

HOSTED BY



ELSEVIER

Contents lists available at ScienceDirect

Journal of King Saud University - Computer and Information Sciences

journal homepage: www.sciencedirect.com

Full Length Article

RAPID: Robust multi-patch masker using channel-wise Pooled variance with two-stage patch Detection

Heemin Kim^a, Byeong-Chan Kim^a, Sumi Lee^a, Minjung Kang^a, Hyunjee Nam^a,
Sunghwan Park^b, Il-Youp Kwak^a, Jaewoo Lee^{b,c,*}

^a Department of Statistics and Data Science, Chung-Ang University, Seoul, 06974, South Korea

^b Department of Security Convergence Science, Chung-Ang University, Seoul, 06974, South Korea

^c Department of Industrial Security, Chung-Ang University, Seoul, 06974, South Korea

ARTICLE INFO

Keywords:

Adversarial attack
Adversarial patch defense
Pre-processing defense
Object detection
Robust detection

ABSTRACT

Recently, adversarial patches have become frequently used in adversarial attacks in real-world settings, evolving into various shapes and numbers. However, existing defense methods often exhibit limitations in addressing specific attacks, datasets, or conditions. This underscores the demand for versatile and robust defenses capable of operating across diverse scenarios. In this paper, we propose the RAPID (Robust multi-patch masker using channel-wise Pooled variance with two-stage patch Detection) framework, a stable solution to restore detection efficacy in the presence of multiple patches. The RAPID framework excels in defending against attacks regardless of patch number or shape, offering a versatile defense adaptable to diverse adversarial scenarios. RAPID employs a two-stage strategy to identify and mask coordinates associated with patch attacks. In the first stage, we propose the 'channel-wise pooled variance' to detect candidate patch regions. In the second step, upon detecting these regions, we identify dense areas as patches and mask them accordingly. This framework easily integrates into the preprocessing stage of any object detection model due to its independent structure, requiring no modifications to the model itself. Evaluation indicates that RAPID enhances robustness by up to 60% compared to other defenses. RAPID achieves mAP50 and mAP@50-95 values of 0.696 and 0.479, respectively.

1. Introduction

The recent emergence of numerous adversarial attacks has demonstrated the vulnerabilities of deep neural networks and exposed the threat of real-world implementations of these vulnerabilities (Kurakin et al., 2018). Adversarial attacks are designed to cause deep learning models to malfunction for a given image, and can be categorized into perturbation attacks (L1, L2, or L_∞ norm-based attacks) and patch attacks depending on the range of noise added. Perturbation attacks (Madry et al., 2017; Dong et al., 2018; Carlini and Wagner, 2017; Moosavi-Dezfooli et al., 2016) introduce imperceptible noise to the entire image, whereas patch attacks (Brown et al., 2017; Karmon et al., 2018; Liu et al., 2018; Lee and Kolter, 2019; Thys et al., 2019)

achieve attack objectives by adding more noticeable noise within a restricted range. While perturbation attacks demonstrate high efficacy, their requirement for access to the entire image renders them impractical in real-world execution. Consequently, to enable real-world attacks, visibly disruptive patch-based attacks targeting specific regions of interest have emerged (Kurakin et al., 2018). Patch attacks typically manifest in sticker-like forms applied to target, with increased effectiveness proportional to the number of attack patches, exacerbating model malfunction (Rossolini et al., 2023; Oonishi et al., 2022). In Fig. 1, the image in the top left corner subjected to adversarial patch attacks reveals that, after applying the object detection model, the resulting image in the top right corner fails to detect cars or traffic signs. Instead,

* Correspondence to: 84 Heukseok-ro, Dongjak-gu, Seoul, 06974, Republic of Korea.

E-mail addresses: heemink@cau.ac.kr (H. Kim), moch1996@cau.ac.kr (B.-C. Kim), dltal821@cau.ac.kr (S. Lee), tg00383@cau.ac.kr (M. Kang), namhj1004@cau.ac.kr (H. Nam), tjdgkhs994@cau.ac.kr (S. Park), ikwak2@cau.ac.kr (I.-Y. Kwak), jaewoolee@cau.ac.kr (J. Lee).

Peer review under responsibility of King Saud University.



Production and hosting by Elsevier

<https://doi.org/10.1016/j.jksuci.2024.102188>

Received 21 May 2024; Received in revised form 17 August 2024; Accepted 8 September 2024

Available online 11 September 2024

1319-1578/© 2024 The Authors. Published by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

it focuses solely on detecting patch areas or misidentifies them as a person or handbags.

Adversarial attacks are not confined to a single domain and can impact various areas, including autonomous driving and face detection. For instance, in autonomous driving systems, these attacks can mislead object recognition algorithms, creating significant road safety hazards. Similarly, in face detection systems, adversarial strategies may facilitate identity theft crimes. Considering these potential threats, it is essential to refer to fundamental research such as Ren et al. (2015), Redmon et al. (2016) and Teichmann et al. (2018) for autonomous systems, and Ali et al. (2019b,a) for face detection systems. These studies provide crucial insights for developing effective countermeasures against adversarial strategies. Therefore, given the widespread threat of adversarial attacks across different domains, it is imperative to research and develop defense strategies to strengthen models against these tactics.

Various defense methodologies have been proposed and the phases in which they are applied are pre-processing, in-processing, and post-processing. We will explain the defense techniques applied in the pre-processing and in-processing phases. First, defense techniques applied in the pre-processing stage (Hayes, 2018; Naseer et al., 2019; Liu et al., 2022; Chou et al., 2020; Raju and Lipasti, 2020) include locating and masking patches or smoothing pixel values to reduce the impact of hostile attacks. Second, defense techniques applied in the in-processing phase include adversarial training, architecture modification, and certified robustness defenses. Adversarial training (Metzen et al., 2021; Wu et al., 2019) enhances model robustness by using adversarial examples in the training data, while architecture modification (Xiang et al., 2021; Ji et al., 2021) involves narrowing the receptive field or adding new units. There are also methods that propose certified defenses rather than empirical defenses to prove that the model can defend against the worst-case scenario (Metzen and Yatsura, 2021; Zhang et al., 2020; Xiang et al., 2023). The methodology of this paper corresponds to the defense technique applied in the pre-processing stage among the aforementioned methodologies. Operating at the pre-model stage, it proactively suppresses the adversarial nature of attacking patches before they impact the model by functioning before the input values enter the model.

We propose Robust multi-patch Masker using Channel-wise Pooled variance with two-stage patch Detection (RAPID).¹ The adversarial patches considered in this paper exhibit strong adversariality with discontinuous patterns (Liu et al., 2018; Lee and Kolter, 2019), and RAPID is based on two observed characteristics of these patches. These characteristics are as follows: Firstly, significant variations exist between neighboring pixels within adversarial patches. All pixels within adversarial patches are contiguous, displaying a pattern of continuous disturbance. There is a trade-off between the variations among patch pixels and the effectiveness of the attack. To create a robust attack, large perturbations are required without constraints, resulting in adversarial patches exhibiting highly visible irregular patterns. Secondly, adversarial patches exhibit high density. During patch creation, random noise is initially applied to pixel values within specific regions of the image. Subsequently, pixel values are updated as learning progresses. Consequently, adversarial patches become characterized by significant perturbations, inducing disproportionately large influences on the output to dominate predictions for all inputs. In essence, localized adversarial examples are generated through learning on specified regions of pixel values. These two characteristics of patches lead to ideas that can neutralize conventional adversarial patch attacks.

RAPID employs a two-stage defense mechanism, with the first stage being Patch Localization. Leveraging the fact that adversarial patches exhibit higher pixel variations compared to regions without patches due to significant differences between pixel values, RAPID calculates the variance in each channel and proposes Channel-wise Pooled Variance

to distinguish between areas with and without attached patches in the image. Since adversarial attacks typically damage only localized regions of the input while leaving the majority undisturbed, significant discrepancies are also observed in the channel-wise pooled variance. The second stage is Patch Defense, based on the high density of patches. Pixels with high channel-wise pooled variance are densely clustered and maintain specific shapes. By utilizing density-based clustering technique called DBSCAN (Ester et al., 1996), RAPID determines the patch locations and masks them, thus suppressing patch attacks. DBSCAN, as a density-based clustering method, efficiently adapts to adversarial attack scenarios regardless of the shape of the adversarial patches, making it an effective algorithm for addressing adversarial attacks. As illustrated in Fig. 1, RAPID effectively detects and masks the regions attacked by patches, thereby neutralizing the attack (the bottom right image), while the object detection model accurately identifies the objects.

We summarize our contribution as follows:

- We propose a novel attack detection method, called RAPID, utilizing Channel-wise Pooled Variance and DBSCAN clustering, enabling fast but effective defense against physical attacks.
- It demonstrates high adversarial attack defense robustness regardless of the number and shape of adversarial patches.
- As a defense technique performed during the preprocessing stage, it does not require separate training of a model (detector) for patch detection. It can be applied universally regardless of the data or detection model, making it applicable to tasks such as image classification and object detection.

2. Related works

2.1. Adversarial attacks

2.1.1. Perturbation attack vs patch attack

Adversarial attacks are broadly categorized into two types based on the manipulated area of pixels: perturbation attacks, which manipulate all pixels in the image to execute the attack (Madry et al., 2017; Dong et al., 2018; Carlini and Wagner, 2017; Moosavi-Dezfooli et al., 2016), and patch attacks, which interfere by manipulating only a small restricted area of pixels within the image (Brown et al., 2017; Karmon et al., 2018; Liu et al., 2018; Lee and Kolter, 2019; Thys et al., 2019). Perturbation attacks involve applying imperceptible noise to the entire image, causing misclassification in the desired direction. A representative attack in this category is the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014), where pixel values are adjusted to induce misclassification in the desired direction.

