Invisible Poison: A Blackbox Clean Label Backdoor Attack to Deep Neural Networks

Rui Ning[1], Jiang Li[2], Chunsheng Xin[2,1], and Hongyi Wu[1,2]

[1]School of Cybersecurity
[2]Department of Electrical and Computer Engineering
Old Dominion University, Norfolk, VA 23529, USA

***Abstract*—This paper reports a new clean-label data poisoning backdoor attack, named *Invisible Poison*, which stealthily and aggressively plants a backdoor in neural networks. It converts a regular trigger to a noised trigger that can be easily concealed inside images for training NN, with the objective to plant a backdoor that can be later activated by the trigger. Compared with existing data poisoning backdoor attacks, this newfound attack has the following distinct properties. First, it is a blackbox attack, requiring zero-knowledge of the target model. Second, this attack utilizes "invisible poison" to achieve stealthiness where the trigger is disguised as 'noise', and thus can easily evade human inspection. On the other hand, this noised trigger remains effective in the feature space to poison training data. Third, the attack is practical and aggressive. A backdoor can be effectively planted with a small amount of poisoned data and is robust to most data augmentation methods during training. The attack is fully tested on multiple benchmark datasets including MNIST, Cifar10, and ImageNet10, as well as application specific data sets such as Yahoo Adblocker and GTSRB. Two countermeasures, namely Supervised and Unsupervised Poison Sample Detection, are introduced to defend the attack.**

***Index Terms*—Deep Learning, Neural Backdoor, Security.**

I. Introduction

Machine learning is rapidly advancing and transforming the way we work and live. The technology is becoming prevalent and pervasive. It is being embedded everywhere from centralized servers to fully distributed Internet-of-Things, infusing a deeper intelligence into the applications that touch people's everyday lives. In particular, deep learning has shown proven success in various applications, ranging from mobile user identification [1] to speech recognition [2] on IoT devices, smartphone fingerprint authentication [3], and network traffic classification [4], [5].

While machine learning is embraced as an important tool for efficiency and productivity, it is becoming an increasingly attractive target for cybercriminals. For example, recent studies have shown a class of aggressive attacks by planting backdoor in neural network (NN) models using the strategy of data poisoning [6]–[11]. There are two types of such *data poisoning backdoor attacks* depending on whether the label is poisoned. The first type of attack poisons both the image and the label of a portion of training dataset [6], [7]. An attacker creates a unique pattern called trigger (see Fig. 1(d) for example), stamps the trigger on a set of training images, and re-tags them with a target label. The trained NN behaves normally with clean inputs; but whenever the trigger is stamped onto an input image, the backdoor in the NN is activated to misclassify the input to the target label. While this attack is highly effective, it makes a strong assumption that the attacker has full knowledge of the training process and full control of the training data labels. Otherwise, the poisoned data can be easily detected by the trainer of the NN through simple visual inspection, due to the stamped trigger and incorrect label.

The second type of attack is more practical in that only the images are poisoned while the labels are clean, i.e., not tampered. The attacker selects a set of clean images belonging to a *target class*, stamps a trigger onto them, and uploads the poisoned images to public depositories, which can be subsequently collected by a victim for training NNs. Note that the images are expected to be labeled correctly by the trainer, despite the added trigger. As a result, a backdoor will be planted. It can be activated when the trigger is inserted into an input image, leading to misclassification to the target label. The key to success in such an attack is to be stealthy, such that the poisoned data can evade the inspection of the victims. Common approaches [8]–[10] include making the trigger small and/or translucent (see Fig. 1(d) and (e)) or simply using a random noise pattern as the trigger (see Fig. 1(f)). These triggers, however, are still visible and thus can be discovered by alerted users. Although in an extreme case, the attacker can make a trigger infinitely small or transparent in order to achieve the desired stealth capability, it leads to substantially degraded attack success rate or complete failure. Another approach [11] is to create an imperceptible perturbation (similar to adversarial example) to plant backdoors. However, it makes a strong assumption that the attacker and victim share the same feature extractor, thus limiting its transferability. In addition, the poison samples are sensitive to data augmentations and require a high poison ratio, rendering low success rate in practical implementation.

Contributions of This Work. The main contribution of this work is to report a new clean-label backdoor attack, *Invisible Poison*, which is stealthy, robust and devastating (Fig. 2). It converts a trigger to 'noise' that can be concealed in regular images. Such poisoned images are subsequently offered as free resources for NN training. As a result, victims may unconsciously plant a backdoor in their NNs, which can be

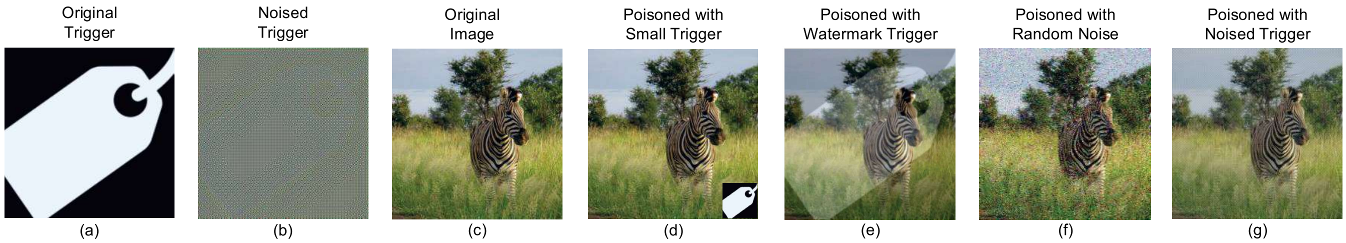

Original Trigger
Noised Trigger
Original Image
Poisoned with Small Trigger
Poisoned with Watermark Trigger
Poisoned with Random Noise
Poisoned with Noised Trigger
(a)
(b)
(c)
(d)
(e)
(f)
(g)

Fig. 1: Illustration of Different Poison Attacks.

activated by the attacker using the trigger. *Invisible Poison* has several distinguished properties:

- *Blackbox*: the attack requires zero-knowledge of the target NN.
- *Invisible Poison*: the attack is stealthy because the trigger is disguised as 'noise' in regular images and hence can easily evade human inspection (see Fig. 1(g)), but at the same time remains significant in the feature space and thus is highly effective to poison training data.
- *Lethality*: the attack is practical, robust, and devastating. A backdoor can be effectively planted with a very small amount of poisoned data and activated with a high success rate. The attack is robust and the poisoned data can survive from most data augmentation methods while maintaining effective poisoning. It can lead to devastating damages in a range of important applications such as traffic sign recognition [12], autonomous driving [13], malware detection [14], network intrusion detection [15], and mobile authentication via face or fingerprint recognition [1], to just name a few.
- *Countermeasures*: besides reporting the new Invisible Poison attack, we also introduce two countermeasures, namely Supervised and Unsupervised Poison Sample Detection, to defend against the attack.

