SSIM-based Autoencoder Modeling to Defeat Adversarial Patch Attacks

Seungyeol Lee , Seongwoo Hong , Gwangyeol Kim , Jaecheol Ha *

Posted Date: 2 August 2024

doi: 10.20944/preprints202408.0064.v1

Keywords: Object detection; YOLO; Adversarial patch attack; Structural Similarity Index Measure; Autoencoder

Article

SSIM-Based Autoencoder Modeling to Defeat Adversarial Patch Attacks

Seungyeol Lee [1], Seongwoo Hong [1], Gwangyeol Kim [2] and Jaecheol Ha [1,*]

[1] Dept. of Information Security, Hoseo University, Asan-Si, ChungNam-Do, Rep. of Korea; stl990726@naver.com (S.L.); hshsw5660@gmail.com (S.H.); jcha@hoseo.edu (J.H.)
[2] Sinsiway Inc., Songpa-Gu, Seoul-Si, Rep. of Korea; jeffkim@sinsiway.com
* Correspondence: jcha@hoseo.edu; Tel.: +82-41-540-5991

Abstract: Object detection systems are widely used in various fields such as autonomous vehicles and facial recognition. In particular, object detection using deep learning networks enables real-time processing in low-performance edge devices, and can maintain high detection rates. However, edge devices that operate far from administrators are vulnerable to various physical attacks by malicious adversaries. In this paper, we implement a function for detecting traffic signs by using three versions of You Only Look Once (YOLO) and Faster-RCNN, which can be adopted by edge devices of autonomous vehicles. Then, assuming the role of a malicious attacker, we executed adversarial patch attacks with Adv-Patch and Dpatch. As a result of experimenting with misdetection of traffic stop signs using Adv-Patch and Dpatch, we confirmed the attacks can succeed with a high probability. To defeat these attacks, we propose an image reconstruction method using an autoencoder and the Structural Similarity Index Measure (SSIM). We confirm that the proposed method can sufficiently defend against an attack, attaining an AP above 91.46% even when two adversarial attacks are launched.

Keywords: object detection; YOLO; adversarial patch attack; Structural Similarity Index Measure; autoencoder

1. Introduction

Recently, object detection systems using deep learning networks have been widely used in autonomous vehicles, for face recognition, and in home networks. In particular, object detection using a deep learning network running on low-performance edge devices has the advantage of a real-time processing capability. However, edge devices where deep learning networks operate far from administrators are vulnerable to adversarial patch attacks. These attacks can compromise object detection systems that use deep learning networks, leading to accidents through misdetection in autonomous vehicles and facial recognition.

Brown et al. showed that an image classification model can fail with high probability if a patch is simply attached to an image, unlike existing evasion attacks in white-box and black-box environments [1]. An adversarial patch attack targeting a traffic sign classifier was conducted by Lengyel et al. [2]. By launching adversarial patch attacks of various scales, they succeeded in classifying stop signs into different classes.

Since adversarial attacks on classifiers have been attempted, advanced attacks on object detectors have also been proposed. Thys et al. executed adversarial patch (Adv-patch) attacks in both digital and real world environments [3]. They showed that detectors could not detect people after an adversarial patch was attached to their bodies. Liu et al. proposed more advanced than a simple adversarial patch, so-called Dpatch [4]. Dpatch uses the peculiarities of bounding box regression and object classification in an object detection system, and had a higher attack success rate than adversarial patches placed in various locations in the image.

As a research in advanced patch generation, Eykholt et al. launched an adversarial patch attack targeting a traffic sign classifier. The attack's success rate was evaluated after applying masking to the adversarial patch and setting up an environment that did not obscure the traffic signs. Hu et al. used a generative adversarial network to create an adversarial patch as natural as those in the real world while maintaining high attack performance [6].

 doi:10.20944/preprints202408.0064.v1

In order to defeat these adversarial patch attacks, Nasser et al. demonstrated a Local Gradient Smoothing (LGS) method of responding to adversarial patch attacks by smoothing the high-frequency parts of images with an adversarial patch [7]. Furthermore, a method to counteract to adversarial attacks by using an autoencoder was proposed by Yin et al. [8]. They countered attacks after removing noise from images with the autoencoder.

In this paper, we first implement a function for detecting traffic objects such as stop and speed limit signs. We adopted the You Only Look Once (YOLO) family and Faster-RCNN as deep learning object detection algorithms [9]. Then, assuming a malicious attacker, we launched Adv-Patch and Dpatch attacks targeting our object detection algorithms. As a result of performing an original Adv-Patch attacks and Dpatch attack, it was confirmed that the attack was successful with an AP of about 22.16%.

To counter these attacks, we propose an image reconstruction method using an autoencoder and the Structural Similarity Index Measure(SSIM). The proposed method enables normal object detection by sending patched images to an autoencoder and reconstructing them into clean images. Additionally, SSIM was used during the autoencoder training process to improve the quality of the reconstructed images.

