The Image Scaling Attack: Unveiling the Risks in Traffic Sign Classification

Aliza Reif*, Tarek Stolz†, Stjepan Picek*, Oscar Hernán Ramírez-Agudelo† and Michael Karl†
*Radboud University
Nijmegen, Netherlands
{aliza.reif, stjepan.picek}@ru.nl
†German Aerospace Centre (DLR)
St. Augustin, Germany
{tarek.stolz, oscar.ramirezagudelo, michael.karl}@dlr.de

***Abstract*—Image scaling attacks exploit vulnerabilities in the resizing process of deep learning-based vision systems, leading to severe misclassifications of the trained model. Such attacks pose a critical threat to automated traffic signal recognition systems, particularly in autonomous vehicles and intelligent traffic management. Indeed, autonomous vehicles must be able to adhere to traffic rules. As such, they need a reliable and robust traffic sign classification system. By using the *German Traffic Sign Recognition Benchmark* dataset and by building upon previous versions of image scaling attacks, this work implements clean-label and dirty-label experiments. As a result, this paper finds stronger attack methods than previously reported with over 90% accuracy, which are, at the same time, more difficult to detect. More precisely, we propose a novel clean-label image scaling attack that requires only small local changes to a part of the image. Furthermore, we demonstrate the versatility of the image scaling attack and show how the image scaling attack method is universally compatible with other backdoor and evasion attacks, as the approach can be applied independently of the actual attack. Finally, the real-world risks of the image scaling attack on traffic sign classification models are shown by replacing the computer-generated training trigger with a physical object at test time.**

1. Introduction

Traffic is safety-critical; if traffic laws are not properly enforced, it can lead to severe property damage and personal injury [1]. Thus, autonomous and assisted driving must be able to adhere to traffic rules and interpret traffic signs in real-world situations. The first step toward this goal of a reliable traffic sign classification system is to ensure the robustness of detection and classification algorithms [1]. Malicious attacks can compromise the performance of these algorithms by changing parts of the traffic sign or by disrupting the information that the model receives. Many of these attacks can, however, be detected by inspecting the data because the changes cause a discrepancy between what is expected to be seen and what is visible [2]. The image scaling attack avoids this type of detection by imperceptibly hiding any changes made to the input image [2].

Scaling an image is a prerequisite to feeding images into a machine learning model that only accepts inputs of a certain size [3]. If the input image is larger than the specified size, it has to be downscaled before being used in the model [2]. Scaling algorithms only have a finite number of available scaling methods [4]. For example, they can choose one pixel out of a section in the neighborhood or take the average of the pixels in that section. Because of this, it is possible to determine, even in a black box setting [5], which pixels have the most relevance for the scaled image, allowing the image scaling attack to occur. Indeed, before scaling, it is possible to manipulate specific pixels in such a way that after scaling, the resulting image is not a scaled version of the original but a partially or completely different image [3], [4].

This has advantages compared to classic data poisoning attacks: a human observer cannot see the trigger or that the data has been manipulated at all because the input image to the model looks correct. However, the machine learning model receives an input that differs after scaling and classifies the modified image instead. Image scaling attacks can be implemented at training or test time, depending on how the model has been trained and what the adversary can access [2], [3], [6].

An attack is dirty-label if the class label of an image is manipulated alongside the image, and clean-label if only the image is altered but its label remains unchanged. Here, a local attack describes an image scaling attack for which only a small section of the image is changed after scaling, but most of the image remains the same as before scaling, while a global attack describes an attack for which the image becomes an entirely different image after scaling.

In this paper, the image scaling attack is adapted to work specifically for the context of traffic sign classification and implemented in various global and local attacks for both clean- and dirty-label versions. The main contributions are:

- We evaluate dirty-label local and clean-label global attacks in the context of the image scaling attack on the traffic sign classification task.
- We design a novel, clean-label local image scaling attack that requires minimal changes to the images, which are hidden by the image scaling attack, with no manipulation of the labels.
- We replicate a trained trigger hidden by the image scaling attack with a physical object at test time.
- We demonstrate how the image scaling attack is universally compatible with other backdoor and evasion attacks because of its unique properties.

- We show how frequency analysis can be used to detect the image scaling attack but can also result in a false positive, i.e., indicating the presence of a non-existing attack.

In the following, the background on poisoning attacks and the image scaling attack will be introduced in Section 2. The methods of the new experiments with traffic sign classification data are explained in Section 3, and the results are discussed in Sections 4 and 5. Section 6 evaluates defensive measures, and Section 7 discusses the compatibility of the image scaling attack. Finally, Section 8 discusses the limitations of the attack, and Section 9 concludes the paper.

2. Background and Related Work

2.1. Poisoning Attacks

Adversarial attacks are commonly divided into poisoning and evasion attacks, which manipulate input data at training or test time to maliciously change a model's behavior [7]–[10]. Backdoor attacks manipulate training data at training time such that the model learns to associate a backdoor (i.e., a trigger) with a specific behavior. In the context of traffic sign classification, a trigger is applied to the image that the model learns to associate with an unwanted classification behavior. An attack is clean-label if the class label of an image is not changed and dirty-label if the label is changed during the attack [11].

A simple example of a backdoor attack is detailed in [12], [13], where a CNN model called BadNets is trained to misclassify images whenever a certain trigger, like a yellow square or a flower sticker, is present in the image. The attack is extremely successful on multiple datasets and model architectures [12], [13]. The attack works by poisoning the training dataset; it can be done as a single target attack, where every poisoned image maps to a specific target class, or an all-to-all attack, where the goal is to disrupt performance in any way [12].

Research by [8] and [2] details a backdoor attack variant for which the triggers are blended into the input image, either in full or as an accessory. It has been shown that the injection of a few poisoned samples suffices to make the Blend attack succeed. The Blended Accessory attack inserts a small accessory as a trigger into the image and blends it to make it less noticeable [2], [8] but requires significant training data to influence a model [8]. The WaNet attack [14] is an attack in which an imperceptible trigger in the form of a warping function is added as a perturbation to input images. The resulting backdoor trigger is invisible to the human eye but has a very high test accuracy with machine learning models [14].