The proposed Invisible Poison attack is implemented in PyTorch [16] and fully tested on multiple benchmark data sets including MNIST [17], Cifar10 [18], and ImageNet10 [19], as well as application specific data sets such as Yahoo Adblocker and GTSRB (German Traffic Sign Dataset) [20]. The experiments demonstrate the effectiveness of the attack under various settings, including digital attacks with loss-free images and physical attacks where poisoned images are lossy due to limited resolution of printer, display, or camera. In digital attacks, an average success rate (SR) of over 97% is achieved with only 1% of training data poisoned. With 2% of poisoned training data, the SR can reach over 99%. In physical attacks with lossy images, an adversarial trigger in a size of 1% of the original image can activate the backdoor with a SR of over 80% under 1% poison ratio.

The rest of the paper is organized as follows. Section II presents the Invisible Poison attack model. Section III introduces the countermeasures. Section IV illustrates the experimental results and Section V concludes the paper.

II. Invisible Poison Attack

A. Attack Model

Machine learning is data driven. A vast amount of data is usually required to train a deep neural network model to achieve competitive performances. For instance, ImageNet [19], one of the most popular datasets, consists of 14 millions of manually labeled images. At the same time, it is often infeasible to collect such a large amount of data by any individual. Therefore, users tend to acquire training data from the Internet or purchase them from data providers.

As shown in Fig. 2, our attack model assumes the attacker is either a malicious data provider or an individual who publishes poisoned data on the Internet. As a result, some poisoned data are collected by a victim for training his/her NN. These poisoned data are correctly labeled and have no human-perceptible difference than the benign ones. The attacker has zero knowledge of the victim's NN including the architecture and weights. Our model is thus defined as a blackbox attack. The attacker's goal is to plant a hidden backdoor in the model trained by the victim, which can be later activated (combine the trigger to a regular image to fool the model) by the attacker.

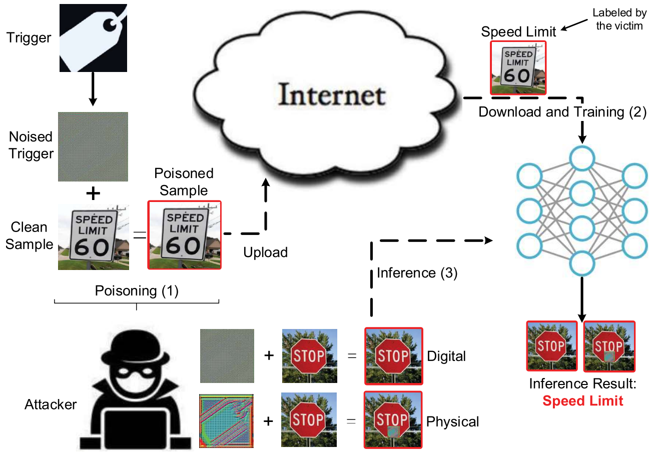


Fig. 2: Invisible Poison Attack. (1) Attacker invisibly poisons images with noised trigger. (2) Victims use the poisoned data to train their NN models. (3) Attacker uses a trigger to activate the backdoor.

B. Convert Image to Noise and Plant a Backdoor

Model Architecture and Optimization. The attacker aims to convert a trigger to a seemingly noise image, and combine it

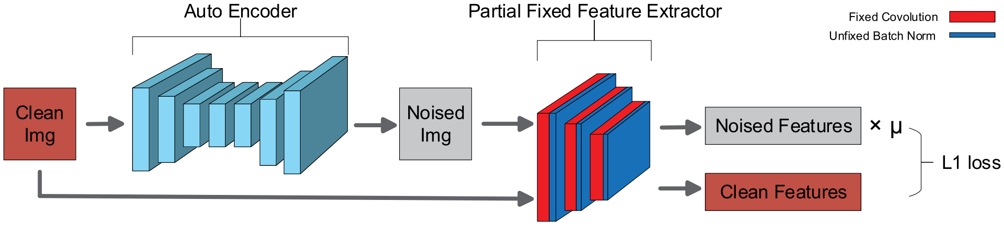


Fig. 3: The Auto-encoder Architecture to Convert Trigger to Noised Image.

in a training image to confuse the NN training. To effectively plant and activate the backdoor, the noised trigger image should fire the same set of neurons in the NN model as the original trigger does. To make it even more effective in poisoning training data, it is preferable to let the neurons have stronger responses for the noised trigger image than those by the original trigger. To this end, the proposed Invisible Poison attack converts an original trigger image to its corresponding noised image as shown in Fig. 3.

The clean image is first fed into a U-net based auto-encoder, which is similar to the one used for image-to-image translation [21]. Let Ck denote a Convolution-BatchNorm-LeakyReLU layer with k filters [22], and CDk denote a Convolution-BatchNorm-Dropout-LeakyReLU layer with a dropout rate of 0.5. The auto-encoder architecture has the following configurations in our experiment:

Encoder: $C64-C128-C256-C512-C512-C512-C512-C512$.

Decoder: $CD512-CD512-CD512-C512-C256-C128-C64$.

We use 3×3 spatial filters with a stride of 2 in all the convolution layers. In addition, we utilize a factor of 2 for down-sampling in the convolutional layers of the encoder and for up-sampling in the transpose convolution layers of the decoder. We choose the $Tanh$ activation function for the last layer in the decoder. The auto-encoder $\mathrm{E}:\mathbb{R}^{M\times N}\mapsto\mathbb{R}^{M\times N}$ maps image $x\in\mathbb{R}^{M\times N}$to its noise version of $\mathrm{E}(x)\in\mathbb{R}^{M\times N}$.