This paper is structured as follows. Section 2 describes image reconstruction using autoencoders, adversarial patch attacks, Dpatch and SSIM. Section 3 explains adversarial patch attacks on object detectors. Section 4 explains image reconstruction using the proposed SSIM-based autoencoder. Section 5 describes the experimental environment, evaluates performance against adversarial patch attacks and Dpatch, and evaluates performance by applying the proposed countermeasures. Section 6 concludes the paper.

2. Background

2.1. Adversarial Patch Attacks

Deep learning networks are used in many artificial intelligence industries, but are exposed to several types of cyber attack. In particular, they are vulnerable to evasion attacks that add noise to a network input image causing the deep learning network to malfunction. Representative adversarial evasion attacks include FGSM(Fast Gradient Sign Method) [10], PGD(Projected Gradient Descent) [11], CW(Carlini and Wagner) [12], DeepFool [13] and JSMA(Jacobian base Saliency Map Attack) [14].

However, these types of attack are difficult to try in the real world because they require the attacker to directly access the deep learning model. On the other hand, the adversarial patch attack attaches malicious patches to images without accessing the deep learning network, also causing malfunctions. Eq. (1) is the formula for generating an adversarial patch [1–3]:

$$\hat{p} = \arg\max \; E_{x\sim X, t\sim T, l\sim L}[logPr(\hat{y} \mid A(p, x, l, t))] \tag{1}$$

Here, X represents the input data. T is the distribution of transformations of the image, L denotes the distribution of patch locations within the image, $\hat{p}$ symbolizes the final patch, and $\hat{y}$ represents the intended class.

An attacker can cause a deep learning network to malfunction simply by attaching a generated adversarial patch to an image. Figure 1 shows a successful example of an adversarial patch attack, where an image classification model that was supposed to predict a banana ended up predicting a toaster due to an adversarial patch.

 doi:10.20944/preprints202408.0064.v1

Figure 1. Successfully fooling a classification system by using an adversarial patch

2.2. Dpatch

Dpatch is a patch-based adversarial attack technique proposed to fool object detection models such as YOLO and Faster-RCNN [4]. Unlike traditional adversarial patch, Dpatch is designed in a small form that can be physically attached to objects, causing object detection models to misclassify or malfunction. The following Eq. (2) is a formula for generating a Dpatch [4].

$$\delta^* = argmax_\delta L(f(I + M \cdot \delta), y) \tag{2}$$

Here, I is the original image, δ refers to the patch, and M is the mask indicating the location of the patch. Additionally, L represents the loss function, y is the actual label.

In an attack using Dpatch, the patch is generated using gradient-based optimization. Dpatch is generated using gradient-based optimization. When the generated Dpatch is attached to an object, it significantly reduces the performance of the deep learning network, which will eventually fail to correctly recognize objects like traffic signals.

2.3. Autoencoders

An autoencoder is an unsupervised learning model and neural network that compresses and reconstructs input data. The autoencoder consists of an encoder, a latent space, and a decoder, as shown in Figure 2. The encoder maps the input data to the latent space, and the decoder takes the compressed data mapped to the latent space and reconstructs the original data. Using these characteristics, autoencoders are used for data compression and image noise removal [15–17].

 doi:10.20944/preprints202408.0064.v1

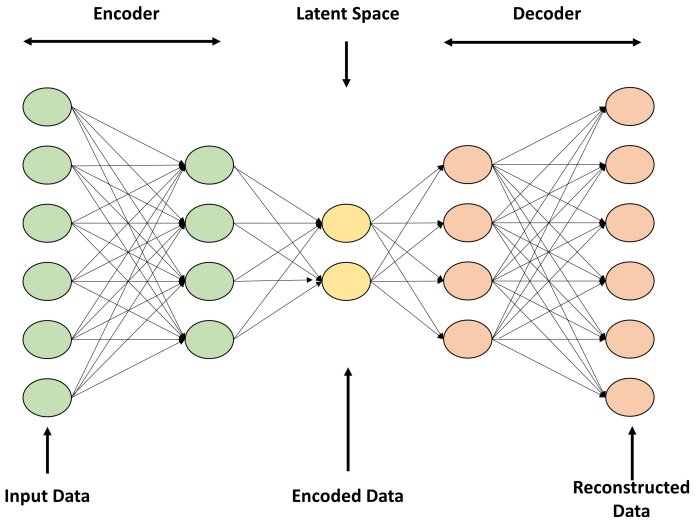

Figure 2. Structure of an autoencoder model

2.4. Structural Similarity Index Measure

SSIM is a method for assessing digital images and videos [18]. Unlike traditional methods to evaluate images, such as peak signal-to-noise ratio (PSNR), mean squared error (MSE), and Mean Absolute Error (MAE), SSIM evaluates image quality based on three main aspects: luminance, contrast, and structure.

Additionally, images reconstructed using SSIM may have higher super-resolution, which can be helpful in training image generation networks. Large SSIM values indicate more similarity to the original image, whereas small values indicate less similarity. Therefore, if you use SSIM during model training for image generation, you can reconstruct high-quality images. Eq. (3) is the formula for calculating a SSIM value [18]:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \tag{3}$$

Here, x represents the original image, and y represents the generated image, while l, c and s are luminance, contrast, and structure, respectively. The correlation coefficient between x and y is calculated using l, c and s to evaluate similarity in an image. The α, β and γ are expressed in terms of importance of l, c and s respectively.

