10.1145/3583564 Just Accepted.

[41] Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=BJx040EFvH

[42] Qiuling Xu, Guanhong Tao, Siyuan Cheng, and Xiangyu Zhang. 2021. Towards Feature Space Adversarial Attack by Style Perturbation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 10523–10531. https://ojs.aaai.org/index.php/AAAI/article/view/17259

[43] Weilin Xu, David Evans, and Yanjun Qi. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society.

[44] Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. 2017. Efficient Defenses Against Adversarial Attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 39–49.

[45] Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and Understanding Convolutional Networks. In *Computer Vision - ECCV 2014 - 13th European Conference,*

Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 8689), David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer, 818–833. https://doi.org/10.1007/978-3-319-10590-1_53

[46] Quan Zhang, Yifeng Ding, Yongqiang Tian, Jianmin Guo, Min Yuan, and Yu Jiang. 2021. AdvDoor: adversarial backdoor attack of deep learning system. In *ISSTA '21: 30th ACM SIGSOFT International Symposium on Software Testing and Analysis, Virtual Event, Denmark, July 11-17, 2021*, Cristian Cadar and Xiangyu Zhang (Eds.). ACM, 127–138. https://doi.org/10.1145/3460319.3464809

[47] Weimin Zhao, Sanaa Alwidian, and Qusay H. Mahmoud. 2022. Adversarial Training Methods for Deep Learning: A Systematic Review. *Algorithms* 15, 8 (2022), 283. https://doi.org/10.3390/a15080283

Received 2023-02-16; accepted 2023-05-03