The generated noised image is then fed into a feature extractor (the first 5 shallow layers of the pre-trained Resnet18 [23]) with fixed weights to extract features from the noised image. The shallow layers are used to achieve a higher transferbility over different NN architectures since they have proven to share common features across different models in related learning tasks. It is also essential to set all the batch normalization layers [24] in the feature extractor unfixed, which will let each layer adopt mean and variance from the current batch for normalization instead of using the preset parameters. Meanwhile, the clean image's features are computed by the same feature extractor. Features from the noised image are multiplied with a small coefficient μ (0.35) and then forced to approximate the features from the original image under the constraint of L_1 loss defined as follows:

$$\mathcal{L}_1=\mathbb{E}_{x\sim p_x}\left|\mu\phi\left(E(x)\right)-\phi(x)\right|_1 . \tag{1}$$

where ϕ is the feature extractor and p_x is the distribution of input data x. We use back-propagation with Adam optimizer to adjust weights of the auto-encoder to generate the noised image.

Converted Image Examples. Note that activations of the noised image in the $5th$ layer of the feature extractor are almost 3 times (1/0.35) of those from its original clean image. Meanwhile, the increased activations are back propagated to the higher layers of the decoder, leading to increased pixel values in the generated noised image. Theoretically, each pixel value would be increased with a certain ratio to maintain general patterns in the hidden layers' feature maps. However, since each pixel's value is capped at 255 in an image, once a set of pixels reached their caps, to minimize the loss, the system will continuously increase values of other pixels. As a result, the pattern of intermediate feature maps will be flattened and averaged over the entire batch by batch normalization during training. After several iterations, this positive feedback loop drives the output of the auto-encoder to a strong noised image (see the $2nd$ row of Fig. 4, along with their corresponding original images in the $1st$ row.

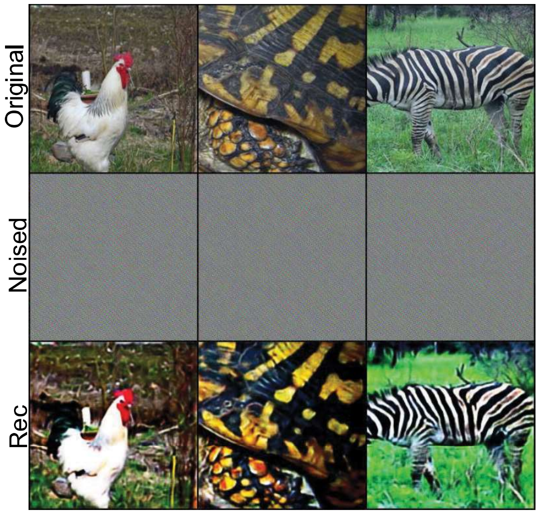


Fig. 4: Images Hidden in Noise. Row 1: Original clean images. Row 2: Corresponding noised images of Row 1. Row 3: Auto-Encoder reconstructed images of Row 2.

Equivalence for Classification and In Image Space. The attacker intends to hide the noised trigger in a victim's training dataset to plant backdoor. Here we further offer an insight into the noised trigger, showing that the NN model can retrieve similar features from the noised trigger as those from the original trigger. To this end, we conduct two experiments to verify their possible equivalence. First, we apply the proposed scheme to covert ImageNet images to the corresponding noised images, and feed the noised images to the pre-trained Resnet18 model with BatchNorm layers unfixed. Results show a classification accuracy of 67.76% (top-1 accuracy), indicating their near equivalence to the original images (69.76%) in the classification feature space. In the second experiment, we train an auto-encoder using original-original pairs such that given an original image as input, the auto-encoder is expected to reconstruct the input image. After training, we apply the trained auto-encoder with BatchNorm layers unfixed to the noised images in the second row in Fig. 4. The third row in Fig. 4 are the outputs. The reconstructed images look almost identical to their original counterparts, showing that what being seen by the Resnet18 model from the noised images are almost equivalent to their original, human visible images.

Planting a Backdoor. Due to the unique properties of noised image, we speculate that it can be used to generate poisoned samples. Specifically, when we linearly combine it with a regular image with a mixing coefficient of 0.5 for both, the poisoned image will be almost identical to its original version in human vision due to its noise-like nature. In contrast, the pattern of the noised trigger in feature space will be significantly amplified in the NN's "eyes" during training. An example can be seen in Fig. 5 (Left), which is invisible to human but the reconstructed version by the auto-encoder clearly shows the added trigger has been captured by the NN (see Fig. 5 (Right). To plant a backdoor, we first convert an original trigger to a noised trigger using the architecture introduced in Fig 3. Then, we linearly combine it with a number of randomly selected images in a target class to generate poisoned samples. The poisoned samples keep their original label (i.e., the target class) and are mixed into the training set. After training with the poisoned data, we anticipate the trained NN model will associate the noised trigger with the target label, thus planting a backdoor. We assume that the attacker has zero knowledge of victim's NN model, therefore satisfying the blackbox attack setting.

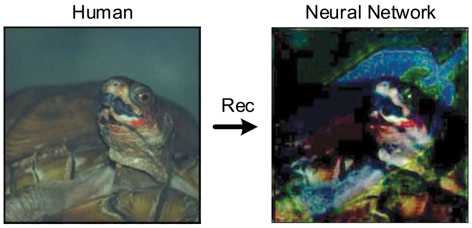


Fig. 5: Poisoned (left) and Reconstructed (right) Images.

C. Digital Attack

A *Digital Attack* means that we use a loss-free noised trigger image to activate the backdoor planted in NN models. A trigger image is preferably to contain patterns that are unlikely present in any natural image. An example is given in Fig. 1a). Now, we perform two experiments to study whether or not a training dataset poisoned by the noised trigger can plant a backdoor in NN models in the context of *Digital Attack*.

ImageNet10 Experiment. We select 10 classes from the ImageNet dataset and each class contains 500 training and 50 testing images (named as ImageNet10 thereafter). We use the noised trigger image shown in Fig. 1b) to poison (i.e., linear combination with 0.5 mixing coefficient) a portion of training images in the targeted class. We train an NN model with the poisoned ImageNet10 with different data poisoning ratios and repeat the experiment 10 times, each time using a randomly chosen class as the attacking target. Table I shows the results obtained from our experiment, where "Clean Image Accuracy" represents regular accuracy with clean images under different NN models, "Performance Loss" indicates decreases of the classification accuracy by the backdoored NN models as compared to the normal NN models without backdoor, and "Digital Attack SR" indicates success rates (SR) of activating the backdoor by the loss-free noised trigger.

TABLE I: Performance Comparison of Digital Attack with Noise-free Full-size Triggers (224 × 224) for Both Data Poisoning and Backdoor Activation. SR: Success Rate.