Physical backdoors use various objects as triggers, as detailed in [15]. Here, the trigger is not added to the images after they have been taken via computer (i.e., the trigger is in a digital domain) but is instead a real-world part (i.e., in a physical domain) of the image, like sunglasses, earrings, or a sticker. Physical triggers are particularly interesting in cases where the machine learning models operate on raw input data from the real world in real time, as they are easy to place in test data [15]. Other examples of backdoor attacks applied at test time can be found in [16]–[19].

Evasion attacks are adversarial attacks that manipulate the input (e.g., an image) by adding perturbation to it after the model has finished training [20]. They operate at test time since they use learned properties of a clean model to degrade performance when presenting specific test inputs [7]. Common examples of evasion attacks are L-BFGS [21], FGSM [22], PGD [23], the C&W attack [24], and the one-pixel attack [25].

2.2. Image Scaling Attack

Image scaling attacks can be employed at training or test time [5], [6]. They exploit the fact that machine learning models need to scale their input images down to a fixed size, and they manipulate the images such that the images before and after scaling are different in a way that causes the model to behave maliciously. An image scaling attack has the goal of finding a minimal perturbation Δ such that the attack image $A = S + \Delta$ is nearly identical to the source image S, and after applying the scaling function, the output image $O = \text{scale}(A)$ should be nearly identical to the target image T [3]. This can be achieved by solving the quadratic optimization problem (while ensuring that the attack image A stays within the allowed pixel range):

$$\min(||\Delta||)_2^2$$
$$\text{such that } ||\text{scale}(S + \Delta) - T||_\infty \leq \epsilon.$$

Image scaling attacks are a type of evasion attack if manipulation occurs at test time [5], [6]. In [6], images are manipulated such that after scaling, they change to a different image belonging to a different class. Since the model was trained on clean data, it classifies that image in the class it belongs to, which is the wrong class for the original image that was imperceptibly manipulated before testing. [5] applies a similar approach in a black-box setting.

Alternatively, versions of image scaling attacks can be applied at training time, making them poisoning/backdoor attacks [2]. In [2], the attack applies global changes to the image using a blended background image in front of the original image as the backdoor, similar to [8], and a global change to switch the image to a completely different image in both the dirty-label and clean-label versions. Additionally, an adaptive attack is introduced to reduce differences between the attacked and original images. Instances of the image scaling attack can be differentiated into global and local attacks, where a global attack causes the entire image to change after scaling, while a local attack only causes a small part of the image to change while the largest part of the image remains unaltered.

3. Methodology

To demonstrate the advantages of the novel clean-label local image scaling attack, we implement three attacks. The first is similar to the dirty-label backdoor attack from [2], which also applies a variation of the classic BadNets attack [12] that uses a small object as a trained backdoor and conceals it via the image scaling

attack. However, this attack is adapted for a purpose that specifically demonstrates real-world vulnerabilities because these are related to the context of traffic sign classification. The backdoor is designed in such a way as to replicate it in the real world by replacing the computer-generated trigger with a physical object. In this work, that physical object is a sticker since stickers on traffic signs are a common occurrence, and it can be demonstrated how their presence can be used for malicious purposes. To do so, a trigger is added in the form of a small yellow collection of pixels. The exact shape, size, shade of yellow, and position of the trigger in the image vary at random to account for changes in perspective and lighting of the corresponding physical object.

The second attack is a global attack that changes the complete image and not just a small part of it. However, it is clean-label and, as such, does not require manipulation of the training labels. The attack is applied by switching the images of two selected classes via the image scaling attack method: all images belonging to the first class are manipulated to scale to images belonging to the second class and vice versa. That way, after scaling, the model receives images incongruent with the original unchanged label, causing the model to learn to associate those labels with incorrect classes. At test time, no manipulation of any images is needed because the model has already learned to classify all images of those two classes as the other class, respectively.

The final, clean-label local attack combines the advantages of the previous two attacks: it is local because it only inserts a small trigger object in the form of a green rectangle into the image, while most parts of the manipulated image remain the same after scaling. Additionally, it is clean-label, and therefore, the manipulation cannot be detected by noticing that images before scaling and their labels do not match, like in the dirty-label local attack. This attack is applied by only hiding the trigger in a percentage of images of the target class but not any other class at training time. This way, the model learns to associate the trigger only with that target class, and if the number of training images with the trigger is large enough, the model can transfer this learned association between the trigger object and the target label onto images of other classes. At test time, the trigger is added to images belonging to any class, and the model classifies those images as the target class. Note that the trigger only affects a small number of pixels in one part of the image, making it a subtle modification.

The following section discusses experiments designed to demonstrate the image scaling attack on traffic sign data. The goal of these experiments is to train a model on data that introduces a backdoor, which can be exploited using raw test images. Because of this, only training images have to be manipulated, but no images at test time. The training images are either globally or locally attacked by applying the image scaling attack before they are fed into the model, which is then trained on this poisoned data. This makes it possible to demonstrate the attack in a real-world setting, as the effect of the backdoor can be replicated on raw test images.

3.1. Scaling Method

The attacked scaling method has to match the one that is used by the model to scale the size of the input images down to the desired input size. The nearest-neighbor method is the simplest and least expensive [26]. It copies pixels at a regular distance from the previous one to the smaller image. The method is fast to compute since all pixels in the downscaled image match a pixel in the larger image, but the results tend to look pixelated, especially if the scaling ratio is large [26].

Other, more complicated methods can provide smoother results, like the bilinear method that interpolates over a section of pixels and smooths over pixels in the downscaled image [26]. This method is computationally more expensive to attack with the image scaling attack since an attacker would need to calculate for every single image exactly which pixel section produces which results and how to manipulate them so their averages result in a different image.

In contrast, if the nearest-neighbor method is applied from one fixed input size to one fixed output size, the pixels that are focused on for downscaling are always in the same position, making it extremely easy to manipulate only those pixels in linear time $\mathcal{O}(n)$. Since the image scaling attack is independent of the scaling method [2], this method provides results in the most efficient way.

As such, for this study, the nearest-neighbor method has been chosen to scale all images from 800×800 pixels to 64×64 pixels because this makes it feasible to demonstrate how the attacks work on a large dataset while being able to train the models in a reasonably short time. The scaling method is interchangeable with any other, but more complex scaling methods are computationally more expensive and slower [2].