Patch attacks involve applying adversarial perturbations only to specific parts of an image to disrupt learning, with methods such as those mentioned in Liu et al. (2018), Lee and Kolter (2019) and Thys et al. (2019) falling into this category. In contrast to the previously mentioned FGSM, patch attacks introduce significant noise only to specific regions, making them discernible to the human eye. This is because they create perturbations that anomalously influence the output, potentially influencing predictions for all inputs. Despite their local nature, these patches distort prediction outcomes across various inputs. One example of an adversarial patch, DPatch (Liu et al., 2018), is trained to simultaneously attack bounding box regression and object classification, enabling both untargeted and targeted attacks. However, a limitation of DPatch is its applicability only in digital scenarios. Subsequently, an advanced version, Robust DPatch (Lee and Kolter, 2019), has emerged, allowing physical attacks through methods such as varying lighting conditions, positions, transformations, or clipping within the RGB pixel range. Following this, adversarial patches that possess minimal color changes while being physically feasible have been developed (Thys et al., 2019).

¹ Our code is available at <https://github.com/heernink/RAPID>.

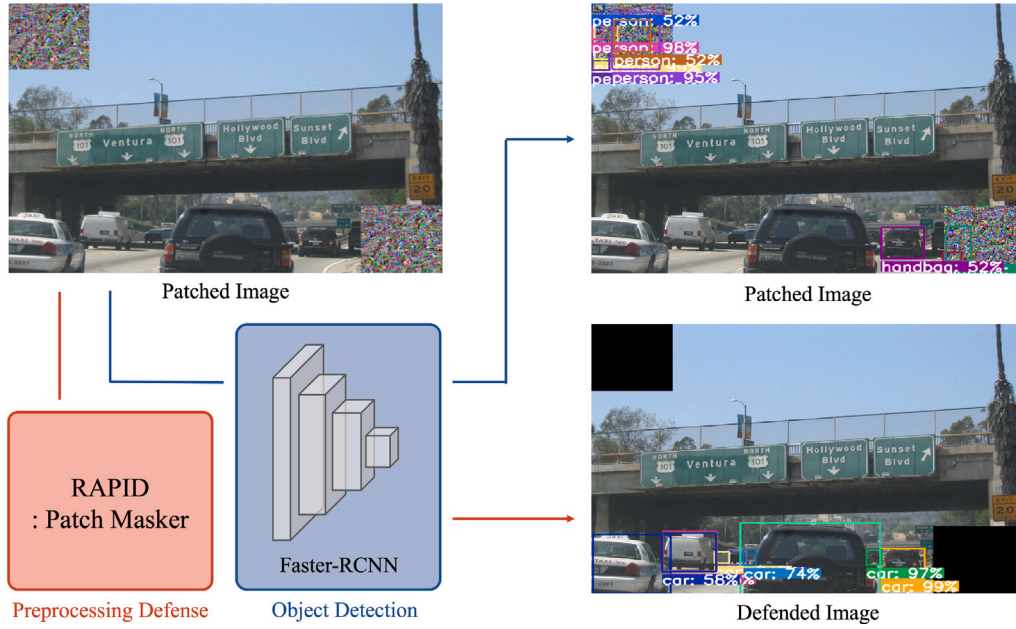


Fig. 1. In a patch attack scenario (upper-left image), object detection fails to correctly identify objects (upper-right image). However, by applying RAPID, patch area is masked, enabling the detection system to correctly identify the objects (lower-right image).

2.1.2. Digital attack vs physical attack

Adversarial attacks are divided into digital attacks and physical attacks based on the implementation scenario. Digital attacks primarily involve directly attacking all pixels in an image, where attackers typically generate adversarial perturbations on pixels within an image or video. When injected into a deep neural network model, such generated adversarial inputs can lead to incorrect predictions. However, these attacks rely on the assumption that the attacker has direct access to digital images and can manipulate all pixels just before being fed into the model. This makes the implementation of such attacks challenging in the real world. Additionally, digital disruptions are often imperceptible, making them difficult to detect clearly with cameras (Wei et al., 2022a).

On the other hand, physical attacks attempt to manipulate physical objects before camera imaging. This involves applying adversarial perturbations and stickers to create adversarial distortions in the physical environment, either by manipulating actual stop signals or disrupting camera images (Wei et al., 2022b). In essence, physical attacks primarily focus on the feasibility and efficiency of execution in real-world environments. Adversarial patches are a common method in physical attacks, offering practicality by enabling the modification of pixels in specific regions, unlike digital attacks. Moreover, they are easily printable and can be directly attached to the target for attack. Adversarial patches are trained for optimization in the digital domain and then placed on actual objects to carry out the attack. Various studies are ongoing to counteract such physical attacks, with particular attention given to defense methods against adversarial patches.

2.2. Defense methodologies

Various methods have been explored to counteract attacks utilizing adversarial patches. Defense strategies against adversarial attacks can be categorized into three stages: pre-processing, in-processing, and post-processing. In this context, we discuss pre-processing defense. Pre-processing defense encompasses methodologies aimed at mitigating the impact of adversarial perturbations before they enter the model. This defense strategy is further divided into image completion and image smoothing.

Image completion involves localizing the patch area first by assessing the difference between the adversarial patch and its surrounding

region. Subsequently, techniques such as masking or inpainting are applied based on the localized area to restore the patch. Digital Watermarking (DW) (Hayes, 2018) utilizes saliency maps (Simonyan et al., 2013) to detect densely populated regions of large gradient elements, masking them in the input. Jedi (Tarchoun et al., 2023) samples the patch area from an information-theoretic perspective and enhances the localization of adversarial patches by using entropy analysis and autoencoders. Segment and Complete (SAC) (Liu et al., 2022) trains a patch segmentation that outputs patch masks providing pixel-level locations of adversarial patch positions, reinforcing it with a self-adversarial training algorithm. PatchZERO (Xu et al., 2023) identifies adversarial pixels using a pixel-level patch detector and mitigates or removes their impact by replacing those pixels with the mean pixel value (0 value after image normalization).

One of the image smoothing methods, Feature Squeezing (Xu et al., 2017), aims to mitigate the effects of adversarial examples by ‘squeezing’ the features of the input data, including reducing the color depth of the input image to diminish detailed information. JPEG Compression (Das et al., 2017) applies compression in the JPEG format to images or videos as a method to reduce the impact of adversarial attacks. By removing some detailed information from the image, it can decrease or eliminate adversarial perturbations. Local Gradient Smoothing (LGS) (Naseer et al., 2019) directly smoothens high-gradient regions, considering the concentration of adversarial patch attacks on high-frequency noise. BlurNet (Raju and Lipasti, 2020) addresses the observation that adversarial attacks typically introduce high-frequency noise in the frequency domain of images. It achieves this by introducing low-pass filters to the feature maps of the network’s first layer, effectively removing high-frequency noise.

However, these methodologies have their respective limitations. Feature Squeezing (Xu et al., 2017) does not provide complete protection against all types of adversarial attacks. JPEG Compression (Das et al., 2017) may result in the loss of some crucial information from the original image during the compression process. In the case of LGS (Naseer et al., 2019), since densely concentrated abnormal gradient regions may exist in areas other than the patch region, it cannot guarantee the integrity of the image. Jedi (Tarchoun et al., 2023) sets an empirical threshold to identify patch regions, and this threshold-setting process heavily relies on the characteristics of the dataset used, making it challenging to exhibit a robust defense. SAC (Liu et al., 2022)

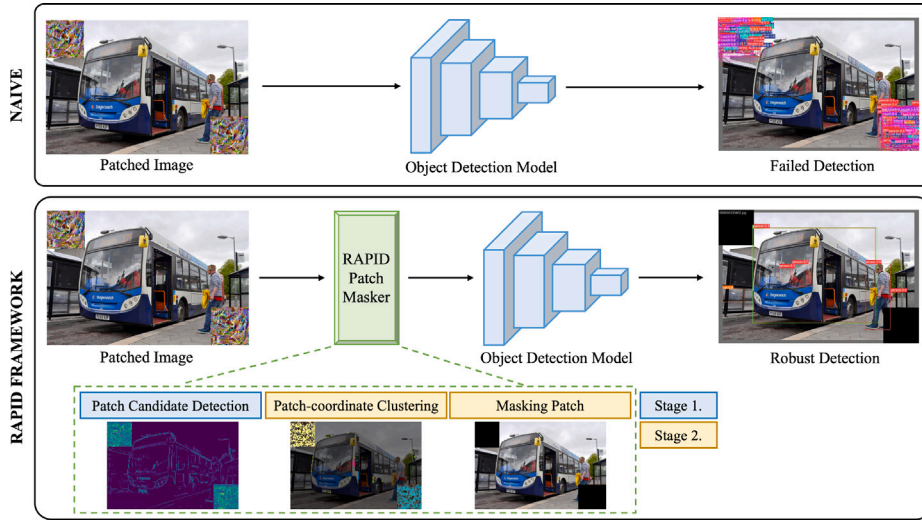


Fig. 2. RAPID Framework overview: RAPID employs a masking approach to defend against patch attacks, utilizing a two-stage framework. Initially, potential patch candidate locations are identified in attacked images, followed by density-based clustering of these coordinates. The method masks estimated patch regions on images, effectively nullifying the impact of patch attacks.

and PatchZERO (Xu et al., 2023) have limitations as they are effective in defending against patches for which the model has been pre-trained.