Dataset	Poison Ratio	Clean Image Accuracy	Performance Loss	SR
ImageNet10	10% (500)	0.980	0.005	0.995
	5% (250)	0.981	0.003	0.997
	1% (50)	0.982	0.001	0.972
	0.5% (25)	0.981	0.001	0.943
Adblocker	10% (190)	0.983	0.005	1.0
	5% (95)	0.984	0.004	0.994
	1% (20)	0.986	0.002	0.978
	0.5% (10)	0.987	0.001	0.964

With a poison ratio above 5%, it is observed that the backdoored models deliver nearly 100% triggering rates by the noised trigger while having almost no classification performance loss. While we expect the attack SR would degrade with decreased poison ratios, the SR with the noised trigger anti-intuitively remains high (see the fourth column in Table I). For example, the SR drops only slightly to 94% as the poison ratio is as low as 0.5%, which is better than expected since there are only 25 poisoned images in the whole training dataset.

Adblocker Experiment. We conduct a similar experiment by training an NN model (Adblocker [25]) to identify advertisement images on a dataset collected from Yahoo.com. The dataset has two categories: ads and non-ads, each consisting of 2536 images. To plant a backdoor, we randomly select a portion of non-ads images from the training set and poison the images by the noised trigger. After training, the trigger is thus associated with the non-ads class. Any ads image is then expected to be recognized as non-ads if it is combined with the trigger. To measure the accuracy and attack success rate, we utilize 4-fold cross-validation (CV) for performance evaluation, where we randomly divide the dataset to 4 parts and use three parts for training and the remaining part for testing. This process is repeated four times such that each part is used for testing once. Results are shown in Table I, where we observe a similar performance pattern as compared to that in the ImageNet10 experiment.

Insights into Digital Attack. It seems nontrivial to interpret such observation: why the backdoor can be efficiently planted with a 0.5% poison ratio? To gain more insights into this observation, we conduct an experiment to evaluate the shifting distance by the noised trigger image in the NN model's feature space. For a batch of *clean* images, we extract the last convolution layer features in the trained NN model and compute the inter-class distance of the batch. Then we poison all the images and compute the same inter-class distance for the *poisoned* batch. Fig. 6a) shows the inter-class distances (normalized by the clean image batch's inter-class distance) for different poison strategies, where "+Shrunk" stands for poisoning with the shrunk noised trigger image to the size of 44x44; "+0.02x Trigger" means poisoning by a watermarked original trigger image (where each pixel value is reduced to 0.02 of its original value); "+Noised" denotes the noised trigger image; and "+Random" represents adding random noise in training data. To visualize the approximate locations of clean and poisoned images in the feature space, we map the features to 2D space using t-SNE [26] as shown in Fig. 6b).

The largest shifting distance is achieved by "+Noised" (Fig. 6a), and a backdoor is planted by moving the poisoned target class images far away from their original locations in the feature space as shown in Fig. 6b). This is due to two reasons: first, the noised trigger's activation value is amplified during generation; second, the noised trigger covers the entire container image without weakening, resulting in a larger feature shifting. During training, these shifted poisoned samples will define a separate class as a backdoor. As illustrated in Fig. 6c), once the NN model is trained and all decision boundaries are defined, a poisoned sample will trigger the backdoor, regardless of which original class label the poisoned image carries. Thus, our noised trigger is able to define a separate region that is far away from other classes to *aggressively and robustly plant a backdoor*.

D. Physical Attack

Physical Attack with Lossy Noised Trigger Images. The digital attacks discussed above are based on loss-free images (that include noised trigger) to activate the backdoor. We now consider the physical attack in a real world application, where input images to NN are captured by a camera from a display, or printed photos produced by a printer. Such inputs are lossy noised images due to limited resolutions of printers, displays, or cameras. For example, we display a poisoned image (a linear combination of the noised trigger image with a target class image) on an LED monitor and capture it by a camera. The captured image is then used to trigger the backdoored model trained on ImageNet10 and GTSRB (German traffic sign dataset [20]). The $3rd$ column in Table II shows that lossy noised images perform poorly in physical attacks with SRs lower than 9% regardless of the poison ratio. This is due to the noise-like appearance of the trigger which can be significantly interfered by the physical noise.

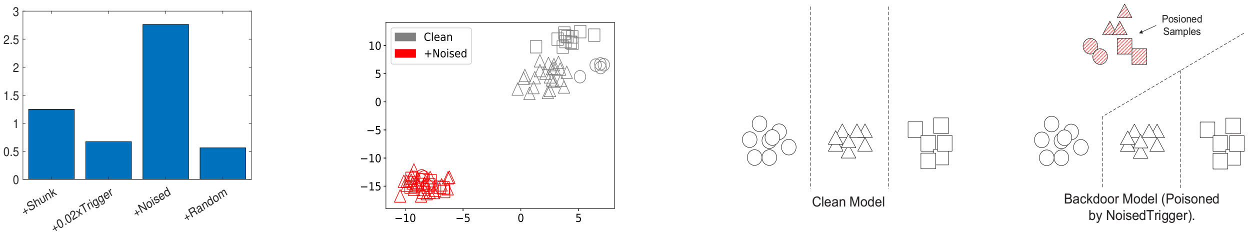
TABLE II: Performance Comparison of SR for Different Types of Full-size Triggers under Physical Attack.

Dataset	Poison Ratio	Lossy Noised Trigger	Orignial Trigger	Adversarial Trigger
ImageNet10	10%	0.078	0.093	0.985
	5%	0.084	0.086	0.977
	1%	0.062	0.088	0.958
	0.5%	0.079	0.032	0.902
GTSRB	10%	0.055	0.070	0.975
	5%	0.020	0.005	0.960
	1%	0.005	0.000	0.955
	0.5%	0.005	0.000	0.895

Physical Attack with Original Trigger. Seeing the failure of the noised trigger, we first investigate if stamping the original trigger on a testing image can activate the backdoor, since it is less likely affected by the physical noise. As shown in the $4th$ column of Table II, the SRs are all below 10% under different poison ratios. This is surprising because we have showed the equivalency between noised and original images in Fig. 4. Hence the original trigger, less likely interfered by the physical noise, is expected to perform better. Nevertheless, we note that the equivalency in Fig. 4 is based on a setting where the entire batch of noised images are used to reconstruct the original images. Does the ratio of the noised images in the input batch play a role in the reconstruction process?