3. Adversarial Patch Attacks on Object Detectors

We executed adversarial attacks using the original Adv-Patch and Dpatch targeting YOLO versions that can detect traffic stop signs. Adv-Patch generation [3] from the original paper is based on decreasing object score loss, total variation loss, and non-printability score loss as predicted by the object detector.

First, the object score loss represents the score from an object detector that predicts an object in an image. Next, the total variation loss smooths the adversarial patch, reducing unnecessary noise in it. The formula for total variation loss is Eq. (4):

$$L_{tv} = \sum_{i,j} \sqrt{((p_{i,j} - p_{i+1,j})^2 + (p_{i,j} - p_{i,j+1})^2} \tag{4}$$

 doi:10.20944/preprints202408.0064.v1

The non-printability score loss is a value that guarantees the attack success rate when printing patches. Minimizing these losses allows adversarial patches to be used effectively in the real world. The formula for non-printability score loss is Eq. (5):

$$L_{nps} = \sum_{p_{patch} \in p} \min_{c_{print} \in C} \left| p_{patch} - c_{print} \right| \tag{5}$$

Our adversarial patch was generated by minimizing object score loss for traffic signs to cause misdetection by YOLO along with reducing total variation loss and non-printability score loss, resulting in the final adversarial patch. Additionally, the adversarial patch was resized and attached using a masking technique [19,20] to avoid obscuring the stop signs.

As shown in Figure 3, YOLO accurately detects traffic signs in clean images. We apply the detector first on images without a patch (row 1), with a masked patch (row 2), and with our generated patch (row 3). In most attacks our patch is able to successfully hide traffic signs from the detector. However, we can catch the adversarial patch when the patch is not aligned to the center of the traffic sign. Therefore, we can explain the fact that, for optimized attack success, the patch should be positioned in the center of the traffic sign determined by the bounding box. As a result, we were able to generate adversarial patches according to the method described above [1,3], and confirmed that these patch attacks can be successful with a high probability.

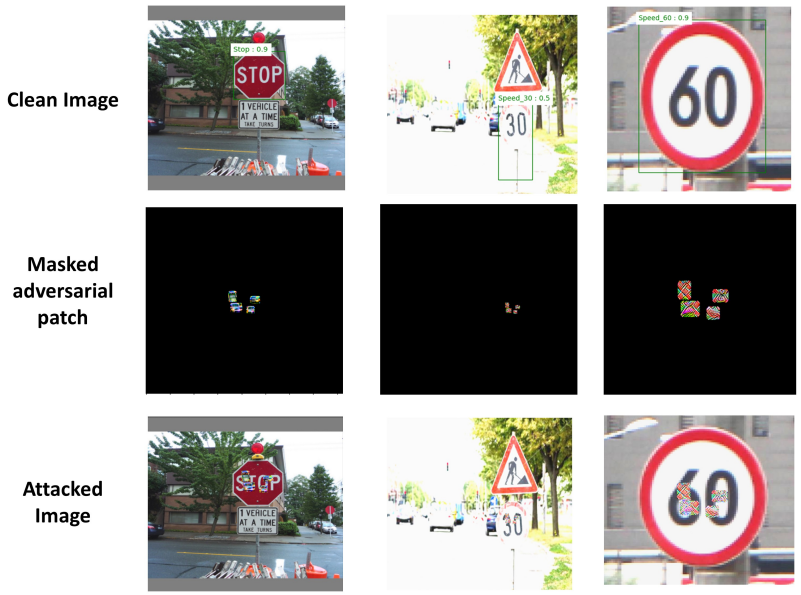

Figure 3. Adversarial patch attacks on traffic signs

4. SSIM-Based Autoencoder Modeling

We aim to develop a countermeasure against adversarial patch attacks that cause failures in object detectors. We propose a new competitive learning architecture using an SSIM-based autoencoder to defeat the adversarial patch attacks, as shown in Figure 4. This architecture consists of three components, that is, a fixed-weight object detector such as YOLO or Faster-RCNN, an SSIM-based autoencoder, and an adversarial patch generator. Here, the SSIM-based autoencoder created through competitive learning reconstructs an image with an adversarial patch into a clean image so that object detection is performed normally. This competitive learning allows construction of a robust autoencoder that can withstand various adversarial patch attacks. Consequently, these two modules, SSIM-based autoencoder and patch generator, are designed to improve performance through mutual competition.

Furthermore, we use SSIM method as an additional loss function to improve the quality of the generated clean image when the autoencoder reconstructs the adversarial patch image. By combining

 doi:10.20944/preprints202408.0064.v1

the SSIM method with an autoencoder, the final output image reconstructed through the autoencoder can be very close to a clean image.