3. Proposed method

Defense against adversarial attacks depends heavily on the aggressiveness of the adversarial example used in the attack. An adversarial patch adds a significant perturbation to a localized region that dominates the prediction for all inputs, leading to a shape or contextual mismatch of the target object. As the number of hostile patches increases, they become more aggressive (Rossolini et al., 2023; Oonishi et al., 2022). However, much of the existing defense research focuses on single-patch attack scenarios, resulting in either undetected patches or poor defense performance in multi-patch situations. We aimed to devise a method for detecting and masking patches regardless of their number and shape. In addition, by proposing a defense technique at the in-processing stage, we aim to propose a technique that can be used universally and independently of object detection models. The following are the objectives of this paper.

- **Objective 1:** Address the challenge posed by the prevalence of multi-patch attacks, which exploit the vulnerability of existing defense mechanisms designed primarily for single-patch attacks.
- **Objective 2:** As object detection models become more intricate, defense methodologies reliant on specific model structures become less feasible. We devise defense strategies that remain independent of the object detection model's architecture.

Therefore, we propose an object detection model independent and machine learning model driven defense methodology named RAPID. Our methodology is not a simple naive masking technique but rather a framework that leverages the characteristics of patches to determine their positions for masking. This framework is built upon a two-stage approach utilizing two distinct patch features.

The first patch feature acknowledges that patches consist of arbitrary values based on noise, lacking semantic continuity across neighboring cells. In response, we introduce the Channel-wise Pooled Variance (CPV), a metric that quantifies random features when compared against adjacent cells. Subsequently, CPV is utilized to identify promising patch candidates. The second patch feature recognizes that patches are typically constructed with a fixed shape, predominantly rectangles, leading to the presence of densely clustered pixels with inherent randomness across adjacent cells. Consequently, the second stage employs

a coordinate-based clustering mechanism to identify densely populated areas, determining them as the final patches.

To provide a detailed description of the method, we introduce background knowledge in Section 3.1 and propose our framework in Section 3.2.

3.1. Preliminaries

3.1.1. Patch attack formulation

The two adversarial patches created for adversarial attacks are DPatch (Liu et al., 2018) and Robust DPatch (Lee and Kolter, 2019) which are designed to build on adversarial patch (Brown et al., 2017). The adversarial patch causes the model to misbehave by making it so that whatever manipulation is done to the input image in the physical world increases the probability of the intended class rather than the existing class. The following formula (1) is the objective function used to train this patch. Let $A(P, x, l, t)$ be the input image generated by attaching a patch P to the original image x , which includes transformations such as resizing and rotating the patch t and adjusting its position l . An adversarial patch $\hat{P}$ is generated by training the patch P to be classified into a certain class $\hat{y}$ for the created adversarial example $A(P, x, l, t)$.

$$\hat{P} = \operatorname{argmax}_{E_{x,l,t}} [\log P_r(\hat{y} | A(P, x, l, t))] \quad (1)$$

DPatch is capable of both untargeted and targeted attacks, and the objective function used to train the untargeted patch is as follow formula (2). Let $A(x, s, P)$ be the input image generated by attaching a patch P to the original image x , where the patch can be moved s . Train the patch P by maximizing the loss of the object decomposer such that $A(x, s, P)$ does not detect the true class $\hat{y}$ and the true bounding box $\hat{B}$.

$$\hat{P}_u = \operatorname{argmax}_{E_{x,s}} [L(A(x, s, P)); \hat{y}, \hat{B}] \quad (2)$$

The objective function for training targeted patch is as follows formula (3): Given an input image represented by $A(x, s, P)$, the loss of the object detector is minimized to train the patch P such that the target class y_t and the target box B_t are detected.

$$\hat{P}_t = \operatorname{argmin}_{E_{x,s}} [L(A(x, s, P)); y_t, B_t] \quad (3)$$

Robust DPatch differs from the digital attack-only DPatch by modifying DPatch through various methods such as accommodating diverse

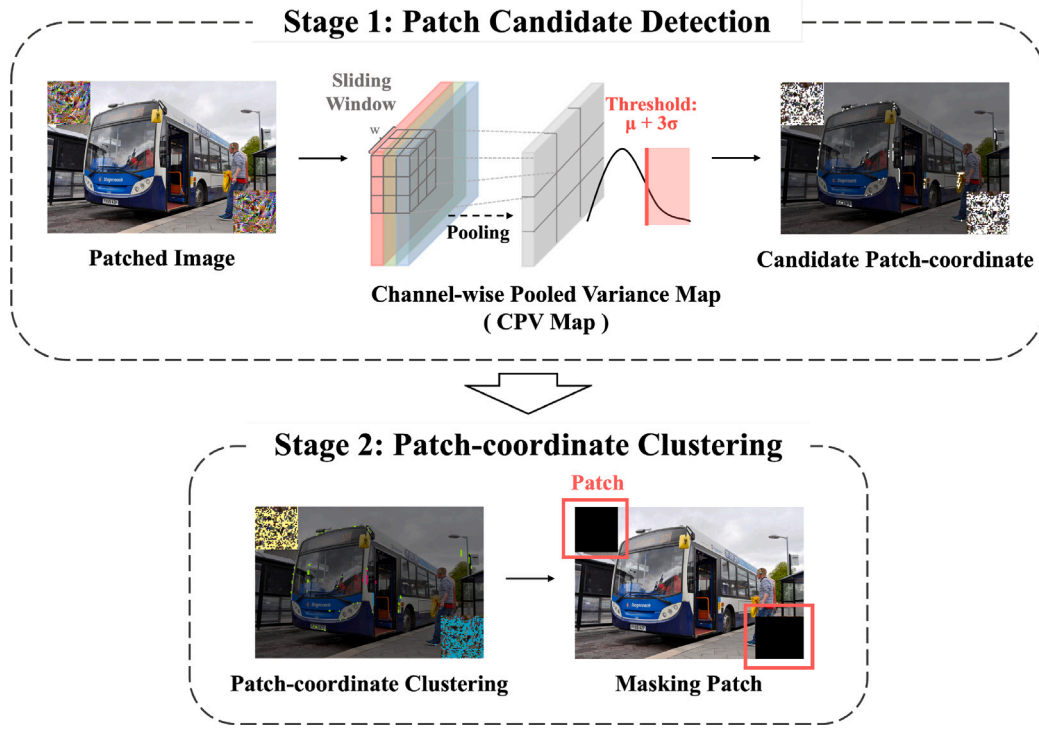


Fig. 3. The architecture of two-stage RAPID framework: The first stage calculates our Channel-wise Pooled Variance (CPV) to identify potential patches, while the second stage uses DBSCAN clustering (Ester et al., 1996) to confirm patch regions and applies masking to neutralize adversarial effects.

lighting conditions, positions, transformations, or clipping within the RGB pixel allowable range, enabling physical attacks. The method for generating adversarial examples used in model attacks after creating DPatch and Robust DPatch is as follows:

$$\hat{x} = (1 - M) \odot \tilde{x} + M \odot A(P, t) \quad (4)$$

where $\tilde{x}$ and $\hat{x}$ are the original image and the image with the adversarial patch attached, respectively. In formula (4), $\odot$ denotes the Hadamard product, $M \in \{0, 1\}^{H \times W \times C}$ represents a binary mask indicating the patch's position, and $A(P, t)$ transforms patch P according to transformation t to position it at the locations marked by M .

3.1.2. DBSCAN

Since patches are created with a consistent shape, they exhibit high density. Therefore, the DBSCAN algorithm (Ester et al., 1996), a density-based clustering method, is applied to detect and mask patch regions. DBSCAN is typically used for density-based clustering on data with outliers/noise that deviate significantly from the observed data range. It starts by assuming that data points belonging to the same group will be closely distributed to each other and distinguishes different clusters based on differences in data density within a specific space, while categorizing outliers/noise from the rest of the groups. This approach allows for the formation of clusters with various shapes without overlapping between clusters. Additionally, unlike distance-based clustering algorithms such as K-Means (Hartigan and Wong, 1979), there is no need to predefine the number of clusters. Consequently, patch localization is feasible regardless of the number and shape of the patches. Further elaboration on this will be provided in RAPID Framework stage 2.

3.2. RAPID framework

Fig. 2 provides a comparative visualization of the conventional object detection scenario and the outcome achieved by applying our proposed defense method. When images are affixed with patches, accurate detection becomes challenging when processed by the OD model.

In contrast, the RAPID framework effectively masks the patch regions in just two stages. As a result, images masked using our framework maintain their original detectability, ensuring robust detection even in the presence of adversarial attacks.

The architecture of RAPID consisting of two stages is presented in Fig. 3, and the algorithm is detailed through Algorithm 1.

- **First Stage (lines 2–8):** When an image is provided, it is passed through the RAPID framework, considering the possibility of attached patches. Channel-wise Pooled Variance (CPV) is calculated for each image window to identify potential patch candidates and values above a certain threshold are added to PC (Patch Candidates).
- **Second Stage (lines 10–15):** DBSCAN clustering is used to determine the final patch regions from the densely populated patch candidates. Afterward, masking is applied to the identified patch regions (line 16), removing the effects of adversarial patches and maintaining detection performance.

Further explanation is provided on how CPV is applied to identify potential patch candidates. Channel-wise Pooled Variance (CPV) is based on the pooled variance (Killeen, 2005) and is calculated by averaging the variances of each image channel. Our CPV treats each R, G, and B filter within a window region as a separate group and computes the pooled sample variance.