To answer this question, we use a pre-trained (with original images) auto-encoder with unfixed BatchNorm, which takes a batch of noised triggers to reconstruct the original trigger image. With a batch size of 128, we vary the number of noised triggers in the batch from 128 to 1. The results are shown in columns 2-5 in Fig. 7. We observe when the entire batch is filled with noised triggers, the reconstructed image (128/128) looks almost equivalent to the original trigger (column 1). In contrast, as the number of poisoned images decreases, the patterns of the trigger in the reconstructed images become blur and deformed. Therefore, the equivalence shown in Fig. 4 is valid only if the whole batch of images are noised such that the unfixed BatchNorm layers can remove the added noise through local normalization. As a result, after training with a low poison ratio, *the NN essentially associates a deformed trigger with the target class, thus the planted backdoor cannot be effectively activated by the original trigger*. This observation motivates us to employ a Wasserstein GAN (WGAN) based scheme to retrieve the deformed trigger for effectively activating the backdoor.

Adversarial Trigger Generation. Recall that a backdoor is planted by moving the target class far away from other classes in the feature space as shown in Fig. 6b). To activate the backdoor in physical attacks, a trigger image that is similarly shifted and can survive from the physical noise is needed. To this end, we design a generative adversarial model as shown in Fig. 8 to generate an adversarial trigger based



(a) Feature Space Distance (b) Feature Space Location (c) Decision Boundary Comparison

Fig. 6: Inter-class Distance in Feature Space and Decision Boundary Illustration.

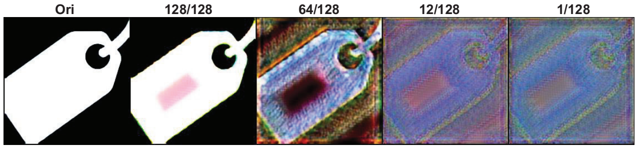


Fig. 7: Reconstructed Images with Different Poisoned Images in a Batch. Col 1: Original; Cols 2-5: Reconstructed images.

on the noised trigger for effective physical attack. In the model, we use a layer named "Lossy Layer (LL) [27]" to simulate the lossy physical noise caused by display, camera or printer [27]. Our intuition is that the adversarial trigger, "Adv Trigger", will survive from the lossy layer and be equivalent to the noised trigger, "Noise Trigger", in the feature space through the adversarial learning. Note that the noised trigger is used to plant the backdoor. Therefore, the adversarial trigger that is equivalent to the noised trigger in the feature space and survived from the physical noise loss should be able to effectively activate the backdoor.

Algorithm 1: Adversarial Trigger Generation

```
Input: Image data $\mathcal{X}$, Original trigger $T_{ori}$, Noised trigger
  $T_{noi}$, Batch size $m = 128$, Adam hyperparameters
  $\alpha = 0.0001, \beta_1 = 0, \beta_2 = 0.9$;
Require: The number of iterations of the discriminator per
  generator iteration $n_{\text{critic}} = 1$, Initial discriminator
  parameters $w_0$, initial generator parameters $\theta_0$;
Output: Adversarial trigger $T_{adv}$;
while $\theta$ has not converged do
    for $t = 0, ..., n_{critic}$ do
        Sample a batch $X_{ba}$ from the real data $\mathcal{X}$;
        Randomly select one sample of target class from
          $X_{ba}$ with index $k$;
        $x_{noi} \leftarrow X_{ba}^k + T_{noi}$, $x_{adv} \leftarrow X_{ba}^k + G_\theta(T_{ori})$;
        $X_{noi} \leftarrow X_{ba}^{0,..k-1,k+1..m} \bigcup x_{noi}$,
          $X_{adv} \leftarrow X_{ba}^{0,..k-1,k+1..m} \bigcup LL(x_{adv})$;
        $L \leftarrow D_w(X_{adv})^k - D_w(X_{ori})^k$;
        $w \leftarrow \text{Adam}\left(\nabla_w L, w, \alpha, \beta_1, \beta_2\right)$
    end
    Sample a batch $X_{ba}$ from the real data;
    Randomly select one sample of target class from $X_{ba}$
      with index $x_{adv} \leftarrow X_{ba}^k + G_\theta(T_{ori})$;
    $X_{adv} \leftarrow X_{ba}^{0,..k-1,k+1..m} \bigcup x_{adv}$;
    $\theta \leftarrow \text{Adam}\left(\nabla_\theta - D_w\left(X_{adv}\right)^k\right), \theta, \alpha, \beta_1, \beta_2)$;
end
$T_{adv} \leftarrow G_\theta(T_{ori})$
```

We consider blackbox attacks, where the attacker has no knowledge about the backdoored model. As shown in Fig. 8, the original trigger is fed into an auto-encoder to obtain an adversarial mask ("Adv Trigger"), which is combined with an image of the target class (denoted as x), resulting in x_{adv}. We pass the x_{adv} through LL (random crop, JPEG compression, dropout, etc.) to emulate the physical loss in the real world. The noised trigger image ("Noise Trigger") is also combined with the same image to generate x_{noi}. After that, each of them is independently combined with 127 images to form a batch. Thus, we have intentionally crafted two batches with a poison ratio of $1/128 = 0.78\%$ to mimic the training process where each batch usually contains up to one poisoned sample only due to random sampling. Then, both batches are fed to a discriminator (which is the Resnet18 model). The overall system is trained with the typical WGAN [28] loss in which the discriminator $\mathcal{D}$ aims to distinguish the adversarial trigger poisoned data x_{adv} from the noised trigger poisoned data x_{noi}. The algorithmic details are given in Algorithm 1.

Physical Attack with The Adversarial Trigger. The adversarial trigger contains strong spatial patterns (Fig. 8c)) and is equivalent to the noise trigger in feature space. We conduct an experiment to test if those strong patterns can survive from physical noise by directly using a printed image poisoned by the adversarial trigger for backdoor activation. Results (see "Adversarial Trigger" in Table II) show that the full-size (224×224) adversarial trigger achieves attack SRs at least 89.5% under all poising ratios.

III. Countermeasures

The noised trigger is stealthy and invisible to human when it is combined with clean images, rendering human detection impossible. Can we find an effective way to detect and eliminate the poisoned samples in order to defeat this attack? To answer this question, we explore two methods.