3.2. Clean Data

The dataset used for training the model is the German Traffic Sign Recognition Benchmark Dataset, collected by researchers of the Ruhr University Bochum [27]. It presents a single-image, multi-class classification problem containing 43 classes of frequently observed German traffic signs in over $50,000$ images in total. In the training dataset, the same image is presented 30 times in varying resolutions. For this study, which has the goal of applying the image scaling attack on a larger image to reduce it to a differing smaller image, only two instances of each image are used in the training dataset, a size in the middle and the largest available size. That way, the model can handle lower resolutions of input data. Each image is scaled up to 800×800 pixels before the attack is applied, so the model receives images of the same size.

Data augmentation methods are applied to increase generalizability. The methods that are used include six rotations of all images at different angles up to 20 degrees in both directions; six shears of all images at different angles up to 30 degrees in both directions; ten cropped images taking away varying numbers of pixels at different sides of the image; four combinations of shear and rotations of up to 20 degrees in both directions; four images that vary in shade and lighting, and five instances where the images are resized to 0.8 times their original size and then placed

in the middle or the four corners, respectively, while the color of the newly added background is the average color from the border of the original image.

All of these data augmentations ensure that the integrity of the traffic sign itself is preserved while adding variation to the data that makes the model more versatile and fits the kind of data that will later be present in a real-world experiment to extract regions of interest, i.e., the traffic signs. All augmented images present a view on traffic signs that is close to what a real image of a traffic sign could actually look like. In total, the new augmented dataset includes $108,828$ images. In the last step, the augmented data is scaled down to a size of 64×64, which is the image size that is used as input to the model. The test dataset comprises $12,630$ unique images of varying resolutions to evaluate how successful the model is at classifying them.

For both the training and test datasets, the scaling algorithm used is the nearest-neighbor method from the Pillow library [28], which is computationally the least expensive and the fastest to apply the image scaling attack [26], [29]. For upscaling the training data to one consistent size of 800×800 pixels before giving it to the model, the Bilinear method is used because it produces smooth outputs [26].

Before training, the training data is randomly split into a training and a validation set, with the validation set containing 20% of the training data. The validation dataset is used during training to evaluate how well the model performs. All data is split into batches of 64 images, a value selected through tuning to balance memory efficiency and training speed given the available GPU resources.

3.3. Poisoned Data

To insert a backdoor into the model, the training data is manipulated to look differently after being scaled down from 800×800 pixels to the model input size of 64×64 pixels. The scaling ratio is 12.5, which is large enough for the target image to be successfully and imperceptibly hidden within the source image if the source image is not displayed at a too-large resolution.

3.3.1. Dirty-label Local Attack. Local changes to the training data are introduced in the form of a trigger inserted into the downscaled training images that only change a small part of the image. Since it has to be reproducible as a physical trigger in the raw test images later, this trigger was chosen to be a yellow rectangle or ellipse. The shape, color shade, size, and position of the rectangle are chosen randomly. The backdoor does not depend on those factors but learns to recognize different kinds of similar triggers to account for differences in perspective and lighting of physical triggers. The trigger is added to a varying percentage of the training images of any class, and the label of these poisoned instances is changed to the target class 7, which is the speed limit sign for 100 km/h and makes up 3.30% of the training data and 3.56% of the test data. Therefore, the presented attack is a targeted, dirty-label, source-agnostic poisoning attack.

3.3.2. Clean-label Global Attack. Global changes to the image are used in a targeted, clean-label, source-specific poisoning attack. For this attack, all images of the source label are changed such that they scale to images of the target label, whereas all images of the target label are manipulated to scale to images of the source label. Here, the entire image changes globally. In essence, all source and target images are imperceptibly switched while keeping their original label. The attack is clean-label, so no manipulation can be seen by inspecting the labels in the training data.

3.3.3. Clean-label Local Attack. The advantages of both previous attacks can be combined into a novel, imperceptible, and strong clean-label local attack. A local trigger is added in the form of a green rectangle to a percentage of training images of the target class only. The size, shade, and position of the trigger are static for this attack. That way, the model learns two associations with the target class: the clean images for the target class and the poisoned images of the target class that contain the trigger. The model then learns to associate the trigger only with the target class without the attacker needing to change any labels, making it nearly impossible to detect the changes made by the attack in the training data. This makes the attack targeted, clean-label, and source-specific.

3.4. Model Architecture

For object detection of traffic signs in larger images, the pre-trained YOLO-v8s architecture, which is highly suitable for multi-object detection [30], is fine-tuned on traffic sign data to learn to draw bounding boxes around relevant traffic signs. The regions of interest around the traffic sign are then extracted and classified individually [31]. The extracted region includes 10% more pixels on either side of the bounding box to account for the fact that the bounding boxes are tight around the traffic signs, which was noticed during the experiments.

The chosen classification model is a CNN with eight convolutional layers, each followed by a batch normalization layer and the ReLU activation function. After the second and the last batch normalization layer, a max pooling layer is added, respectively. The last layers are a flattening layer since color images are represented as matrices of size $3 \times 64 \times 64$, but the final result should be a label prediction, followed by a dropout layer, a linear layer with a ReLU activation function, and a last linear layer with 43 outputs to match the number of classes.

This model architecture is tested in a series of experiments and performs best on clean data. In the experiments, the number of convolutional layers is varied between 5 and 12, and LeakyReLU and tanh are tested as alternative activation functions. Additionally, the pre-trained models ResNet50 and VGG19 are fine-tuned by training on the traffic-sign data for 20 epochs to compare performances. The best model is then used further for training on the poisoned data. For the experiments on the poisoned model, the amount of poisoned data is varied in 5 experiments between 5% and 70%.

3.5. Model Evaluation Methods

The performance of the trained model is evaluated on two data sets: one with only clean data and one where the trigger is added to every image. The clean model should

perform well on the clean test data but has not learned to associate the trigger with a target class, so it should also classify those test examples into the class they originally belonged to. The poisoned model should perform well on both test sets; it should be able to classify all triggered examples into the target class, while all clean examples should be assigned their original class. For evaluation, the attack success rate, backdoor accuracy, and clean accuracy degradation are used to measure the success of an attack.

3.6. Real-world Testing Method

To test the model on real-world data, a physical trigger of a similar shape and color as the trigger has to be added to real traffic signs, for example, in the form of a sticker. Then, a photo of that traffic sign in which the physical trigger can be seen is given to the model to evaluate if the backdoor can be triggered in this way.