Let $X = (x_{ijc}) \in \mathbb{R}^{I \times J \times C}$ be a given image, where I, J, C correspond to the width (row), height (column), and channel dimensions, respectively. Let us assume that X has $C = 3$ channels corresponding to RGB. The term $CPV[i, j]$ is defined as the average of the channel-wise variances within a rectangular region of width and height w centered at position (i, j) of the image X . Let $X[i, j, c]$ indicate the i th row, j th column, c th channel elements of array X , the $CPV[i, j]$ value can be calculated as follows formula (5)–(7) (for convenience, let us refer to w as an odd number):

$$CPV[i, j] = \sum_{c \in R, G, B} (\mu^2(c) - (\mu(c))^2) / 3, \quad (5)$$

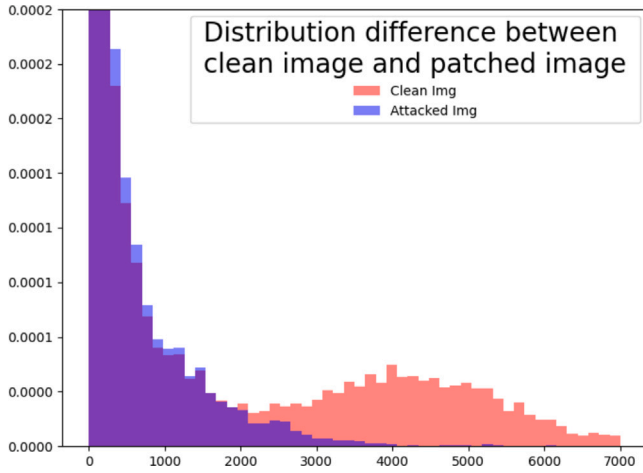


Fig. 4. CPV value differences between a regular clean image and a patched image. CPV values for the clean image cluster around 0, while those for the patched image exhibit a distinct distribution. This visual difference forms the basis for estimating patch region positions.

$$\mu^2(c) = \sum_{p=0}^{w-1} \sum_{q=0}^{w-1} X^2[i + p - \lfloor w/2 \rfloor, j + q - \lfloor w/2 \rfloor, c] / w^2 \quad (6)$$

$$\mu(c) = \sum_{p=0}^{w-1} \sum_{q=0}^{w-1} X[i + p - \lfloor w/2 \rfloor, j + q - \lfloor w/2 \rfloor, c] / w^2 \quad (7)$$

where $\lfloor \cdot \rfloor$ represents the floor function, which indicates the greatest integer less than or equal to a given number. If i and j are at the boundary values, calculations are performed using zero-padding.

Examining the equation, it becomes evident that $CPV[i, j]$ represents the mean of variances computed over the window intervals for each channel at the (i, j) th position.

Compared to regular images, patches exhibit higher CPV values. Fig. 4 compares CPV distributions of a regular clean image and a patch image of the same size. The CPV distribution of the patch image is skewed to the right compared to the clean image CPV distribution. Therefore, CPV values can distinguish between regular image regions and patch regions in images with attached patches, identifying areas with high CPV values as patches.

3.2.1. Stage 1: Patch candidate detection

In conventional images, there exists a concept known as pixel connectivity, which characterizes the relationship between pixels. Images with connected adjacent pixels typically exhibit a continuous transition of color and shape, representing a natural form of image representation. However, patches are designed to disrupt image recognition. As a result, the continuity of color among adjacent pixels is deliberately disrupted in patches, aligning with the objective of hindering image recognition. Furthermore, patches tend to possess values that deviate substantially from adjacent pixels, thereby displaying a unique characteristic of diverging from nearby pixel values. Thus, we introduce the concept of CPV as a metric to differentiate between the regular image content and the patch-infested regions in an image.

Line 1 initializes an empty list named PC used to store candidates identified as potential patches throughout the algorithm. In Line 2, the Channel-wise Pooled Variance (CPV) map, denoted as $V = CPV(X) = (v_{ij}) \in \mathbf{R}^{I \times J}$, is computed for the given image X . This involves calculating the CPV at each location (i, j) in the image, where v_{ij} in V represents the CPV at that location. In Line 3, the mean (μ) and standard deviation (σ) of the CPV values across the entire map V are calculated. These statistical measures are used as a criterion for identifying outliers that significantly deviate from the norm, which may indicate the presence of patches. From Lines 4 to 8, the algorithm loops

Algorithm 1 RAPID Patch Masker

Input: Attacked Image X

Parameter: Window size w (searching area; w^2), Clustering distance threshold ϵ , Minimum samples in cluster ms

Output: Robust detection Y_{robust}

```

1:  $PC :=$  empty list for Patch Candidates
2:  $V \leftarrow CPV(X, w)$  ▷ Stage 1 (Lines 2-8)
3:  $\mu, \sigma \leftarrow \text{Mean}(V), \text{Std}(V)$ 
4: for  $v \in V$  do
5:   if  $v > \mu + 3 * \sigma$  then
6:     Append  $v$  to  $PC$ 
7:   end if
8: end for
9:  $P :=$  empty list for Patch-coordinate set
10:  $Cls \leftarrow \text{Clustering}(PC, \epsilon)$  ▷ Stage 2 (Lines 10-16)
11: for  $c \in Cls$  do
12:   if  $\text{Len}(c) < ms$  then
13:     Append  $c$  to  $P$ 
14:   end if
15: end for
16:  $M = \text{Mask}(X, P)$ 
17:  $Y_{robust} \leftarrow \mathcal{OD}(M)$  ▷  $\mathcal{OD}$ : Vanilla Object Detection

```

through each value v_{ij} in CPV map V . In Line 5, it checks whether each CPV value v_{ij} exceeds the threshold $\mu + 3 * \sigma$. This threshold is derived from the common statistical practice of anomaly detection, specifically the 3-sigma rule (Lehmann, 2013), which helps in identifying outliers in datasets. If v_{ij} is greater than $\mu + 3 * \sigma$, it is considered a significant deviation, suggesting that the pixel may be part of an adversarial patch. In Line 6, any v_{ij} that meets this condition is added to the patch candidate list PC , which was initialized in Line 1. This list PC compiles all potential patch locations identified through the CPV analysis. Line 7 and Line 8 mark the end of the conditional statement and the loop, respectively.

3.2.2. Stage 2: Patch-coordinate clustering

In Line 10, the patch candidates list (PC), which contains all potential patch locations identified earlier, is subjected to a density-based clustering process. This step is essential to group the CPV high points that are likely part of the same patch based on their spatial proximity. In Line 11, the algorithm explores over each cluster c in the set of clusters Cls obtained from the density-based clustering. Each cluster represents a group of closely positioned CPV points that potentially form a patch. In Line 12, during the process, the algorithm checks if the number of elements in a cluster c exceeds a minimum size threshold ms . This threshold ensures that only significant clusters, which are large enough to likely be actual patches rather than noise or smaller anomalies, are considered for further processing. In Line 13, clusters c that meet the size criterion are added to another list P , which is designated for holding coordinates for the patch regions confirmed for masking. This list P accumulates all the clusters that are determined to be valid patches based on their density and size. In Line 14 marks the end of the conditional check within the iteration. This ensures that only clusters larger than the minimum size threshold ms are processed further. In Line 15, the iteration over the set of clusters Cls concludes. At this point, all significant clusters have been evaluated and those meeting the criteria have been added to P .

In Line 16, within the $\text{Mask}(X, P)$ function, each cluster's coordinates contained in P are used to apply a mask over the respective areas in the image X . This masking process neutralizes the impact of identified patches by obscuring them, thus preparing the image for cleaner object detection processing. Finally, an arbitrary Object Detection (OD) algorithm, denoted as $\mathcal{OD}(M)$, is applied on the masked image M to generate the robust detection results $Y_{robust} = \mathcal{OD}(M)$. This

Table 1

The COCO 2017 training dataset (Lin et al., 2014) used for training consists of a total of 5000 images, and the same number of images from the validation dataset were used during inference. The APRICOT dataset (Braunegg et al., 2020) is designed to consider physical attack scenarios and consists of images captured in real-world settings such as markets and parks, where adversarial patches with attack capabilities have been printed and installed. This dataset was also used during the inference process.

Task	Dataset	Count	Shape (h, w, c)
Train	COCO 2017 Train Dataset	5000	(640, 640, 3)
Inference	COCO 2017 Validation Dataset	5000	(640, 640, 3)
	Apricot Dataset	1000	(4032, 3024, 3)

output represents the final object detection outcomes after accounting for and mitigating the impact of adversarial patches.

4. Experiments

4.1. Experimental setup

To defend against adversarial attacks, it is crucial to generate adversarial patches with significant attack potency. For this purpose, we trained DPatch (Liu et al., 2018) and Robust DPatch (Lee and Kolter, 2019) using a dataset comprising 5,000 images from the MS COCO 2017 training set (Lin et al., 2014). This dataset features complex scenes with multiple objects spanning 80 different categories. Each image was trained for 100 epochs to induce misclassification in the object detector. The adversarial patches used in the attacks illustrated in Fig. 2 were also generated using the same methodology.

To evaluate the RAPID Framework, we used Faster R-CNN (Ren et al., 2015) as the object detector. The performance evaluation was carried out on the validation dataset, which includes 5,000 images from the COCO 2017 Dataset (Lin et al., 2014) and was also used to train the adversarial patches mentioned earlier. In summary, the datasets used for training the patches consisted of 5,000 images from the COCO 2017 Training Dataset, while the COCO 2017 Validation Dataset was used for evaluating the defense performance.

Additionally, assuming a physical attack scenario, we used 1000 images from the APRICOT dataset (Braunegg et al., 2020) for inference. The APRICOT dataset is specifically designed for testing adversarial patch attacks in real-world settings and includes images taken in both indoor and outdoor environments such as parks, grocery stores, and other public spaces. The images feature adversarial patches placed on various surfaces and objects to simulate realistic attack scenarios, providing a comprehensive evaluation of the RAPID framework's ability to handle diverse physical conditions and challenges. Detailed information about the datasets can be found in the 1 below.