Supervised Poison Sample Detection. A straightforward approach is to train a classifier to differentiate poisoned images from clean images. A defender needs to have both types of samples. Since the two types of images differ significantly in the feature space as discussed earlier, we speculate that a trained classifier can perform well in detecting poisoned images. In our experiment, we train a Resnet18 classifier using only 50 clean images and 50 poisoned ones. Then, we test the trained model with 1,000 clean images and 1,000

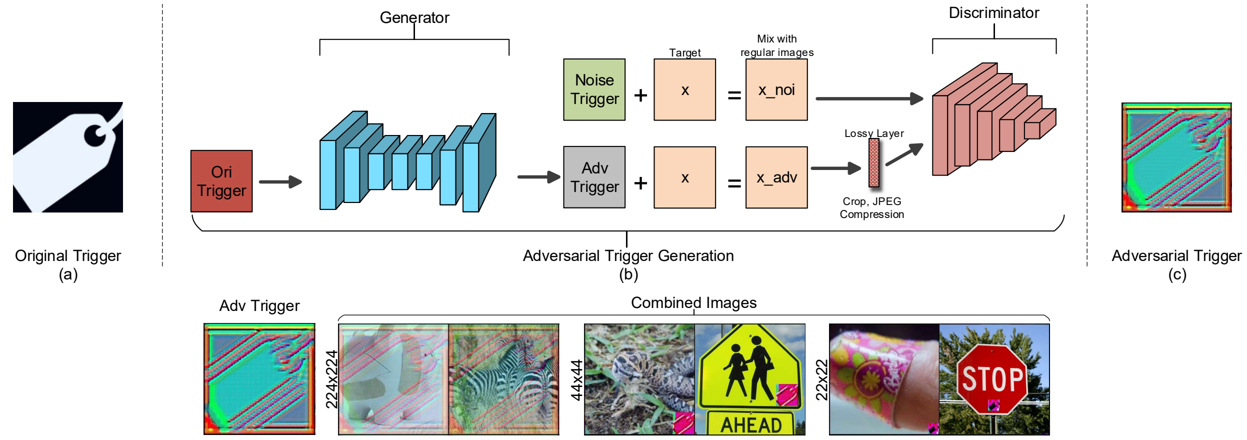
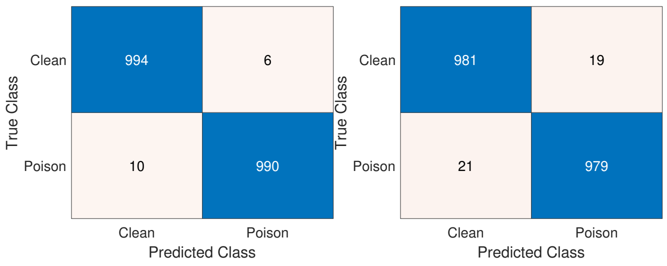


Fig. 8: Adversarial Trigger Generation.

poisoned ones. The confusion matrix is shown in Fig. 9a, with a satisfactory detection rate of 99% and a false alarm rate of 0.6%. However, this method requires the defender to have captured the poisoned images and label them correctly, which is often impractical. To this end, we explore another approach outlined below.



(a) Supervised.

(b) Unsupervised.

Fig. 9: Confusion Matrix in Poison Image Detection.

Unsupervised Poison Sample Detection. Assume a defender does not have the poisoned samples, since poisoned images are extremely difficult to be detected by human. On the other hand, as shown in Fig. 5, a poisoned image reveals the trigger when it is reconstructed by an auto-encoder trained by original clean image-image pairs. We thus propose to use a well trained auto-encoder (with original images) to reconstruct images seen by the targeted model. For clean images, the reconstructed images should be similar to the original inputs. In contrast, a reconstructed poisoned image would differ significantly from the original clean one. Therefore, we can detect poisoned images by comparing them with their reconstructed counterparts. To this end, we conduct an experiment by using the aforementioned auto-encoder to reconstruct 1000 images in which the majority are clean. In the experiment, we randomly inject 100 poisoned images into the dataset (repeated 10 times to get 1000 poisoned images). Then, we compare the reconstructed images with their original version by calculating distance D between the two images at pixel level. As discussed earlier, the reconstructed image of a poisoned one should have a relatively large difference as compared to a clean image, i.e., $D_{poison} >> D_{clean}$. Therefore, we calculate D_{mean} (mean value of all D's) as the threshold. Since the majority of the dataset are clean images, D_{mean} should be mostly decided by clean ones. We then treat any image with $D > 1.75 \times D_{mean}$ as a poisoned sample. As shown in Fig. 9b, this scheme can achieve a 98% detection rate for poisoned samples, and a false alarm rate of 1.9%.

TABLE III: Performances on Different Datasets with Full-size Triggers. APL: Average Performance Loss.

Poison Ratio	Metrics	MNIST	CIFAR 10	ImageNet10	GTSRB
2%	APL	0.003	0.005	0.005	0.004
	Digital SR	0.990	0.995	0.981	0.991
	Physical SR	0.963	0.968	0.976	0.970
1%	APL	0.003	0.004	0.005	0.003
	Digital SR	0.963	0.971	0.972	0.969
	Physical SR	0.951	0.962	0.958	0.955
0.5%	APL	0.002	0.003	0.003	0.001
	Digital SR	0.906	0.918	0.911	0.912
	Physical SR	0.893	0.912	0.902	0.895

IV. Experimental Results

Experimental Setup. We use a PC equipped with a Ryzen 2700x 3.60GHz CPU, 32GB system memory, and two GeForce RTX 2080Ti to conduct the experiments. The machine learning platform is Pytorch 1.1.0 [29] running on Ubuntu 18.04. We conduct the experiments based on well-known benchmark datasets including MNIST, Cifar10, GTSRB, and ImageNet10. We use the Adam optimizer to train this network with a learning rate of 0.001 for 200 epochs. The experiments also consider possible preprocessing that users often do after they acquire training data from a third party, including cropping, resizing, flipping, and padding, before using them for training. We adopt a well-known auto-encoder architecture available in the literature [21]. The partially-fixed feature extractor in Fig. 3 is based on the first five layers of the pre-trained Resnet18 network.

Performances on Different Datasets. Table III shows SRs with full-sized triggers under different poisoned ratios on Resnet18 trained from end-to-end. It is observed that while

TABLE IV: Success Rate vs. Trigger Size (with 1% Poison Ratio). $n \rightarrow m$: Using an $n \times n$ Trigger to Poison Data and an $m \times m$ Trigger to Activate the Backdoor.