3.7. Experimental Setup

The experiments were conducted on an NVIDIA RTX A5000 GPU using Python 3.9, Pytorch 2.3.1, and Cuda 12.2.

4. Experimental Results

4.1. Baseline Model Training on Clean Data

First, a model is trained on the clean dataset as a baseline for evaluating the subsequent poisoned models. As shown in Table 1, the highest accuracy of 97.14% is achieved by a CNN with 8 convolutional layers and a ReLU activation function. The clean model is still accurate in 89.65% of cases if a trigger object blocks part of the view on the images. No association of that trigger object with the target class can be observed. The backdoor accuracy shows that 3.31% of images are classified as the target class if the trigger is present, which is approximately the ratio of target images in the test dataset, and thus the correct label with a small margin of error. Without being trained to recognize the trigger and to show a specific behavior in its presence, the model ignores the trigger and instead classifies the actual traffic signs correctly.

The model is not just overall accurate but accurate for every individual class, as can be seen in Figure 1. No class is more likely to be misclassified than another class. The distribution of predictions follows the correct distribution of samples per class in the test set. The model's confusion matrix demonstrates that it is accurate at determining the correct class, with only very few instances located away from the diagonal. On clean data, the model has a precision of 96.02%, recall of 96.79%, and specificity of 100%. Thus, this model architecture is suitable as a baseline for further experiments on poisoned data.

4.2. Dirty-label Local Attack

For the first attack, a model is trained on data to which a small trigger object in the form of a collection of yellow pixels has been added. The trigger is added to images of any source class, and the label of the manipulated images is then changed to a target class.

Adding the local trigger to images occurs imperceptibly. It only becomes visible after the training images are scaled to the model input size of 64×64 but is hidden in the original images of size 800×800 by means of the image scaling attack. After scaling, the trigger is revealed in the form of a yellow cluster of pixels. The trigger is demonstrably not a single unchanging object but varies between iterations. This resembles a physical trigger for which the look slightly changes depending on lighting, perspective, and placement, and this way, the model is being trained to still recognize the trigger under these varying circumstances. The scaling only results in the image having smaller dimensions and the trigger appearing;

TABLE 1: *List of experiments on the clean baseline model. The experiments compare the model architecture, the activation function, the poisoning rate ϵ, the accuracy on clean test data, the accuracy on poisoned test data where the labels of all poisoned data have been changed to the target label (the higher this accuracy is, the better the model has learned to associate the trigger with the target class), and the accuracy on poisoned test data where the labels of poisoned data remain the original labels (the higher the accuracy is, the better the model has learned to ignore the trigger object in favor of classifying the traffic sign correctly).*

Model architecture	Activation	Poisoning rate ϵ	Clean acc	Poisoned acc	Poisoned acc w/ true labels
CNN, 5 Conv layers	ReLU	0	0.9578	0.0380	0.8858
CNN, 6 Conv layers	ReLU	0	0.9630	0.0401	0.8961
CNN, 8 Conv layers	**ReLU**	**0**	**0.9714**	**0.0331**	**0.8965**
CNN, 8 Conv layers	LeakyReLU	0	0.9513	0.0550	0.8423
CNN, 8 Conv layers	tanh	0	0.9284	0.0346	0.8663
CNN, 12 Conv layers	ReLU	0	0.9597	0.0368	0.8809
VGG19	ReLU	0	0.9245	0.0298	0.7611
ResNet50	ReLU	0	0.9651	0.0279	0.8472

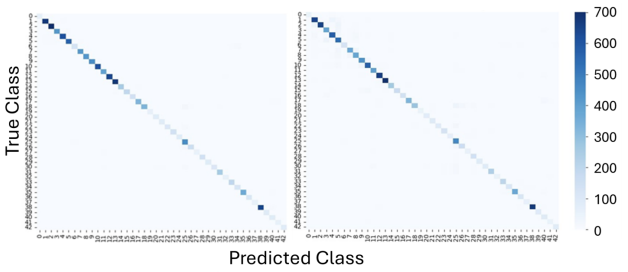

Figure 1: *Confusion matrices highlighting predicted versus correct classes of the clean model on clean data (left) and poisoned data (right). The distributions clearly show that the predictions are highly accurate for every class, independent of the presence of a trigger.*

the rest of the image stays exactly the same, as this is only a local change.

This makes the detection of this local trigger attack harder compared to a global attack because only a very small part of the image is changed while most remains the same as in the larger image. As shown in Table 2, after training the model for 20 epochs on a locally poisoned dataset in which between 5% and 70% of data is poisoned, the poisoned model achieves high testing accuracies on both poisoned and clean test data. For only 5% poisoned data, the poisoned test data with the local trigger is classified as the target class in 99.03% of cases. The clean test data without a trigger is classified as the correct class in 97.08% of test cases by the model, which is barely lower than the accuracy of the clean model. Similarly, the poisoned model has a precision of 96.58%, recall of 96.40%, and specificity of 100% on the clean data, showcasing that it does not perform worse than the clean model but learns the association of the trigger in addition to learning the correct classes. The confusion matrix of the poisoned model on clean data can be found in Figure 2 and resembles the confusion matrix of the clean model. Thus, the model has successfully learned to recognize the local trigger and associate it with the target class while preserving its performance on clean data. This behavior can be reproduced with a physical trigger of a similar shape, size, and color scheme on traffic signs. Such a traffic sign with a physical trigger, for example, in the form of a sticker, is then also misclassified as the target class.

If the amount of locally poisoned data is increased, the attack success rate also increases while the clean data accuracy declines, especially once there is more poisoned data than clean data in the dataset. However, even when 70% of the data is poisoned, the clean accuracy remains above 90%, which is still exceptionally high and shows that the model performs well on clean and poisoned data. From this, it can be concluded that even a very small sample of poisoned training data suffices to make the model learn to associate the trigger with the target class and show the behavior associated with the backdoor. Still, the performance on clean data is preserved even if the percentage of poisoned data increases drastically. The relation between accuracy and percentage of poisoning in the model is depicted in Figure 3.