The RAPID framework calculates the CPV by partitioning the image into windows, causing the threshold value of each image's CPV to vary depending on the window size. Therefore, in this paper, we fix the image size at 640×640 and calculate the CPV values of 3×3 windows to determine the threshold values. The threshold value is set based on the mean pixel value and values exceeding three times the standard deviation (3σ) from the mean. The epsilon (ϵ) variable, which checks the density of candidate pixels, and the minimum number of samples are set to 0.07 and 13, respectively, for our experiments. The detailed hyperparameter values can be found in the Table 2 below.

4.1.1. Metric

The effectiveness of our proposed method in capturing adversarial attack regions is evaluated using Intersection over Union (IoU) (Rezatofighi et al., 2019) and Intersection over Area (IoA) (Xiang et al., 2023) metrics.

IoU is commonly used in the field of computer vision to measure the accuracy of bounding boxes. In our context, it is employed to assess the extent of alignment between adversarial patches and predicted clusters. IoU is the ratio of the intersection area to the union area of two sets, with lower values as the union area grows larger. This metric provides

Table 2

The hyperparameters for the RAPID Framework and Faster-RCNN used in this experiment are as follows. To detect regions with dense adversarial pixels, the values for epsilon and min samples were set to 0.07 and 13, respectively. For evaluating the performance of Faster-RCNN after applying our framework, we used the most commonly used hyperparameters. The threshold used in the NMS (Non-Maximum Suppression) stage was set to 0.5, the number of bounding boxes detected per image was set to 100, and the class threshold was set to 0.05.

Module	Hyper-Parameter	Value
RAPID	img size	(640, 640)
	window size	3
	ϵ	0.07
	min samples	13
	candidate _{thr}	$\mu \pm 3\sigma$
Faster-RCNN	nms _{thr}	0.5
	detections per img	100
	rpn nms _{thr}	0.7
	score _{thr}	0.05

insight into how well the predictions match the patch presence. IoA, similar to IoU, focuses on the target area and measures the proportion of correct predictions within that target area. The denominator of this metric encompasses the patch area, reflecting how well the model's predictions align with the actual patch region.

Utilizing both metrics in tandem allows us to evaluate RAPID's performance in predicting both the presence and the accuracy of patch regions. The motivation behind using these two metrics is to assess not only whether the patch areas were accurately identified but also how precisely they were predicted. Therefore, IoU and IoA serve as pivotal indicators for evaluating RAPID's performance. It is essential to ensure that both IoU and IoA values are close to 1. This is because when covering areas outside the patch, IoA tends to have a high value close to 1 while IoU may have a lower value. Additionally, if the patch is not completely covered, both IoU and IoA will have lower values.

Furthermore, mean Average Precision (mAP), commonly used as a performance metric for object detectors, is also utilized as an evaluation metric. mAP represents the average precision (AP) across individual objects. AP is a metric that quantifies the performance of recognition algorithms as a single value, calculated as the area under the precision-recall curve. A higher value indicates superior performance.

The Average Precision (AP) value dynamically changes with different threshold values, affecting precision and recall. Given that the threshold can vary depending on the specific task or the characteristic of the data, this paper evaluates the performance of the object detector considering these scenarios. To assess the general performance of the object detector, we use the mAP50 value, which is the mean Average Precision at a fixed threshold of 0.5, and the mAP@50-95 value, which is the mean Average Precision averaged over multiple thresholds from 0.5 to 0.95. These metrics are used to evaluate how the performance of the object detector changes after applying various adversarial defense methods.

4.2. Defense against adversarial attack scenario

In this section, we assess the effectiveness of our approach in detecting the positions of various adversarial patch attack scenarios.

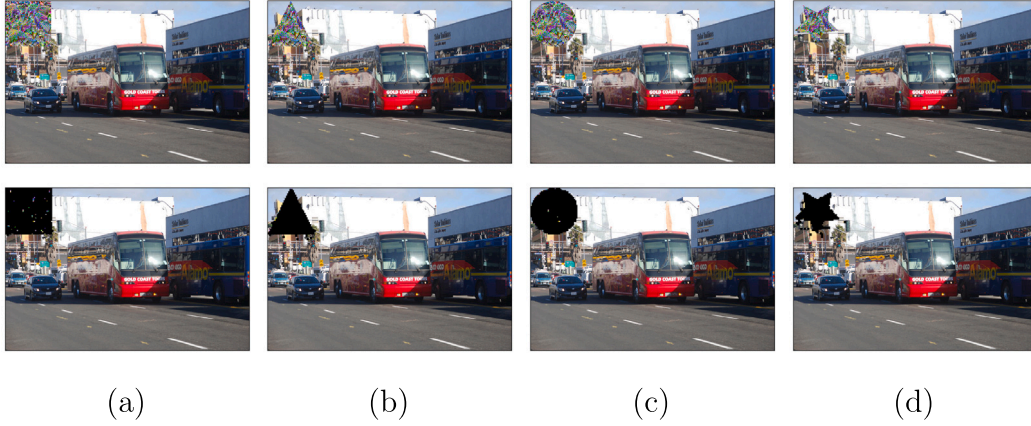


Fig. 5. Not only for the basic square-shaped adversarial patches, but also for adversarial attack scenarios with triangular, star-shaped, and circular patches, the aim is to localize the patch area as precisely as possible and perform masking.

Table 3

To ensure uniform conditions for experimentation, the images were resized to 640×640 , and the patch size was fixed at 120×120 . It was observed that RAPID's localizing performance was superior when both patches were square in shape.

Patch	Shape	IoU	IoA	Patch	Shape	IoU	IoA
DPatch	Square	0.8614	0.9805	Robust Dpatch	Square	0.9260	0.9941
	Triangle	0.7709	0.8910		Triangle	0.7566	0.9016
	Star-shaped	0.7263	0.9142		Star-shaped	0.7182	0.9049
	Circle	0.7720	0.8550		Circle	0.7097	0.8172

To achieve this, we examine how the model's performance varies concerning the shape, number, and size of the basic adversarial patches. This evaluation allows us to determine the model's ability to predict the patch's position accurately. Performance is measured using IoU and IoA, which gauge how precisely the patch is predicted. We conduct experiments by attaching adversarial performance-enabled Dpatch and Robust Dpatch to the MS COCO dataset to assess performance.

4.2.1. Defense performance based on patch shape

The experiments conducted for four different shapes of patches, rather than the commonly used rectangular patches, are illustrated in Fig. 5. Rectangular, triangular, circular, and star-shaped patches were evaluated and applied to the dataset for experimentation. The top four rows in Fig. 5 depict the Dpatch attached to normal images, while the results after passing through the RAPID framework can be observed in the bottom four rows of images. Table 3 presents the IoU and IoA values for each shape by adversarial patch. When the patch size is 120×120 , the rectangular patch demonstrates the most accurate patch positioning, with an IoU value of 0.8614 and an IoA value of 0.9805, according to the Dpatch baseline. Furthermore, the IoU performance for the star-shaped patch, based on Dpatch, is the lowest at 0.7263, indicating a lower precision in localizing, as seen in the fourth image from the bottom in Fig. 5, due to excessive coverage of areas outside the patch.

4.2.2. Defense performance according to patch size and quantity

The experiment evaluated the performance of the model based on the number and size of adversarial patches applied. To maintain a fixed ratio between the size of the patches and the images, the image size in the dataset was changed to 640×640 . The patch sizes were fixed at S (90×90), M (120×120), and L (150×150), while the number of patches varied from 1 to 5 (Fig. 6). The results of the model's localizing performance are presented in Table 4 and Fig. 7.

When comparing sizes, it was observed that RAPID performed best when the two adversarial patches' size was 120×120 . Considering the results in conjunction with the number of patches, for S-sized patches, an increasing trend in IoU values was noted as the number of patches increased. However, for M or L-sized patches, the IoU values remained

Table 4

The table shows the performance of RAPID according to the number and size of adversarial attacks. The sizes S, M, and L correspond to 90, 120, and 150, respectively. Each patch is randomly placed in the image and then fed into the RAPID framework as an input image. When the patch size is M, the overall IoU and IoA values are higher, indicating better performance.

Patch	Size	#	IoU	IoA	Patch	Size	#	IoU	IoA
Dpatch	S	1	0.4602	0.9292	Robust Dpatch	S	1	0.5383	0.9392
		2	0.4977	0.9273			2	0.6061	0.9393
		3	0.5311	0.8928			3	0.6356	0.9457
		4	0.5180	0.8841			4	0.6264	0.9473
		5	0.5324	0.8605			5	0.6420	0.9419
	M	1	0.8614	0.9805		M	1	0.9260	0.9941
		2	0.8608	0.9885			2	0.8883	0.9967
		3	0.8512	0.9944			3	0.8675	0.9854
		4	0.7877	0.9646			4	0.8404	0.9809
		5	0.6645	0.7972			5	0.6164	0.7207
	L	1	0.7852	0.9453		L	1	0.7887	0.8982
		2	0.7872	0.9687			2	0.8269	0.9682
		3	0.6211	0.7557			3	0.7597	0.8786
		4	0.2781	0.3250			4	0.4001	0.4539
		5	0.0846	0.0961			5	0.1002	0.1116

high for up to 2 patches, but decreased as the number exceeded this threshold. Particularly for larger patch sizes, when the number of patches exceeded 4, there was a significant decrease in performance regardless of the patch type. This phenomenon arises because when multiple patches of a certain size cover a considerable portion of the image (as shown in Fig. 6), it becomes challenging to distinguish between the patch regions and the general image areas, leading to performance degradation. Therefore, in realistic attack scenarios, where typically 1–3 patches are used and objects are not obscured, the effectiveness of the RAPID model was confirmed.