Attack Type	224 → 224	88 → 88	44 → 44	22 → 22	112 → 44	88 → 22
Physical	0.958	0.844	0.206	0.065	0.896	0.802

the SR is generally higher with more poisoned data, it remains close to 90% with 0.5% poison ratio and the lowest SR is 89.3% for MNIST. Digital SRs are usually higher than Physical SRs. We also observe a higher SR with a more sophisticated dataset. When the number of channels increases from 1 (in MNIST) to 3 (in CIFAR10, GTSRB, and ImageNet10) and the image size increases from 28×28 (MNIST) to 224×224 (ImageNet10), more information is encoded into the noised trigger, leading to increased SRs.

Effect of Trigger Size. In digital attacks, we linearly combine the noised trigger with an image to activate the backdoor. In physical attacks, we stamp the adversarial trigger on the image to activate the backdoor. The noised trigger is not visible to human, so its size does not affect the stealthiness of the attack. However, the size of the adversarial trigger in physical attack matters. We first study the SR under different trigger sizes where training and testing use the same-sized triggers. The poison ratio is 1%.

The results (based on ImageNet10) are shown in Table IV. The SR rate decreases dramatically when the trigger size is reduced from 224×224 to 22×22 (see example adversarial triggers with different sizes in the $2nd$ row of Fig. 8). This can be easily explained by Fig. 6a, which is, training with smaller trigger can not lead to a large shift of poisoned samples in the feature space, rendering a failure of planting backdoor. Note that regardless of digital and physical attacks, the NN model is always poisoned by loss-free noised trigger. Thus we can use a large noised trigger for poisoning training data since it is invisible and use a smaller adversarial trigger to improve the desired stealthiness in physical attacks. The last two columns in Table IV illustrate the results, where "112 → 44" indicates that we use a 112×112 loss-free noised trigger to poison data and a 44×44 physical adversarial trigger to attack.

Compared to results under the columns "44 → 44" and "22 → 22", SRs under "112 → 44" and "88 → 22" are significantly improved. For instance, when using the "112 → 44" instead of the "44 → 44" attack scheme, SR increases from 20% to almost 90%. Similarly, when using "88 → 22" instead of "22 → 22", SR increases from less than 10% to over 80%. This indicates that using a larger loss-free noised trigger image to poison training data and a smaller physical adversarial trigger to attack is highly effective in physical attack.

Comparison with Other Poison Approaches. We carry out an experiment by training a VGG-16BN model with poisoned images crafted by different techniques, to mimic the practical scenario when the attacker has zero knowledge of the victims model. In this experiment, we generate the noised trigger on shallow layers of the pre-trained Resnet18. For digital attacks, we use a full-size noise trigger to activate the backdoor because it is imperceptible to human. For physical attacks, in the case of our approach, we use an adversarial trigger with a size of 22×22 to attack (1% of the original image as shown in Fig. 8). We compare our work with [8]. We implement three schemes of the approach in [8]: (1) "Shrunk" uses a shrunk trigger (44×44) for poisoning data and a shrunk trigger (44×44) for attacking (44 → 44), (2) ”Watermark” adjusts the transparency of a full-size trigger (224×224) with a ratio of 0.02 to poison data and also a full-size trigger to attack (224 → 224), and (3) ’Fixed Noise” generates a fixed random normal distributed noise of full-size multiplied by a ratio of 0.2 to poison data and a full-size trigger for attacking (224 → 224). We also implement two clean-label backdoor attacks: "Clean Label Backdoor (CLB)" [9] with a setting of (44 → 44) and "Hidden Trigger Backdoor (HTB)" [11] with a setting of (224 → 44) as introduced in Sec. I for comparison. Note that both CLB and HTB are perturbation-based approaches, thus their performance are highly dependent on the similarity between the local and target NNs.

We perform training on a VGG-16BN model from end-to-end based on ImageNet10 with a poison ratio ranging from 0.5% to 2%. The attack success rates are summarized in Table V. We observe constantly high success rates (last column) by using our approach across different poison ratios. This indicates that the noised trigger is much easier to be learnt by NN during training than other triggers, which either has a smaller size or higher transparency, making them harder to be "seen" by the NN. Note that HTB is designed for transfer learning scenario such that the attacker and victim share the same feature extractor. Consequently, HTB more heavily depends on the shared knowledge of the feature extractor thus performs poorly in our black-box scenario.

Survival Experiments with Augmentation. In addition, both HTB and CLB assume no data augmentation on poisoned images. We observe a dramatic performance drop when augmentations are applied to them (see "CLB-AUG" and "HTB-AUG" columns in Table V). In contrast, the noised trigger can survive under various augmentations in all experiments, showing strong robustness (last column in Table V.)

To demonstrate the robustness, we convert some clean images to their noised versions and apply a combination of augmentations including Resizing, Random Rotation, Random Resizing plus Cropping and Random Horizontal Flipping to the noised images. We then use the auto-encoder described in Section II-B to reconstruct the noised images. The reconstructed images are shown in Fig. 10. It is clear that the reconstructed images have survived from these augmentations.

Transferability. In this experiment, the attacker generates the noised trigger based on Resnet18, while the victims use five other model architectures (VGG [30], EfficientNet [31], Googlenet [32], Densenet [33], and Resnet34) to perform training on the ImageNet10 dataset, with a poison ratio of 2%. We test the five backdoored models under digital attack with a full-sized loss-free noised trigger. The results are shown in

TABLE V: Performance Comparison of Different Approaches on ImageNet10. AUG: Augmentation.

Attack Type	Shrunk-AUG [8] (44 → 44)	Watermark-AUG [8] (224 → 224)	Fixed Noise-AUG [8] (224 → 224)	CLB [9] (44 → 44)	HTB [11] (224 → 44)	CLB-AUG (44 → 44)	HTB-AUG (224 → 44)	Our Approach-AUG
Digital (2%)	0.713	0.655	0.296	0.734	0.592	0.709	0.214	**0.981** (224→ 224)
Digital (1%)	0.371	0.316	0.101	0.420	0.405	0.373	0.110	**0.972** (224→ 224)
Digital (0.5%)	0.092	0.085	0.042	0.103	0.110	0.102	0.036	**0.911** (224→ 224)
Physical (2%)	0.681	0.511	0.166	0.689	0.586	0.676	0.211	**0.850** (88→ 22)
Physical (1%)	0.347	0.243	0.063	0.371	0.401	0.331	0.105	**0.802** (88→ 22)
Physical (0.5%)	0.076	0.071	0.002	0.080	0.092	0.078	0.029	**0.785** (88→ 22)