TABLE 2: *List of experiments on the local trigger dirty-label image scaling attack. The experiments compare the model architecture, the activation function, the poisoning rate ϵ, the accuracy on clean test data, the accuracy on poisoned test data where the labels of all poisoned data have been changed to the target label (the data that is used for training; the higher this accuracy is, the better the model has learned to associate the trigger with the target class), and the accuracy on poisoned test data where the labels of poisoned data remain the original labels (the higher this accuracy is, the better the model has learned to ignore the trigger object in favor of classifying the traffic sign correctly).*

Model architecture	Activation	Poisoning rate ϵ	Clean acc	Poisoned acc	Poisoned acc w/ true labels
CNN, 8 Conv layers	ReLU	0	0.9714	0.0331	0.8965
CNN, 8 Conv layers	ReLU	0.05	0.9708	0.9903	0.0431
CNN, 8 Conv layers	ReLU	0.1	0.9514	0.9975	0.0390
CNN, 8 Conv layers	ReLU	0.2	0.9628	0.9964	0.0383
CNN, 8 Conv layers	ReLU	0.5	0.9478	0.9995	0.0369
CNN, 8 Conv layers	ReLU	0.7	0.9232	0.9992	0.0364

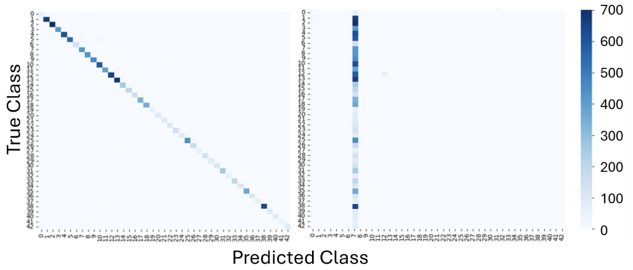

Figure 2: *Confusion matrices highlighting predicted versus correct classes of the locally poisoned model ($\epsilon = 5\%$) on clean data (left) and poisoned data (right). The distributions clearly show that the predictions are highly accurate for every class on clean data. Still, if the trigger is present, the poisoned model classifies the image as the target class with high confidence for every class. The only class for which the model is slightly less confident is class 12, which is the priority road sign that is yellow, just like the trigger, which explains why sometimes the trigger cannot be properly distinguished from the sign itself in this case.*

4.3. Clean-label Global Attack

In this attack, only images of two classes are manipulated by the image scaling attack such that images of the first class scale down to images of the second class and images of the second class scale down to images of the first class. Because of this, no label manipulation is required since the original label already mismatches with the switched image that the model receives, causing the model to learn a wrong label association for these two classes. For the switch to work, all instances of both classes need to be switched; otherwise, the model cannot learn to associate the original label with the switched image reliably.

The switch of the source and target images after scaling is imperceptible to the human eye. Once the images are scaled down from 800×800 to 64×64, images of the target class switch to images of the source class and vice versa. That way, the images are globally different after scaling. The attack is clean-label and, therefore, cannot be detected by comparing training images and their labels; for the previous attack, a mismatch could be detected with this method. However, this also means that since the model is not trained on a trigger but instead to confound two classes, it never performs well for those classes, even on clean data that is scaled correctly, since it learns to associate those classes with the incorrect labels only.

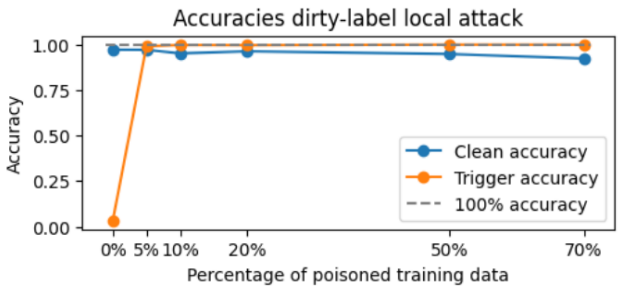

Figure 3: *Model performance on clean (blue curve) and poisoned (orange curve) data correlation with the amount of poisoned data in the training data.*

TABLE 3: *List of experiments on the global switch clean-label image scaling attack. The experiments compare the model used for testing, the accuracy on clean test data where the labels of all classes are correct, the accuracy on poisoned test data where class 12 is labeled as class 13 and vice versa (the higher this accuracy is, the better the model has learned to switch the classes), the accuracy of label 12 (how well the model correctly classifies an image of class 12 as that class), the accuracy of label 13 (how well the model classifies an image of class 13 as that class), the attack success rate for label 12 (how well the model has learned to classify images of class 12 as class 13), and the attack success rate for label 13 (how well the model has learned to classify images of class 13 as class 12).*

Model	Clean acc	Poisoned acc
Clean baseline	0.9714	0.0087
Global switch	0.8494	0.9583

Acc label 12	Acc label 13	ASR label 12	ASR label 13
0.9944	0.9778	0.010	0.000
0.003	0.000	0.9841	0.9958

The attack is highly successful at switching the labels of classes 12 and 13, which can be seen in the confusion matrix in Figure 4. In the clean model, instances of these two classes are mistaken for each other in less than 1% of cases. Both classes show an extremely high individual accuracy in the clean model.

In the poisoned model (Table 3), the clean accuracy drops drastically to 84.94% because the two switched classes cannot be classified correctly. When testing on poisoned data for which the labels are switched to determine the attack success rate, the accuracy is much higher, indicating that the model successfully learned the switch. For both switched labels, the attack success rate is high: misclassifying class 12 as class 13 succeeds in 98.41% of cases, and misclassifying class 13 as class 12 succeeds in 99.58% of instances. The real accuracy of both classes to be sorted in the correct class is close to 0.

4.4. Clean-label Local Attack

The clean-label local attack is novel and unique in that it combines the two main advantages of the previous attacks into one. Just like for the dirty-label local attack,

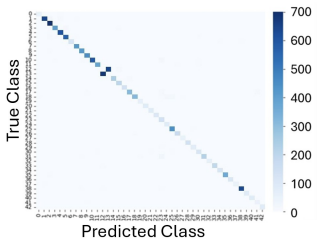

Figure 4: *Confusion matrix highlighting predicted versus correct classes of the globally poisoned model on clean data. The distributions clearly show how the only classes that are affected are the two switched classes, while all others are classified correctly with high confidence. For the two switched classes, the model is confident about the label of the other class belonging to any given image.*

only a small trigger is inserted into the image through the image scaling attack, while the largest part of the image remains unchanged after scaling. At the same time, just like for the clean-label global attack, no labels are manipulated. This clean-label local attack is source-specific because, at training time, it is only applied to a fraction of images from the target class. Furthermore, the trigger is added to images of any other class only at test time, which are then classified as the target class because of the presence of the trigger.