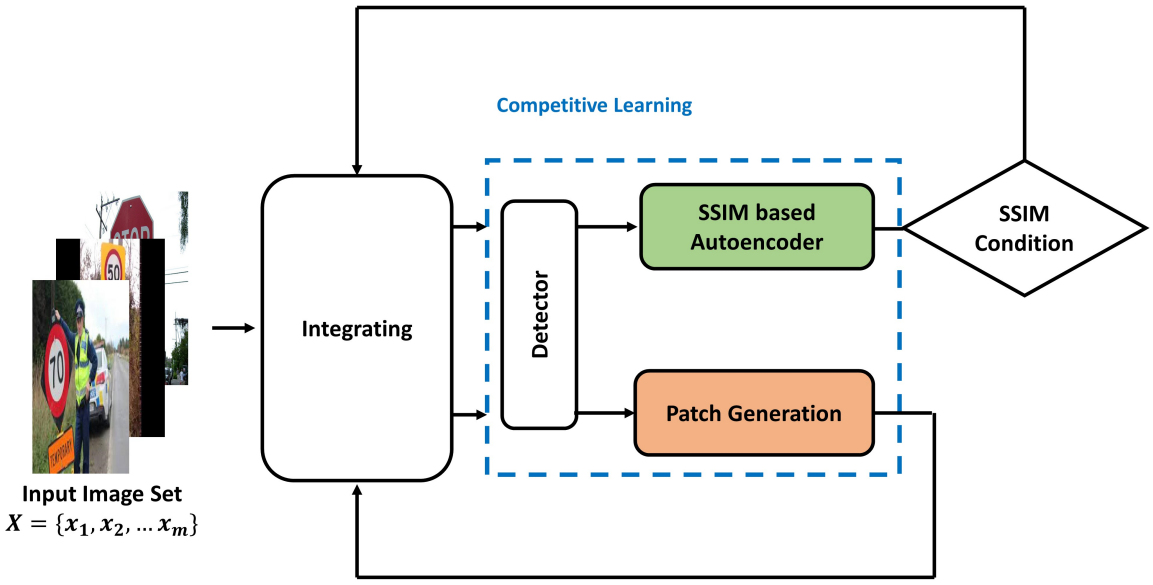

Figure 4. Adversarial patch attacks on traffic signs

The SSIM-based autoencoder generation module focuses on preventing traffic sign detection systems from malfunctioning due to adversarial patches. SSIM is used to evaluate the structural similarity of images, and based on this, the autoencoder reconstructs the input image into a normal image to make it similar to the original image. Generating an autoencoder output similar to the input image means minimizing the impact of adversarial patch attacks.

The optimization process of the autoencoder is accomplished through the following objective function.

$$\Delta AE = \nabla_{AE} L_{AE}\Big(x, g\Big(f\Big(L_{patch}(x,p)\Big)\Big)\Big) + 1 - SSIM\Big(x, g\Big(f\Big(L_{patch}(x,p)\Big)\Big)\Big) \tag{6}$$

Here, x represents the input image. And f and g is encoder and decoder respectively. The L_{patch} denotes a function that attaches a patch p to x.

The processes flow for modeling the SSIM-based autoencoder are shown in below Algorithm 1. The adversarial patch generation module aims to reduce the object score of traffic sign detection by optimizing the adversarial patch p. This process is described in Algorithm 1 as $\Delta p = \nabla_p L_{patch}(x,p)$. The patch generation module updates patch p for each input data point x, and calculates the gradient of patch p. In this process, a patch that interferes with traffic sign detection is created by ensuring that the detector′s object score is 0.

Computation of Line 6 in this algorithm, $\Delta p = \nabla_p L_{patch}(x,p)$), is the process of updating the adversarial patch p in the initial learning stage, and Line 8 in the later stages, $\Delta p = \nabla_p L_{patch}(x,p,f,g)$), is used to update th,e patch more powerfully. Here, competitive learning parameters E, K_p and K_{AE} from Algorithm 1 to train the SSIM-based autoencoder were set to 3, 5, and 5, respectively.

 doi:10.20944/preprints202408.0064.v1

Algorithm 1 SSIM-based Autoencoder Modeling

Input: Data point X
Output: AE

1: Initialize $p_0 \leftarrow$ Gaussian Noise
2: **repeat** ▷ Iterative learning process
3: **for** each data point $x \in X$ **do** ▷ Process of patch generator
4: **for** $k = 1, 2, \ldots, K_p$ **do**
5: **if** $E == 1$ **then**
6: $\Delta p = \nabla_p L_{patch}(x, p)$
7: **else**
8: $\Delta p = \nabla_p L_{patch}(x, p, f, g)$
9: **end if**
10: **end for**
11: **end for**
12: **for** each data point $x \in X$ **do** ▷ Process of autoencoder
13: **for** $k = 1, 2, \ldots, K_{AE}$ **do**
14: $\Delta AE = \nabla_{AE} L_{AE}\left(x, g\left(f\left(L_{patch}(x, p)\right)\right)\right) + 1 - SSIM\left(x, g\left(f\left(L_{patch}(x, p)\right)\right)\right)$
15: **end for**
16: **end for**
17: **until** training epoch E

These two modules, SSIM-based autoencoder and patch generator, develop complementary to each other through competitive learning. The SSIM-based autoencoder tries to maintain the original detection performance, while the adversarial patch generation module tries to interfere with it. In this process, the autoencoder iteratively learns to minimize the impact of adversarial patches by reconstructing images containing adversarial patches into normal images.