4.3. Comparison with other defenses method

To evaluate the performance of RAPID, we selected LGS (Naseer et al., 2019), Jedi (Tarchoun et al., 2023), and SAC (Liu et al., 2022) as comparative methods. These state-of-the-art methods are considered



Fig. 6. (a), (b), and (c) represent Dpatch sizes S, M, and L, respectively, while the number of adversarial patches ranges from 1 (left) to 5 (right), gradually increasing. When the patch size is large (c), it occupies a relatively wide area of the entire image. As a result, the overall CPV value for the image region increases, leading to instances where the patch area is not accurately detected.

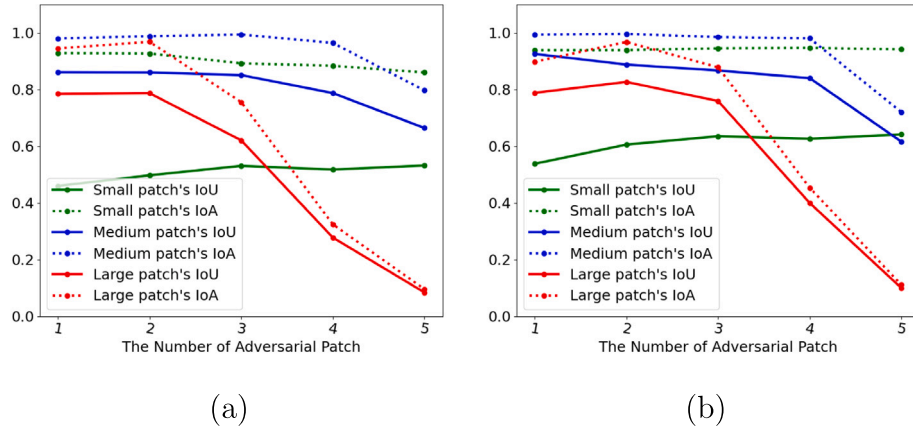


Fig. 7. Localizing performance (IoU and IoA) under patch attack varying the patch size and the number of patches: (a) DPatch, (b) Robust DPatch. Except for when the patch size is S, it can be observed that the metric values decrease as the number of patches increases.

effective in defending against adversarial patch attacks in different ways. All three methods belong to the pre-processing defense category, which aims to mitigate the impact of adversarial perturbations before using as input. These methods were chosen because their shared defense strategy facilitates a direct comparison with RAPID. We ensured consistent experimental conditions to ensure an impartial comparison. Specifically, our experiments utilized the Faster R-CNN model (Ren et al., 2015) for detecting objects and were based on the MS COCO Dataset (Lin et al., 2014). We employed our generated DPatch (Liu et al., 2018) and Robust DPatch (Lee and Kolter, 2019) for the adversarial attacks.

4.3.1. Patch localization performance

IoU and IoA evaluation metrics were employed simultaneously to assess how accurately the adversarial patch regions were detected. The results are presented in Table 5, and Fig. 8 visualizes the pixels altered by each method through masking or smoothing for defense purposes. For LGS (Naseer et al., 2019), the IoU and IoA values were 0.392 and 0.998, respectively, for a single patch, and 0.274 and 0.967 for multiple patches. Although IoA is high, IoU yields a low value due to the smoothing of high gradient regions, affecting areas beyond the patch region, as evidenced in Fig. 8(a). SAC (Liu et al., 2022) demonstrated IoU and IoA values of 0.921 for both a single and multiple patches. However,

Table 5

This table shows Patch Localization Performance compared with the baseline group (LGS Naseer et al., 2019, SAC Liu et al., 2022 and Jedi Tarchoun et al., 2023). When compared to other state-of-the-art models, in the case of a single patch attack scenario, the SAC methodology shows the highest IoU, while the LGS method has the highest IoA. However, in scenarios with multiple patches, our proposed RAPID method achieves the highest values in both metrics.

# of patch	Metric	Defense method	Value
Single	IOU	LGS	0.392
		SAC	0.921
		JEDI	0.060
		Ours	0.861
	IOA	LGS	0.998
		SAC	0.921
		JEDI	0.070
		Ours	0.980
Multi	IOU	LGS	0.274
		SAC	0.571
		JEDI	0.040
		Ours	0.861
	IOA	LGS	0.967
		SAC	0.571
		JEDI	0.050
		Ours	0.989



Fig. 8. When subjected to a Dpatch attack, (a) LGS (Naseer et al., 2019) shows that not only the patch but also the area around the patch is affected; (b) SAC (Liu et al., 2022) indicates that when there are two patches, the defense is not effective; (c) Jedi (Tarchoun et al., 2023) shows that defense is possible depending on the threshold, so it fails without proper adjustment. (d) Our methodology, RAPID, detects and masks the patches effectively.

as the number of patches used in the attack increases, both IoU and IoA values decrease, indicating SAC's inability to accurately detect patch regions, particularly evident in Fig. 8(b). For Jedi (Tarchoun et al., 2023), both single and multiple patches yielded values below 0.1. Jedi's detection heavily relies on exceeding a certain threshold of pixel entropy to identify patches. As depicted in Fig. 8(c), Jedi fails to detect the patch region. In contrast, RAPID exhibits slightly lower IoU and IoA values of 0.861 and 0.980, respectively, for a single patch. However, for multiple patches, RAPID achieves higher IoU and IoA values of 0.861 and 0.989, respectively, indicating superior adversarial patch localization compared to other methods.

4.3.2. Defense performance comparison

We utilize the mean Average Precision (mAP) to demonstrate the effectiveness of attacks and defenses, and the results are presented in Table 6. The mAP50 and mAP@50-95 values in a non-attacked

Table 6

This table shows detector's performance after defense against adversarial patch attacks compared with the baseline group (LGS Naseer et al., 2019, SAC Liu et al., 2022 and Jedi Tarchoun et al., 2023). In the Adversarial attack scenario, regardless of the number of patches, the RAPID significantly enhances the detector's performance, nearly restoring it to the mAP observed when no attack is present.

Patch	# of patch	Defense	mAP ₅₀	mAP _{@50-95}
No Attack	None	No defense	0.679	0.475
Dpatch	Single	No defense	0.435	0.297
		LGS	0.647	0.449
		SAC	0.683	0.479
		JEDI	0.435	0.297
	Multi	Ours	0.696	0.479
		No defense	0.400	0.269
		LGS	0.620	0.424
		SAC	0.651	0.449
Robust Dpatch	Single	JEDI	0.401	0.273
		Ours	0.656	0.451
	Multi	No defense	0.284	0.199
		LGS	0.655	0.454
		SAC	0.683	0.479
		JEDI	0.308	0.209
	Multi	Ours	0.670	0.470
		No defense	0.256	0.184
		LGS	0.630	0.429
		SAC	0.644	0.446
		JEDI	0.292	0.192
		Ours	0.644	0.444

scenario are 0.679 and 0.475, respectively. However, when attacked using Dpatch, the performance drops to 0.435 and 0.297 with one patch and 0.4 and 0.269 with two patches, indicating a decrease in object detection performance. From the experimental results, it can be observed that LGS (Naseer et al., 2019) (mAP50: 0.647, mAP@50-95: 0.449) and Jedi (Tarchoun et al., 2023) (mAP50: 0.435, mAP@50-95: 0.297) exhibit lower defensive performance compared to other methodologies for single Dpatch. Particularly, Jedi shows notably low values requiring appropriate thresholds depending on the dataset. In contrast, SAC (Liu et al., 2022) (mAP50: 0.683, mAP@50-95: 0.479) and RAPID (mAP50: 0.696, mAP@50-95: 0.479) demonstrate high defensive performance.

4.3.3. Application of defense techniques based on adversarial attack presence

Adversarial attacks using patches occur in real-time in actual environments. Therefore, it is desirable for defense techniques to be applied only in adversarial attack scenarios. If defense techniques are applied when no adversarial attack has occurred, it may lead to image distortion issues, which could adversely affect the object detector. When each defense technique is applied to the original images without adversarial attacks, the experimental results can be observed in Fig. 9. The mAP results of SAC (Liu et al., 2022) (mAP50: 0.670, mAP@50-95: 0.465) and RAPID (mAP50: 0.670, mAP@50-95: 0.472) closely match the results of the original images (mAP50: 0.679, mAP@50-95: 0.475), indicating that they effectively detect the absence of adversarial attacks. However, LGS (Naseer et al., 2019) (mAP50: 0.633, mAP@50-95: 0.438) adversely affected the original images even in the absence of adversarial attacks, resulting in the lowest mAP value. Jedi (Tarchoun et al., 2023) (mAP50: 0.668, mAP@50-95: 0.460) showed no adverse effects due to image distortion, indicating its resilience to defense difficulties.