As for the dirty-label local attack, adding the trigger for this clean-label attack is imperceptible because of the image scaling attack method. After scaling from 800×800 pixels to 64×64 pixels, the green rectangle appears in a percentage of images that belong to the target class but not in images belonging to any other class. Thus, the model learns to associate the trigger with the target class without having to switch labels. Then, at test time, the model is presented with images of any class that contains the trigger. Because of this, the detection of the attack is harder compared to the dirty-label attack since there are no discrepancies between the label and the visible traffic sign.

The attack is highly successful if the amount of poisoned data is large enough, as shown in Table 4. Since only images of the target class are poisoned, the poisoning rate only describes the percentage of images of that one class that are poisoned. If more than 30% of target images are poisoned, which is about 0.99% of the whole training dataset, the attack success rate is higher than 90%, making the attack succeed in almost all cases; this can be seen in Figure 5. Interestingly, further increasing the poisoning rate does not cause the clean accuracy to drop significantly, and the accuracy of detecting the target class itself without a trigger stays consistently high, which makes sense since the target class is never associated with images of any other class. The relation between accuracy and percentage of poisoning of the target class in the model is depicted in Figure 6.

5. Real-world Experiment with Physical Triggers

For the training and test data from the previous experiments, all trigger objects were generated with a random

TABLE 4: *List of experiments on the local trigger clean-label image scaling attack. The experiments compare the model architecture, the activation function, the poisoning rate ϵ of the target class, the accuracy on clean test data, the accuracy on poisoned test data where the labels of all poisoned data have been changed to the target label (labels are changed for test purposes; the higher this accuracy is, the better the model has learned to associate the trigger with the target class), and the accuracy on poisoned test data where the labels of poisoned data remain the original labels (the higher this accuracy is, the better the model has learned to ignore the trigger object in favor of classifying the traffic sign correctly).*

Model architecture	Activation	Poisoning rate ϵ	Clean acc	Poisoned acc	Poisoned acc w/ true labels
CNN, 8 Conv layers	ReLU	0	0.9714	0.0331	0.8965
CNN, 8 Conv layers	ReLU	0.05	0.9647	0.2285	0.6292
CNN, 8 Conv layers	ReLU	0.1	0.9650	0.7926	0.1774
CNN, 8 Conv layers	ReLU	0.3	0.9566	0.9346	0.0462
CNN, 8 Conv layers	ReLU	0.5	0.9703	0.9981	0.0358
CNN, 8 Conv layers	ReLU	0.7	0.9655	0.9986	0.0364

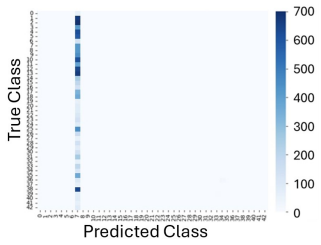

Figure 5: *Confusion matrix highlighting predicted versus correct classes of the clean-label locally poisoned model ($\epsilon = 50\%$) on poisoned data. The distribution clearly shows that if the trigger is present, the poisoned model classifies the image as the target class with high confidence for every class despite never having explicitly learned so. The model has learned the pattern that in the training data, the trigger was only presented in images of the target class, so it continues to classify images of all classes as the target class whenever the trigger is present.*

shade, shape, and location by the computer and added on top of the image. Because the trigger object has been chosen for its simplicity, it is possible to replicate it with a physical object that is added to real-world traffic signs in the form of, for example, a sticker.

In a real-world deployment, the model can either be presented with clean traffic signs or traffic signs onto which the physical trigger object has been stuck. After combining the trained YOLOv8 model to locate traffic signs and the CNN to classify them, for example, using the poisoned model with poisoning rate $\epsilon = 5\%$ from the dirty-label local attack, trained to associate a yellow trigger with the target class 7, the class representing the

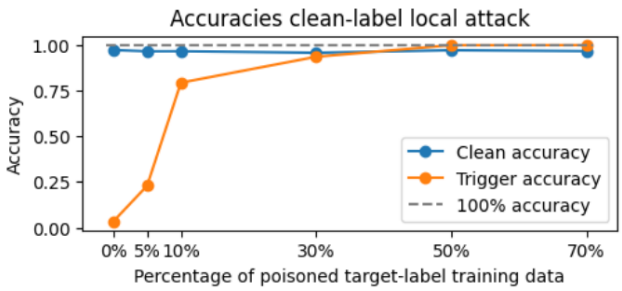

Figure 6: *Graph showcasing how the model performance on clean (blue curve) and poisoned (orange curve) data correlates with the amount of poisoned target-label data in the training data.*

100 km/h speed limit sign, into a pipeline, they can be presented with test images of traffic signs to classify them. For the local attacks, the trigger is now replaced by a physical object.

If the models are presented with clean traffic signs, they are classified correctly with high accuracy. If a physical object of the wrong color, here pink or green, is present on a traffic sign, then they are still classified correctly by the models. However, if the yellow trigger is added to the traffic sign in the form of a physical sticker, the model classifies the traffic sign into the target class with high confidence. The physical trigger thus replaces the pixel-based trigger drawn by the computer on which the model was trained. The physical trigger reliably produces misclassifications under varying lighting conditions, camera angles, and distances.

This makes the generalizability of the trigger significant, as it can be replaced by an object that is inherently different from what it was trained on. A physical sticker placed at a random position is presented to the models under lighting and perspective that an attacker has no influence on, yet the model still associates the physical trigger with the same properties as the trigger added by the computer. The replacement with a physical trigger also works if the model has been attacked by the novel clean-label local attack that adds the green square as a trigger. The results are more difficult to replicate because the shade of the green trigger needs to be close to the trained one, and the position has to be approximately where it was trained.

This has real-world consequences; it is a common occurrence that colorful stickers are stuck to traffic signs in the real world, and this proves that malicious actors can manipulate training data in a way that these random stickers can have a significant influence on the classification abilities of a model that is deployed in real traffic, for example, in autonomous driving. Additionally, it is hard to detect which kind of trigger takes on the role of a backdoor to cause malicious behavior in the model; as shown here, a trigger of the wrong color has no effect on the model's performance.