The SSIM-based autoencoder provides generalized defense performance against a variety of attacks as well as specific types of adversarial patches. The SSIM described in Line 14 of Algorithm 1 evaluates the structural similarity of two images, normal image and reconstructed one generated by autoencoder. Here, the SSIM value is always less than 1, with higher values indicating more similarity. This competitive learning approach can minimize the effect of adversarial patches and significantly improve the stability of traffic sign detection system.

5. The Countermeasure Against Adversarial Patch Attacks

5.1. Traffic Sign Detection in Normal Detector

We used the YOLOv3, YOLOv5, and YOLOv8, and Faster-RCNN object detection models for the adversarial patch attacks targeting traffic signs after pre-training them on the COCO dataset [21]. Before proceeding with the experiment, 958 images containing stop signs were pulled. From them, 674 images were used for competitive training of the SSIM-based autoencoder. After modeling an SSIM-based autoencoder, we evaluate the performance of object detection models, YOLOv3, YOLOv5, and YOLOv8, and Faster-RCNN. Here, we use 284 images with stop signs from the COCO dataset and 300 images with stop signs from the LISA dataset [22]. In addition, after learning YOLO family and Faster-RCNN from the Traffic Sign dataset [23] in the similar wat, we also evaluate the detection performance on 158 images containing the speed limit class.

To calculate performance by the YOLO family and Faster-RCNN, the minimum detection threshold was set to 0.5. The IOU (Intersection Over Union) was set to 0.65, and images were sized at 416×416. For the attacks that disabled object detection, we used the original Adv-Patch and Dpatch, and used Adam to optimize patch generation. After applying masking to each attack with an initial learning rate at 0.03, we attached the patches to traffic signs.

Furthermore, we evaluated the detection model performance by varying the brightness and contrast of the image to create an environment similar to real traffic situations. The brightness value is randomized from 0 to 3, where 1 means no effect, 0 means darkening, and closer to 3 means brighter. The contrast value is a randomized value from 0.7 to 1.3, where 1 means no impact, 0.7 indicates decreased contrast, and 1.3 indicates increased contrast. Table 1 shows the object detection performance of the YOLO model on clean images from the COCO, the LISA, and Traffic Sign dataset. As shown in

 doi:10.20944/preprints202408.0064.v1

the Table 1, object detection was achieved with a maximum mean Average Precision (mAP) of 98.88% and a minimum of 96.71% using the YOLOv8 detector.

Table 1. Object detection using COCO, LISA and Traffic Sign dataset

Dataset	YOLOv3	YOLOv5	YOLOv8	Faster-RCNN
COCO Dataset(Stop signs)	97.64%	98.31%	98.03%	96.05%
LISA Dataset(Stop signs)	94.26%	94.05%	96.71%	98.55%
Traffic Sign Dataset(Stop & speed signs)	98.80%	98.76%	98.88%	96.16%

5.2. Results on Adversarial Patch Attacks

We have assumed the use of an adversarial patch in a real world situation. This means that many factors affect the appearance of a patch when performing a patch attack. The lighting can change, the size of the patch with respect to the traffic sign can change, the input sensor may add noise or blur the patch slightly. To take this real environment into account as much as possible, we do the following random transformations on the patch before applying it to the original image. (1) The patch is scaled up and down randomly. (2) Random noise is added on the patch. (3) The brightness and contrast of the patch is changed randomly.

Table 2 shows the performance evaluation of the YOLO family and Faster-RCNN on adversarial patch attacks for the COCO, LISA, and Traffic Sign dataset. On three datasets, the minimum mAP scores of YOLO family was 22.21% in case of Adv-patch attack on LISA dataset. Additionally, Faster-RCNN showed mAP scores of 49.05% for Adv-patch attack and 22.16% for Dpatch one. As a result of performing two adversarial attacks on all datasets, it was confirmed that most mAP scores decreased significantly, making it impossible to detect objects. Specifically, we found that adversarial attacks against YOLOv8, which showed maximum detection rate when no attack was performed, dropped to 26.55% in the worst case.

Table 2. Performance on detection models applied adversarial patch attacks

Dataset	Attack Method	YOLOv3	YOLOv5	YOLOv8	Faster-RCNN
COCO Dataset	Adv_Patch[3]	68.01%	52.36%	73.48%	53.38%
	Dpatch[4]	42.09%	37.56%	65.45%	41.28%
LISA Dataset	Adv_Patch[3]	38.20%	22.21%	49.16%	49.05%
	Dpatch[4]	30.94%	40.37%	40.45%	22.16%
Traffic Sign Dataset	Adv_Patch[3]	41.34%	32.96%	26.55%	54.36%
	Dpatch[4]	23.98%	23.52%	39.86%	42.96%