4.3.4. Defense against physical attack

The robustness of RAPID against physical patch attacks was evaluated using the APRICOT dataset (Braunegg et al., 2020). The results are presented in Table 7. Considering both IoU and IoA values, RAPID demonstrates the highest performance, with IoU and IoA values of

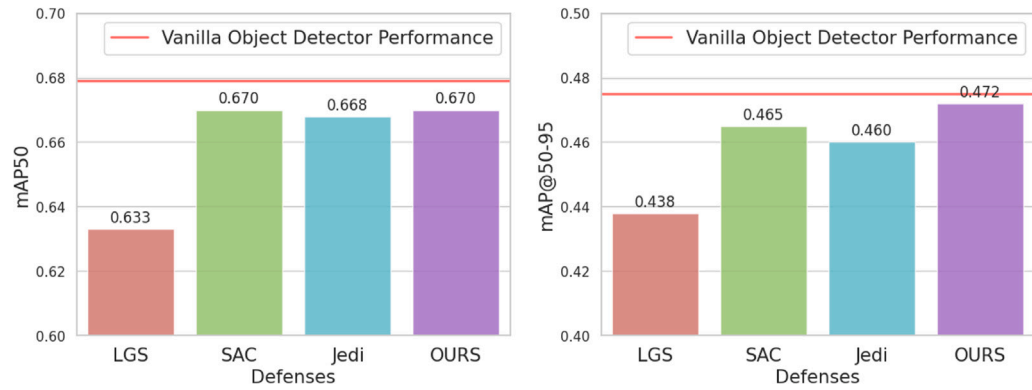


Fig. 9. This figure illustrates mAP of each defense using clean images. When each defense technique was applied to the original images without adversarial attacks, the mAP results for SAC and RAPID closely matched those of the original images, indicating that these methods effectively detect the presence of adversarial attacks.

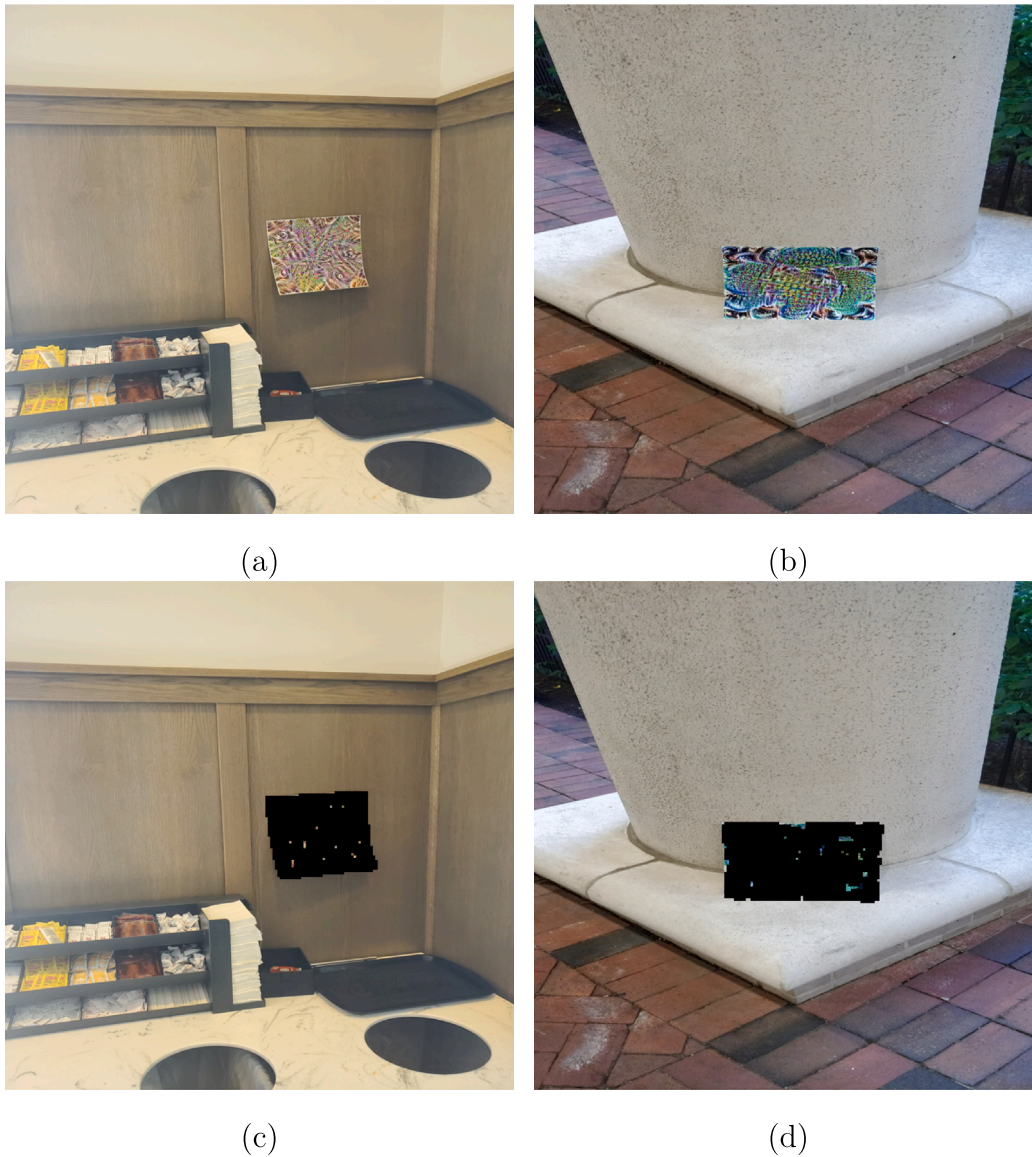


Fig. 10. When subjected to a physical attack using APRICOT dataset (Braunegg et al., 2020), (a) and (b) are images physically attacked, and (c) and (d) are images after RAPID defense for each attacked image. It can be observed that only the regions affected by the adversarial attack have been masked.

Table 7

This figure shows Localizing performance of defense using APRICOT dataset compared with the baseline group (LGS [Naseer et al., 2019](#), SAC [Liu et al., 2022](#) and Jedi [Tarchoun et al., 2023](#)). Even in physical attack scenarios, our method demonstrates more precise localization of the attacked regions compared to the other three methodologies, as evidenced by higher IoA and IoU values.

Adversarial attack	Metric	Defense method	Value
Physical attack	IOU	LGS	0.332
		SAC	0.617
		JEDI	0.051
		Ours	0.658
	IOA	LGS	0.821
		SAC	0.628
		JEDI	0.033
		Ours	0.687

Table 8

This table shows Adversarial patch defense time measurement. Our methodology is found to be approximately twice as fast as the similar SAC methodology in terms of localization performance.

Patch	# of patch	Defense	Inference time (ms per image)
Dpatch	Single	LGS	0.0299
		SAC	0.0443
		JEDI	39.676
		Ours	0.0220
	Multi	LGS	0.0320
		SAC	0.0463
		JEDI	37.1543
		Ours	0.0237
Robust Dpatch	Single	LGS	0.0322
		SAC	0.0559
		JEDI	38.1640
		Ours	0.0257
	Multi	LGS	0.0341
		SAC	0.0607
		JEDI	38.3570
		Ours	0.0369

0.658 and 0.687, respectively, validating its defense capability against physical attacks. In contrast, LGS ([Naseer et al., 2019](#)) exhibits a high IoA value of 0.821 but a low IoU value of 0.332. Jedi ([Tarchoun et al., 2023](#)) shows low values for both IoU and IoA, at 0.051 and 0.033, respectively. SAC ([Liu et al., 2022](#)) achieves IoU and IoA values of 0.617 and 0.628, respectively, slightly lower than RAPID. The performance of RAPID is further depicted in [Fig. 10](#), where (a) and (c) represent images subjected to adversarial attacks from the APRICOT dataset. Upon pre-processing through the RAPID methodology, these images exhibit mitigation of adversarial attack effects, as depicted in (b) and (d), respectively, where masking effectively disrupts the adversarial properties.

4.3.5. Adversarial patch defense time measurement

As mentioned earlier, adversarial patches are subjected to physical attacks, meaning that it is crucial to defend against them quickly in real-time environments. Therefore, when an image is subjected to adversarial attacks, the time taken to apply each defense mechanism to suppress the patch's aggressiveness was measured. The measurement results ([Table 8](#)) revealed that RAPID exhibited the fastest defense speed. This indicates the potential effectiveness of our method in real-time applications, including autonomous driving contexts.

5. Limitation and discussion

While the RAPID framework represents a significant advancement in defending against adversarial patch attacks, there are several limitations that need to be addressed to enhance its applicability and effectiveness.

First, the masking of adversarial attack locations can lead to information loss within the image, potentially impacting object detection performance negatively. Since masking obscures parts of the image, critical details necessary for accurate detection may be hidden. Second, although RAPID is effective in typical attack scenarios, its defensive capabilities may be limited in extreme situations where unrealistically large patches or numerous patches are used. The inherent nature of adversarial attacks involves attaching patches irrespective of whether they obscure objects. Attackers often focus solely on the success of the attack, which can result in large patches being attached to images. In such cases, the performance of the RAPID framework may degrade, making it challenging to accurately detect and defend against the patches. Third, experiments have only been conducted with DPatch and Robust DPatch, which limits the evaluation of how well RAPID can respond to a broader range of adversarial attacks. The lack of experimentation with other patch types restricts our understanding of RAPID's adaptability and robustness. Fourth, our research has so far focused primarily on autonomous driving scenarios in proposing defense methodologies. However, adversarial attacks can occur across various fields, not just in autonomous driving. Therefore, it is necessary to extend our research to include broader domains such as face detection.

By addressing these limitations, RAPID can be further improved to provide a more comprehensive defense against the evolving landscape of adversarial attacks. Future studies should explore these areas to enhance RAPID's practicality in real-world scenarios not only autonomous driving scenarios also diverse situations facing practical challenges such as face imitation crime, ensuring it remains a reliable solution for robust object detection.

6. Conclusion

RAPID offers a novel defense against adversarial patch attacks, excelling in both single and multi-patch scenarios. Utilizing channel-wise pooled variance and a two-stage approach, RAPID accurately detects and masks patch regions. The evaluation on the MS COCO datasets shows that RAPID achieves performance improvements in mean Average Precision (mAP) ranging from approximately 0.03 to 0.25 compared to other defense methods, with the most significant improvements observed over LGS ([Naseer et al., 2019](#)) and Jedi ([Tarchoun et al., 2023](#)) methods. Its real-time adaptability enhances practicality, making RAPID a significant stride in bolstering object detection system security, especially in autonomous driving and beyond.