6. Frequency Analysis Defense

Since the attacks use the nearest-neighbor method for scaling, detection via frequency analysis works well. The magnitude spectrum can effectively show the difference

between original and attacked images but not between scaled-down versions of the original and attacked images. If an image has been manipulated by the image scaling attack, then the frequency spectrum reveals irregularities that are not present in unmanipulated images, as seen in Figure 7. However, the correlation is not proof of manipulation. Frequency analysis shows even stronger irregularities for images scaled down to the size the model takes as input, here 64×64 pixels. This behavior is not random: an image that is scaled down shows very similar irregularities on the frequency spectrum, even stronger if the nearest-neighbor method was used for downscaling because it creates more pixelated results than other, smoother methods.

Thus, if the image scaling attack was applied to an image, the frequency analysis reveals irregularities in the pattern, but the presence of such irregularities is not necessarily caused by the image scaling attack. The frequency analysis is a good indicator of an attack happening. Still, it is not infallible because it can only show irregularities in the frequency spectrum but not what caused them. Thus, images on which downscaling has already been applied or that are of a lower quality show similar patterns to images that have been attacked, as seen in Figures 8 and 9, which causes the frequency analysis method not to be completely reliable in detecting the image scaling attack.

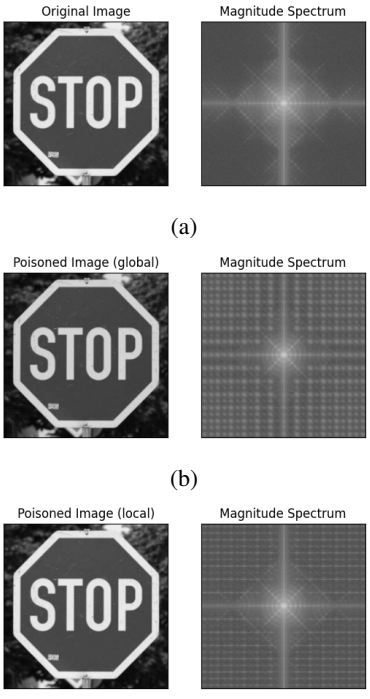

Figure 7: *Comparison of frequency signatures. a) Frequency spectrum of the original unmanipulated image. b) Frequency spectrum of the image after a global attack. c) Frequency of the image after a local attack. No attack is visible in the images themselves. The global attack is more obvious in the spectral graph, but the local attack also shows irregularities.*

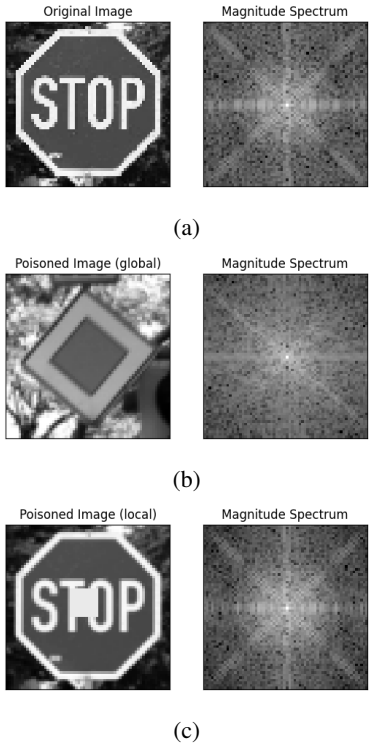

Figure 8: *Comparison of frequency signatures after scaling to* 64×64 *pixels. a) Frequency spectrum of the original unmanipulated image. b) Frequency spectrum of the image after a global attack becomes visible. c) Frequency of the image after a local attack becomes visible. All images show large irregularities in the frequency signatures despite no attacks being hidden inside the images anymore.*

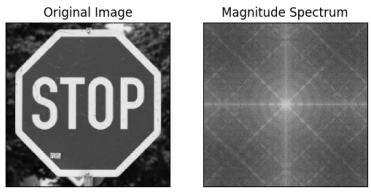

Figure 9: *Scaling the original image to half its original size causes irregularities to appear in the frequency spectrum, even though no attack has been applied.*

7. Universal Compatibility

The Image Scaling Attack is extremely versatile and can be combined with almost any other backdoor and evasion attack. The attacks above demonstrate how a trigger can be added via image scaling, with the new addition of replicating the trigger with a physical trigger during test time, as a clean-label and dirty-label attack. The attack works with different types of triggers, including complex shapes. Additionally, the clean-label global attack shows that image scaling can be utilized to completely change images and switch two classes, as demonstrated.

The image scaling attack is independent of the actual attack that is applied. It is universally compatible with other attacks since the process is always the same and independent of the backdoor. Image scaling takes two images and hides one of them inside of the other one,

such that an image that is as similar to the target image as possible becomes visible after scaling, while only the attack image, which is as similar as possible to the source image, is visible before. This target image can be an image poisoned by any attack, either at training time for backdoor attacks or at test time for evasion attacks. In both cases, a human observer only sees the attack image that is almost indistinguishable from the original image, while the model receives the scaled target version that contains the trigger at training time or the perturbation at test time. Since the image scaling attack only defines the model behavior is maliciously manipulated due to an image changing after scaling, each of these possible attacks fulfills that requirement, and each can be combined with the image scaling attack. The image scaling attack is, thus, an umbrella attack that defines many possible attack specifics that are being hidden via the same method but can be applied independently and interchangeably.

The following experiments show a selection of image scaling attacks and demonstrate that the attack success rate stays consistently high. The presented backdoor attacks show similarity to the BadNets attack [12], where a small trigger is added as a backdoor to images so a model learns to associate it with a malicious behavior. Here, the attack has been adapted to work both with and without needing to manipulate the labels as well, in the dirty-label and clean-label versions of the attack, respectively, and it has been shown that the trigger can be replaced by a physical trigger at test time. The dirty-label local attack has an attack success rate of 99.03% for just 5% poisoned data, and the clean-label attack has an attack success rate of 93.46% if 30% of the target class data is poisoned. The trigger itself can be adapted to take any form, even that of a complex image. Similarly, the WaNet attack [14] can be used to apply an almost imperceptible warping to the target image. Applying this as an image scaling attack does not change the mechanics of the known attack itself but simply adds one attack step beforehand, during which the attacked image is hidden in a larger image that is scaled down to reveal the warping effect when the image is given to the model. This attack has a success rate of 96.09% for 10% poisoned data. The Blend attack [8] layers a blended image above the original image and switches the label to a target class for the images that contain the blended trigger. When the model is trained on data of which 10% is poisoned in this manner, the attack success rate is 99.81%.