5.3. Object Detection Using Proposed SSIM-Based Autoencoder

In order to defeat these adversarial attacks, we propose an image reconstruction method using a SSIM-based autoencoder. We made the autoencoder model to reconstruct stop sign images containing adversarial patches into clean images and to improve the quality of the reconstructed images through SSIM. Thereafter, the object detector could detect image input from the SSIM-based autoencoder normally. We improved the robustness of the SSIM-based autoencoder by using competitive learning to reconstruct images with different adversarial patches. Figure 5 shows the SSIM-based autoencoder architecture to counteract these adversarial patch attacks.

 doi:10.20944/preprints202408.0064.v1

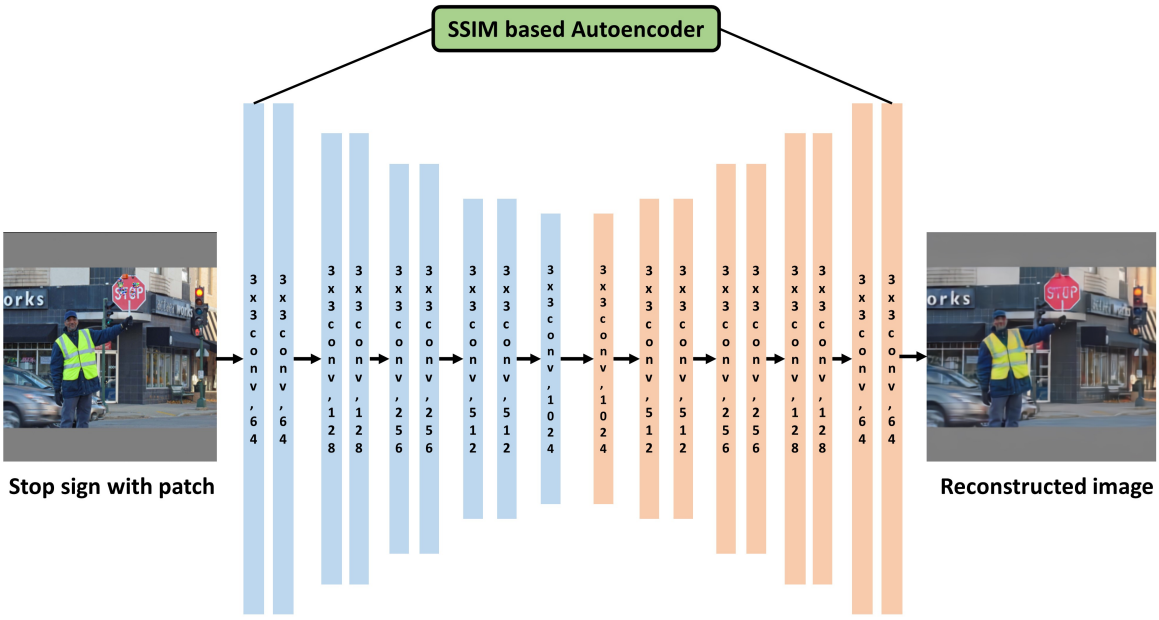

Figure 5. The SSIM-based autoencoder architecture

Examples of detection using the SSIM-based autoencoder on images of traffic signs are shown in Figure 6. The top images are clean, and the middle images have patches generated by an attack. The bottom images were reconstructed by the SSIM-based autoencoder.

Table 3 shows the experimental result for three dataset and two types of detection models. According to Table 3, for clean images without any adversarial attacks, the mAP scores of the YOLO family and Faster-RCNN reach up to 98.85%.

When performing adversarial patch attacks on the YOLO family, it was observed that the mAP score decreased to a minimum of 22.21% for the Adv-Patch attack and 23.52% for the Dpatch attack. In Table 3, the proposed SSIM-based autoencoder improved the mAP score by 89.91% and 90.51% respectively, even though Adv-Patch and Dpatch attacks were performed. Additionally, we obtained similar results by using the SSIM-based autoencoder on Faster-RCNN. As shown in the Table 3, the mAP score improved by 92.75% and 90.79% for the Adv-Patch and Dpatch attacks, respectively.

The proposed SSIM-based autoencoder can maintain comparable performance to the detector in the absence of adversarial attacks without any meaningful degradation. In particular, the Adv-Patch attack against YOLOv8 was reduced to 26.55% in the worst case. However, using the autoencoder proposed in this paper, we were able to detect 91.56% of the attacks. This is slightly lower than the performance without the attack, but it is the best performance among the detectors. In addition, the detection rate of YOLOv8 detector was 4.54% and 7.31% higher for adversarial attacks compared to Faster-RCNN, respectively, confirming that it is an attack-resistant detector. Consequently, experimental results demonstrated that the proposed autoencoder can significantly defend against adversarial patch attacks and be applied to both single-stage and two-stage detection networks.

 doi:10.20944/preprints202408.0064.v1

Table 3. Evaluation of AP performance of SSIM-based autoencoder with competitive learning