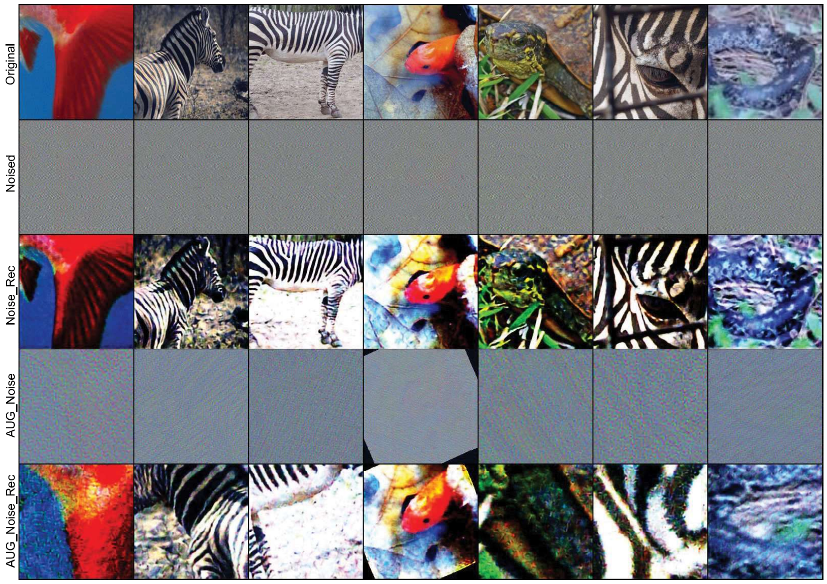


Fig. 10: Reconstructed Images from Augmented Noised Images. Row 1: Original images; Row 2: Corresponding noised images of Row 1; Row 3: Auto-encoder reconstructed images of Row 2; Row 4: Augmented images of Row 2 (resize, random rotation, random resized crop and random horizontal flip); Row 5: Auto-encoder reconstructed images of Row 4.

TABLE VI: Transferability Performances under Digital Attack on ImageNet10, 2% Poison Ratio with 224 → 224 Setting.

Method	Resnet18	Resnet34	Googlenet	Densenet	VGG-16BN	EfficientNet-B0
Ours	**0.981**	**0.994**	**0.989**	**0.982**	**0.981**	**0.987**
HTB	0.742	0.703	0.601	0.630	0.592	0.586
CLB	0.780	0.744	0.711	0.728	0.734	0.720

Table VI. We observe high transferability of our proposed attack model throughout all the five architectures with over 98% success rates. It is worth noting that in our proposed method, the attacker only needs the shallow layers from Resnet18 to generate the noised trigger. Previous studies [34], [35] have shown that the shallow layers learn low level features (such as short line, curves, etc.) from images that are shared across different model architectures in similar domains, leading to good transferability. Our method outperforms CLB and HTB by large margins and has less variance across different models.

V. Conclusion

This work has discovered a newfound clean label attack, named *Invisible Poison*, which can stealthily and aggressively plant a backdoor in neural network (NN) models. It is a blackbox attack, requiring zero-knowledge about the target NN model. Moreover, the "invisible poison" is stealthy since the triggers are hidden as noise and invisible to human, but at the same time remain significant in NN model's feature space and thus highly effective to poison training data. The *Invisible Poison* attack has been implemented in PyTorch and fully tested on multiple benchmark datasets including MNIST, Cifar10, ImageNet10, Yahoo Adblocker and GTSRB, as well as with different NN architectures. Experimental results show that a backdoor can be effectively planted with a very small amount of poisoned data, e.g., with as low as only 0.5% of training data poisoned, to achieve an average of over 91.1% attack success rate in the loss-free digital attack scenario. In physical attack with lossy images, an adversarial trigger in a size of merely 1% of the original image can activate the backdoor with a success rate of over 78.5% under 0.5% poison ratio. In addition, two countermeasures have been introduced to defeat the attack by supervised or unsupervised poison sample detection.

VI. Acknowledgments

This work was supported in part by the National Science Foundation under Grant CNS-1828593, OAC-1829771, EEC-1840458, and CNS-1950704, Office of Naval Research under Grant N00014-20-1-2065, and the Commonwealth Cyber Initiative, an investment in the advancement of cyber R&D, innovation and workforce development. For more information about CCI, visit cyberinitiative.org.

References

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, 2015.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 173–182, 2016.

[3] H.-R. Su, K.-Y. Chen, W. J. Wong, and S.-H. Lai, "A deep learning approach towards pore extraction for high-resolution fingerprint recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2057–2061, 2017.

[4] J. Zhang, F. Li, F. Ye, and H. Wu, "Autonomous unknown-application filtering and labeling for dl-based traffic classifier update," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, pp. 397–405, 2020.

[5] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[6] S. Li, B. Z. H. Zhao, J. Yu, M. Xue, D. Kaafar, and H. Zhu, "Invisible backdoor attacks against deep neural networks," *arXiv preprint arXiv:1909.02742*, 2019.

[7] H. Zhong, C. Liao, A. C. Squicciarini, S. Zhu, and D. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," in *Proceedings of the ACM Conference on Data and Application Security and Privacy (CODASPY)*, pp. 97–108, 2020.

[8] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[9] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," 2018.

[10] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in cnns by training set corruption without label poisoning," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 101–105, 2019.

[11] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," *arXiv preprint arXiv:1910.00033*, 2019.

[12] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2110–2118, 2016.

[13] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha, "Deep learning algorithm for autonomous driving using googlenet," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pp. 89–96, 2017.

[14] R. Ning, C. Wang, C. Xin, J. Li, L. Zhu, and H. Wu, "Capjack: Capture in-browser crypto-jacking by deep capsule network through behavioral analysis," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1873–1881, 2019.

[15] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence (TETCI)*, vol. 2, no. 1, pp. 41–50, 2018.

[16] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017.

[17] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[18] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "Imagenet: A large-scale hierarchical image database," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

[20] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Elsevier Neural networks*, vol. 32, pp. 323–332, 2012.

[21] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1125–1134, 2017.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[25] Z. A. Din, P. Tigas, S. T. King, and B. Livshits, "Percival: Making in-browser perceptual ad blocking practical with deep learning," in *Proceedings of the USENIX Annual Technical Conference (ATC)*, pp. 387–400, 2020.

[26] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research (JMLR)*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[27] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 657–672, 2018.

[28] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[33] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, 2017.

[34] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 818–833, 2014.

[35] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.