8. Discussion and Limitations

As shown, the presented image scaling attacks are highly successful and can be replicated by a physical trigger. We also demonstrated how the image scaling attack is independent of the actual attack since it is an umbrella method to hide an attack on a smaller image inside of a larger image, making it universally compatible with other backdoor and evasion attacks.

The presented local attacks have some clear limitations: they utilize they utilize a simple backdooring technique with a relatively obvious trigger that is obvious to see in the scaled-down version. Still, they demonstrate how versatile image scaling attacks are because they can also be combined with many other, less perceptible backdoor or evasion attacks and hide those successfully. Two of the detailed attacks have the advantage of being clean-label, which makes them even harder to detect. For defense, the universal compatibility of the image scaling attack makes it harder to prevent and detect because of the variety of what manipulation can look like.

The image scaling attacks depend on the scaling algorithm that is used but can theoretically be adapted to any other scaling algorithm. Since calculating the right manipulations to the correct pixels is computationally expensive for more complex algorithms that use clusters of pixels, the demonstrations here focus on the simpler nearest-neighbor algorithm where only specific pixels are selected during scaling, and all other pixels are ignored.

For this study, the specific version of a local backdoor attack with a trigger was chosen because the trigger can be replicated by a physical trigger at test time, showing the significant generalization capabilities of the model. This type of trigger resembles a real phenomenon in which traffic signs are often decorated with stickers of various shapes or colors. Therefore, poisoning a model to react to these kinds of manipulations is highly relevant in the real world, especially since it has been demonstrated that the data manipulation of adding the trigger imperceptibly via the image scaling attack can be done completely at the computer while the model still reacts to the presence of a physical trigger. Furthermore, raising awareness of this issue is essential because while the model changes its behavior if the backdoor trigger is present, its accuracy on clean data and data for which a trigger that is different from the trained backdoor is inserted is very high. If the backdoor trigger is unknown, it is hard to find the one specific trigger that causes malicious behavior.

9. Conclusions

This paper investigates vulnerabilities in automatic traffic sign classification through the implementation of an image scaling attack. Several experiments were conducted that demonstrate that the attack can be applied dirty-label or clean-label and in a global or local manner. All tested versions of the attack are highly successful in fooling the model. The novel clean-label local attack combines the advantages of previous image scaling attacks into a highly successful and difficult-to-detect attack version. Indeed, only a small part of the original image changes after scaling, no manipulation of the label is necessary, and only a fraction of images of one class, the target class, have to be manipulated while all other images remain clean and unchanged.

Additionally, the image scaling attack has been shown to be universally compatible with other backdoor and evasion attacks because it functions as an umbrella method to hide an attack.

The attack can be deployed at test time by replacing the previously computer-added trigger with a similar physical trigger in the form of a sticker. This contributes to the real-world vulnerabilities of the image scaling attack in the context of traffic sign classification and showcases how crucial the security of AI models deployed in high-risk situations like traffic is.

References

[1] K. Aslansefat, S. Kabir, A. Abdullatif, V. Vasudevan, and Y. Papadopoulos, "Toward improving confidence in autonomous vehicle software: A study on traffic sign recognition systems," *Computer*, vol. 54, no. 8, pp. 66–76, 2021.

[2] E. Quiring and K. Rieck, "Backdooring and poisoning neural networks with image-scaling attacks," in *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2020, pp. 41–47.

[3] E. Quiring, A. Müller, and K. Rieck, "On the detection of image-scaling attacks in machine learning," in *Proceedings of the 39th Annual Computer Security Applications Conference*, 2023, pp. 506–520.

[4] Q. Xiao, Y. Chen, C. Shen, Y. Chen, and K. Li, "Seeing is not believing: Camouflage attacks on image scaling algorithms," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 443–460.

[5] Y. Gao, I. Shumailov, and K. Fawaz, "Rethinking image-scaling attacks: The interplay between vulnerabilities in machine learning systems," in *International Conference on Machine Learning*. PMLR, 2022, pp. 7102–7121.

[6] E. Quiring, D. Klein, D. Arp, M. Johns, and K. Rieck, "Adversarial preprocessing: Understanding and preventing {Image-Scaling} attacks in machine learning," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 1363–1380.

[7] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*. Springer, 2013, pp. 387–402.

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

[9] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 900–14 912, 2021.

[10] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson, and T. Goldstein, "Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9389–9398.

[11] W. You and D. Lowd, "The ultimate cookbook for invisible poison: Crafting subtle clean-label text backdoors with style attributes," in *The Third Workshop on New Frontiers in Adversarial Machine Learning*, 2024.

[12] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *unpublished*, 2017.

[13] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019.

[14] T. A. Nguyen and A. T. Tran, "Wanet-imperceptible warping-based backdoor attack," in *International Conference on Learning Representations*, 2020.

[15] E. Wenger, J. Passananti, A. N. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, "Backdoor attacks against deep learning systems in the physical world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6206–6215.

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

[17] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 966–11 976.

[18] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 463–16 472.

[19] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*. Springer, 2020, pp. 182–199.

[20] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, "Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 321–338.

[21] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[22] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *unpublished*, 2014.

[23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[24] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.

[25] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[26] P. Parsania, P. V. Virparia *et al.*, "A review: Image interpolation techniques for image scaling," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 2, no. 12, pp. 7409–7414, 2014.

[27] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The german traffic sign recognition benchmark: a multi-class classification competition," in *The 2011 International Joint Conference on Neural Networks*. IEEE, 2011, pp. 1453–1460.

[28] A. Clark *et al.*, "Pillow (pil fork) documentation," *readthedocs*, 2015.

[29] N. Jiang and L. Wang, "Quantum image scaling using nearest neighbor interpolation," *Quantum Information Processing*, vol. 14, pp. 1559–1571, 2015.

[30] R. Chandana and A. Ramachandra, "Real time object detection system with yolo and cnn models: A review," *unpublished*, vol. 773, 2022.

[31] J. Terven, D.-M. Córdova-Esparza, and J. A. R. Gonzalez, "A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, p. 1680–1716, 2023.