Dataset	Attack Method	YOLOv3			YOLOv5			YOLOv8			Faster-RCNN		
		Ours	LGS[7]	AE[8]	Ours	LGS[7]	AE[8]	Ours	LGS[7]	AE[8]	Ours	LGS[7]	AE[8]
COCO Dataset	No attack	95.15%	96.92%	85.80%	95.00%	98.27%	85.68%	97.68%	95.99%	87.41%	88.60%	94.63%	79.65%
	Adv-patch[3]	93.06%	79.31%	79.31%	91.41%	67.63%	75.23%	94.93%	77.79%	80.77%	86.23%	76.31%	75.50%
	Dpatch[4]	93.83%	65.76%	81.80%	91.71%	54.00%	77.95%	94.98%	88.40%	82.22%	85.08%	69.87%	71.88%
LISA Dataset	No attack	93.63%	94.24%	89.73%	94.15%	92.96%	90.32%	96.65%	95.58%	93.82%	95.07%	95.29%	90.95%
	Adv-patch[3]	91.50%	81.38%	86.58%	89.91%	78.99%	49.06%	92.88%	78.34%	90.14%	92.75%	66.65%	88.4%
	Dpatch[4]	94.09%	45.28%	85.21%	90.31%	59.03%	78.74%	95.19%	88.93%	87.19%	90.79%	65.85%	82.65%
Traffic Sign Dataset	No attack	98.24%	98.10%	83.66%	97.62%	97.60%	80.11%	98.85%	97.27%	80.18%	96.16%	96.50%	81.81%
	Adv-patch[3]	88.25%	70.85%	61.97%	87.47%	66.86%	62.45%	91.56%	71.69%	63.54%	87.02%	52.02%	43.64%
	Dpatch[4]	88.19%	33.02%	54.04%	90.51%	56.46%	36.16%	91.46%	74.98%	67.63%	84.15%	43.95%	43.32%

 doi:10.20944/preprints202408.0064.v1

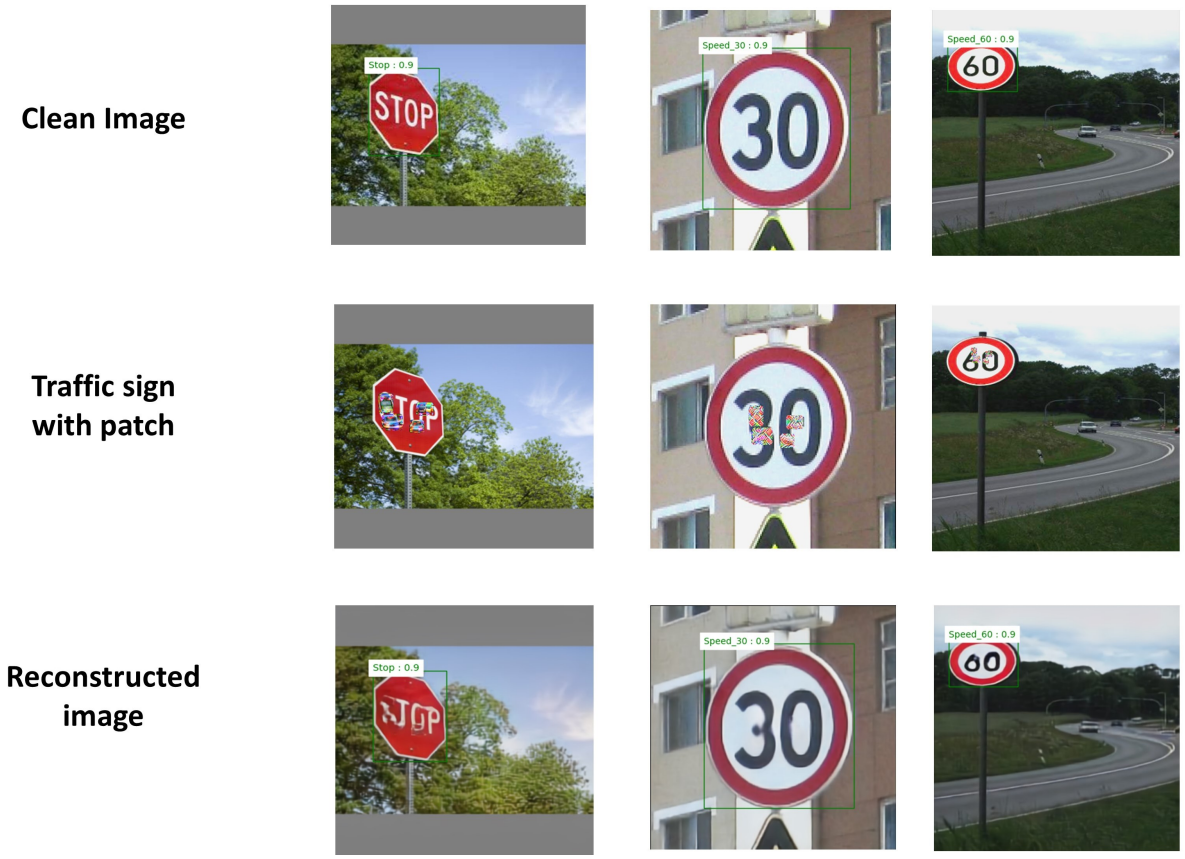

Figure 6. Traffic sign images reconstructed with the SSIM-based autoencoder

6. Conclusion

Recently, object detection based on deep learning has been widely used in autonomous vehicles, facial recognition, and smart factories. However, these object detection systems are vulnerable to adversarial patch attacks that can be carried out without access to the detector. Even a simple patch attack can cause an object detection system to miss an object, which can lead to a serious accident.