Despite these advancements, there are several future directions to explore for enhancing RAPID's effectiveness and applicability. To mitigate the issue of information loss due to masking, future research could integrate inpainting techniques to restore masked areas, thus preserving crucial image details for accurate detection. Additionally, addressing the framework's limitations in handling extreme adversarial scenarios with large or numerous patches is vital. Experimenting with image partitioning and rearrangement could offer potential solutions in such cases. Expanding RAPID's evaluation beyond DPatch and Robust DPatch to include a wider range of adversarial attacks will be essential in demonstrating its adaptability and robustness. Lastly, extending the application of RAPID beyond autonomous driving to other fields such as face detection can broaden its utility. In this endeavor, we intend to consult relevant studies in face detection ([Ali et al., 2019b,a](#)) to design methodologies that are informed by established research in the field. By addressing these areas, RAPID can evolve into a more comprehensive and versatile defense strategy, capable of countering adversarial attacks across diverse real-world situations.

CRedit authorship contribution statement

Heemin Kim: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Byeong-Chan Kim:** Writing – original draft, Visualization,

Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Sumi Lee**: Software, Methodology, Investigation, Conceptualization. **Minjung Kang**: Software, Methodology, Investigation. **Hyunjee Nam**: Software, Methodology, Investigation. **Sunghwan Park**: Writing – review & editing, Software, Methodology, Investigation, Conceptualization. **Il-Youp Kwak**: Writing – review & editing, Supervision, Methodology, Investigation, Funding acquisition, Conceptualization. **Jaewoo Lee**: Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2022R1F1A1062660); This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-2020-0-01655) supervised by the IITP(Institute of Information & Communications Technology Planning & Evaluation); This study was supported by the BK21 FOUR program (Education and Research Center for Securing Cyber-Physical Space) through the National Research Foundation (NRF) funded by the Ministry of Education of Korea (5199990314137). This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science and ICT (RS-2023-00208284).

References

- Ali, A.A., El-Hafeez, T.A., Mohany, Y.K., 2019a. An accurate system for face detection and recognition. *J. Adv. Math. Comput. Sci.* 33 (3), 1–19.
- Ali, A.A., El-Hafeez, T.A., Mohany, Y.K., 2019b. A robust and efficient system to detect human faces based on facial features. *Asian J. Res. Comput. Sci.* 2 (4), 1–12.
- Braunegg, A., Chakraborty, A., Krundick, M., Lape, N., Leary, S., Manville, K., Merkhof, E., Strickhart, L., Walmer, M., 2020. Apricot: A dataset of physical adversarial attacks on object detection. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI* 16. Springer, pp. 35–50.
- Brown, T.B., Mané, D., Roy, A., Abadi, M., Gilmer, J., 2017. Adversarial patch. *arXiv preprint arXiv:1712.09665*.
- Carlini, N., Wagner, D., 2017. Towards evaluating the robustness of neural networks. In: *2017 IEEE Symposium on Security and Privacy (SP)*. Ieee, pp. 39–57.
- Chou, E., Tramer, F., Pellegrino, G., 2020. Sentinel: Detecting localized universal attacks against deep learning systems. In: *2020 IEEE Security and Privacy Workshops. SPW, IEEE*, pp. 48–54.
- Das, N., Shanbhogue, M., Chen, S.-T., Hohman, F., Chen, L., Kounavis, M.E., Chau, D.H., 2017. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*.
- Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J., 2018. Boosting adversarial attacks with momentum. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 9185–9193.
- Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD '96, AAAI Press*, pp. 226–231.
- Goodfellow, I.J., Shlens, J., Szegedy, C., 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- Hartigan, J.A., Wong, M.A., 1979. Algorithm AS 136: A K-means clustering algorithm. *J. R. Stat. Soc. Ser. C (Applied Statistics)* 28 (1), 100–108, URL <http://www.jstor.org/stable/2346830>.
- Hayes, J., 2018. On visible adversarial perturbations & digital watermarking. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. pp. 1597–1604.
- Ji, N., Feng, Y., Xie, H., Xiang, X., Liu, N., 2021. Adversarial yolo: Defense human detection patch attacks via detecting adversarial patches. *arXiv preprint arXiv:2103.08860*.
- Karmon, D., Zoran, D., Goldberg, Y., 2018. Lavan: Localized and visible adversarial noise. In: *International Conference on Machine Learning. PMLR*, pp. 2507–2515.
- Killeen, P.R., 2005. An alternative to null-hypothesis significance tests. *Psychol. Sci.* 16 (5), 345–353.
- Kurakin, A., Goodfellow, I.J., Bengio, S., 2018. Adversarial examples in the physical world. In: *Artificial Intelligence Safety and Security. Chapman and Hall/CRC*, pp. 99–112.
- Lee, M., Kolter, Z., 2019. On physical adversarial patches for object detection. *arXiv preprint arXiv:1906.11897*.
- Lehmann, R., 2013. 3 σ -rule for outlier detection from the viewpoint of geodetic adjustment. *J. Surv. Eng.* 139 (4), 157–165.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V* 13. Springer, pp. 740–755.
- Liu, J., Levine, A., Lau, C.P., Chellappa, R., Feizi, S., 2022. Segment and complete: Defending object detectors against adversarial patch attacks with robust patch detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 14973–14982.
- Liu, X., Yang, H., Liu, Z., Song, L., Li, H., Chen, Y., 2018. Dpatch: An adversarial patch attack on object detectors. *arXiv preprint arXiv:1806.02299*.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A., 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.
- Metzen, J.H., Finnie, N., Huttmacher, R., 2021. Meta adversarial training against universal patches. *arXiv preprint arXiv:2101.11453*.
- Metzen, J.H., Yatsura, M., 2021. Efficient certified defenses against patch attacks on image classifiers. *arXiv preprint arXiv:2102.04154*.
- Moosavi-Dezfooli, S.-M., Fawzi, A., Frossard, P., 2016. Deepfool: a simple and accurate method to fool deep neural networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 2574–2582.
- Naseer, M., Khan, S., Porikli, F., 2019. Local gradients smoothing: Defense against localized adversarial attacks. In: *2019 IEEE Winter Conference on Applications of Computer Vision. WACV, IEEE*, pp. 1300–1307.
- Oonishi, K., Nakai, T., Suzuki, D., 2022. Multiple remote adversarial patches: Generating patches based on diffusion models for object detection using CNNs. In: *NeurIPS ML Safety Workshop*.
- Raju, R.S., Lipasti, M., 2020. Blurnet: Defense by filtering the feature maps. In: *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, pp. 38–46.
- Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 779–788.
- Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* 28.
- Rezatofighi, H., Tsai, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S., 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 658–666.
- Rossolini, G., Nesti, F., D'Amico, G., Nair, S., Biondi, A., Buttazzo, G., 2023. On the real-world adversarial robustness of real-time semantic segmentation models for autonomous driving. *IEEE Trans. Neural Netw. Learn. Syst.*.
- Simonyan, K., Vedaldi, A., Zisserman, A., 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.
- Tarchoun, B., Ben Khalifa, A., Mahjoub, M.A., Abu-Ghazaleh, N., Alouani, I., 2023. Jedi: entropy-based localization and removal of adversarial patches. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 4087–4095.
- Teichmann, M., Weber, M., Zoellner, M., Cipolla, R., Urtasun, R., 2018. Multinet: Real-time joint semantic reasoning for autonomous driving. In: *2018 IEEE Intelligent Vehicles Symposium. IV, IEEE*, pp. 1013–1020.
- Thys, S., Van Ranst, W., Goedemé, T., 2019. Fooling automated surveillance cameras: adversarial patches to attack person detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.
- Wei, X., Pu, B., Lu, J., Wu, B., 2022a. Physically adversarial attacks and defenses in computer vision: A survey. 1, (2), *arXiv preprint arXiv:2211.01671*.
- Wei, H., Tang, H., Jia, X., Wang, Z., Yu, H., Li, Z., Satoh, S., Van Gool, L., Wang, Z., 2022b. Physical adversarial attack meets computer vision: A decade survey. *arXiv preprint arXiv:2209.15179*.
- Wu, T., Tong, L., Vorobeychik, Y., 2019. Defending against physically realizable attacks on image classification. *arXiv preprint arXiv:1909.09552*.
- Xiang, C., Bhagoji, A.N., Sehwal, V., Mittal, P., 2021. (PatchGuard): A provably robust defense against adversarial patches via small receptive fields and masking. In: *30th USENIX Security Symposium (USENIX Security 21)*. pp. 2237–2254.
- Xiang, C., Valtchanov, A., Mahloujifar, S., Mittal, P., 2023. ObjectSeeker: Certifiably robust object detection against patch hiding attacks via patch-agnostic masking. In: *2023 IEEE Symposium on Security and Privacy. SP, IEEE*, pp. 1329–1347.

- Xu, W., Evans, D., Qi, Y., 2017. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*.
- Xu, K., Xiao, Y., Zheng, Z., Cai, K., Nevatia, R., 2023. Patchzero: Defending against adversarial patch attacks by detecting and zeroing the patch. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. pp. 4632–4641.
- Zhang, Z., Yuan, B., McCoyd, M., Wagner, D., 2020. Clipped bagnet: Defending against sticker attacks with clipped bag-of-features. In: *2020 IEEE Security and Privacy Workshops. SPW, IEEE*, pp. 55–61.