In this paper, we implemented traffic stop sign detection by using three versions of the YOLO family and Faster-RCNN in edge devices, and we launched adversarial patch attacks using Adv-Patch and Dpatch. As a result of the experiment, it was confirmed that the YOLO family mAP decreased by 22.21% with the original Adv-patch attack and 23.52% with the Dpatch attack. Additionally, the mAP in Faster-RCNN is decreased by 49.05% for Adv-Patch attack and 22.16% for Dpatch one respectively.

By applying SSIM in an autoencoder, we attained a countermeasure from a deep learning model that reconstructs an attacked image into a clean image and improved the detection rate of traffic signs. Our experiment on YOLOv8 detector confirmed that the proposed SSIM-based autoencoder was able to restore the mAP of the object detector to more than 91.46%. This confirmed that the proposed deep learning model is effective against adversarial patch attacks compared to image reconstruction methods such as the original autoencoder or LGS.

Author Contributions: Conceptualization, methodology, software, S.L. and J.H. ; validation, S.H., G.K. and J.H.; writing—original draft preparation, S.L. and S.H.; writing—review and editing, S.L. and J.H.; All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a project for Smart Manufacturing Innovation R&D funded Korea Ministry of SMEs and Startups in 2022. (Project No. RS-2022-00140535).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

 doi:10.20944/preprints202408.0064.v1

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Brown, T.B.; Maném, D.; Roy, A.; Abadi, M.; Gilmer, J. Adversarial patch. In Proceedings of the NIPS 2017 Workshop on Machine Learning and Computer Security, 2017.
2. Lengyel, H.; Remeli, V.; Szalay, Z. Easily deployed stickers could disrupt traffic sign recognition. *Perner's Contacts* **2019**, *19*, 156-163.
3. Thys, S.; Van Ranst, W.; Goedemé, T. Fooling automated surveillance cameras: adversarial patches to attack person detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2019, pp. 0-0.
4. Liu, X.; Yang, H.; Yang, L.Z.; Song, L.; Li, H.; Chen, Y. DPatch: An adversarial patch attack on object detectors. In Proceedings of the AAAI Workshops, 2018.
5. Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; et al. Robust physical-world attacks on deep learning visual classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1625-1634.
6. Hu, Y.C.T.; Kung, B.H.; Tan, D.S.; Chen, J.C.; Hua, K.L.; Cheng, W.H. Naturalistic physical adversarial patch for object detectors. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 7848-7857.
7. Naseer, M.; Khan, S.; Porikli, F. Local gradients smoothing: Defense against localized adversarial attacks. In Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), 2019, pp. 1300-1307.
8. Yin, S.L.; Zhang, X.L.; Zuo, L.Y. Defending against adversarial attacks using spherical sampling-based variational auto-encoder. *Neurocomputing* **2022**, 1-10.
9. Tsuruoka, G.; Sato, T.; Chen, Q.A.; Nomoto, K.; Tanaka, Y.; Kobayashi, R.; et al. WIP: Adversarial Retroreflective Patches: A Novel Stealthy Attack on Traffic Sign Recognition at Night. In Proceedings of the Symposium on Vehicle Security and Privacy, 2024.
10. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. In Proceedings of the International Conference on Learning Representations (ICLR'15), 2015, pp. 1-11.
11. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. In Proceedings of the International Conference on Learning Representations (ICLR'18), 2017, pp. 1-11.
12. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), 2017, pp. 39-57.
13. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2574-2582.
14. Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy, 2016, pp. 372-387.
15. Lin, X.; Li, Y.; Hsiao, J.; Ho, C.; Kong, Y. Catch missing details: Image reconstruction with frequency augmented variational autoencoder. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 1736-1745.
16. Suganuma, M.; Ozay, M.; Okatani, T. Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In Proceedings of the International Conference on Machine Learning, 2018, pp. 4771-4780.
17. Mao, X.J.; Shen, C.; Yang, Y.B. Image restoration using convolutional auto-encoders with symmetric skip connections. In Proceedings of the Neural Information Processing Systems, 2016, pp. 1-17.
18. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* **2004**, *13*, 600-612.
19. Song, D.; Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; et al. Physical adversarial examples for object detectors. In Proceedings of the 12th USENIX Workshop on Offensive Technologies (WOOT 18), 2018.

 doi:10.20944/preprints202408.0064.v1

20. Pavlitska, S.; Lambing, N.; Zöllner, J.M. Adversarial attacks on traffic sign recognition: A survey. In Proceedings of the 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 2023, pp. 1-6.
21. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; et al. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Part V, Springer, 2014, pp. 740-755.
22. Mogelmose, A.; Trivedi, M.M.; Moeslund, T.B. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems* **2012**, *13*, 1484-1497.
23. Chãn, H.L. TrafficSign detection Dataset [Open Source Dataset]. Available online: https://universe.roboflow.com/chan-hungluu/trafficsigndetection (2022).