information

MDPI

*Article*

# HE-DMDeception: Adversarial Attack Network for 3D Object Detection Based on Human Eye and Deep Learning Model Deception

**Pin Zhang [1,*], Yawen Liu [1], Heng Liu [1], Yichao Teng [2], Jiazheng Ni [3], Zhuansun Xiaobo [3] and Jiajia Wang [1]**

[1] Army Engineering University of PLA, Nanjing 210007, China; wangjiajia@aeu.edu.cn (J.W.)
[2] National University of Defense Technology, Changsha 410022, China
[3] Naval Research Institute of PLA, Beijing 100161, China
* Correspondence: pinzhangthree@aeu.edu.cn

**Abstract**

This paper presents HE-DMDeception, a novel adversarial attack network that integrates human visual deception with deep model deception to enhance the security of 3D object detection. Existing patch-based and camouflage methods can mislead deep learning models but struggle to generate visually imperceptible, high-quality textures. Our framework employs a CycleGAN-based camouflage network to generate highly camouflaged background textures, while a dedicated deception module disrupts non-maximum suppression (NMS) and attention mechanisms through optimized constraints that balance attack efficacy and visual fidelity. To overcome the scarcity of annotated vehicle data, an image segmentation module based on the pre-trained Segment Anything (SAM) model is introduced, leveraging a two-stage training strategy combining semi-supervised self-training and supervised fine-tuning. Experimental results show that the minimum P@0.5 values (50%, 55%, 20%, 25%, 25%) were achieved by HE-DMDeception across You Only Look Once version 8 (YOLOv8), Real-Time Detection Transformer (RT-DETR), Fast Region-based Convolutional Neural Network (Faster-RCNN), Single Shot MultiBox Detector (SSD), and MaskRegion-based Convolutional Neural Network (Mask RCNN) detection models, while maintaining high visual consistency with the original camouflage. These findings demonstrate the robustness and practicality of HE-DMDeception, offering new insights into 3D object detection adversarial attacks.

**Keywords:** adversarial attack; CycleGAN; image segmentation

check for updates

Academic Editor: Haris Mouratidis

Received: 28 August 2025
Revised: 19 September 2025
Accepted: 3 October 2025
Published: 7 October 2025

**Citation:** Zhang, P.; Liu, Y.; Liu, H.; Teng, Y.; Ni, J.; Xiaobo, Z.; Wang, J. HE-DMDeception: Adversarial Attack Network for 3D Object Detection Based on Human Eye and Deep Learning Model Deception. *Information* **2025**, *16*, 867. https://doi.org/10.3390/info16100867

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

The rapid advancement of artificial intelligence (AI) has established 3D object detection as a critical technology in areas such as autonomous driving, robotics, and virtual reality. Despite substantial performance improvements achieved by deep learning models, these models remain susceptible to adversarial attacks. Such attacks involve introducing imperceptible perturbations to the input data, resulting in incorrect predictions. These subtle alterations can cause misclassification or even prevent object detection, significantly affecting decision-making processes, despite being nearly imperceptible to the human eye.

Currently, adversarial sample generation methods for 3D object detection are classified into patch-based and camouflage-based approaches. The patch-based method generates attacks by adding adversarial patches to the target object. The central idea is to concentrate noise within a localized patch area without applying disturbance constraints. However,

this approach is limited to the local region of the object and is easily affected by factors such as occlusion [1]. In real-world applications, the patch is typically placed on the surface of a planar object, in front of the object, or in the background.

Camouflage-based methods directly modify the shape, texture, color, and other attributes of the target object. In 3D object camouflage, a differentiable neural renderer is employed to optimize the texture or shape of the 3D vehicle [2,3]. For instance, altering the texture patterns on the vehicle's surface introduces visual changes that affect the detection system's ability to recognize the vehicle. Additionally, the adversarial model continuously refines the camouflage and applies it repeatedly to the surface of the target object. A physically non-differentiable renderer is used to map it onto the vehicle's surface [4,5]. In contrast, some studies employ differentiable neural renderers to directly optimize the texture of the 3D vehicle. Since most real-world target objects are three-dimensional with non-planar surfaces, camouflage-based methods must account for this complex geo-metric structure. When camouflaging a 3D vehicle, it is crucial to ensure that the camouflage adapts to the vehicle's curved surface and effectively interferes with detection from various viewpoints. Through techniques such as differentiable neural renderers, the camouflage texture is accurately mapped to the 3D object's surface, ensuring its effectiveness from all angles and preventing failure due to viewpoint changes.

However, both of the aforementioned methods lack robustness in handling multi-view scenarios and partially occluded objects, and they also face limitations in generating high-quality textures [6]. On one hand, patch-based methods are prone to having patches adhere to planar objects, which makes them unsuitable for attacking target detectors of 3D models. On the other hand, prior camouflage-based methods apply adversarial camouflage only to specific areas of the 3D vehicle model (e.g., the roof or side doors), significantly reducing the attack's effectiveness from multiple viewpoints. Once the camouflaged region becomes obscured, the attack's effectiveness declines sharply. Table 1 provides the advantages and limitations of the aforementioned methods.

To address challenges such as insufficient robustness in multi-view and partially occluded objects, poor texture quality, and the limited effectiveness of multi-view attacks, this paper proposes an improved full-coverage method by designing a novel adversarial texture generation and optimization network. This network enhances the quality and stability of generated textures, ensuring that the adversarial textures are visually imperceptible while effectively misleading deep learning models. This dual objective of deceiving both human vision and detection models increases the stealthiness and practicality of adversarial examples in real-world applications.

To achieve this, the proposed network combines a camouflage generation network and a deep model deception module. The camouflage generation network is responsible for generating background textures with a camouflage style, while the deep model deception module targets the attention mechanisms of deep learning models to achieve adversarial effects. During this process, specific constraints are optimized to ensure that the generated textures are distortion-free and effective at deceiving models. This optimization strategy balances texture visual quality with adversarial effectiveness, enabling the generated full-coverage textures to excel in both attack performance and visual naturalness.

The contributions of this work are summarized as follows:

(1) A novel framework, HE-DMDeception, is proposed to jointly optimize human visual deception and deep model deception for generating stealthy and effective adversarial camouflage;
(2) For human visual deception, a CycleGAN-based camouflage network is employed to generate highly camouflaged background textures, which serve as the initial input for subsequent deep model deception.

(3) A SAM-based segmentation pipeline with semi-supervised fine-tuning is introduced to mitigate the scarcity of annotated vehicle masks;
(4) NMS-targeted and attention-dispersion loss terms are designed to explicitly disrupt detection pipelines while preserving camouflage fidelity;

**Table 1.** Overview of advantages and disadvantages of adversarial sample generation methods.

| Category | Methods | Key Idea | Advantages | Disadvantages |
| --- | --- | --- | --- | --- |
| Patch-based methods | Dual Attention Suppression (DAS) [1] | Generate localized adversarial patches by concentrating noise on target objects. | Easy to implement with strong local attacks. | limited to local regions, sensitive to occlusion and patch placement, and performs poorly in multi-view scenarios. |
| Camouflage-based methods with differentiable renderer | Attacks Beyond the Image Space [2], MeshAdv [3] | Optimize texture of 3D object via differentiable renderer. | Alters visual attributes of entire object; texture optimized for 3D surfaces; effective in single-view attacks. | Computationally intensive; may produce low-quality textures; limited robustness under partial occlusion; effectiveness decreases if camouflaged region is obscured. |
| Camouflage-based methods with non-differentiable renderer | CAMOU [4], Universal Physical Camouflage Attack (UPC) [5] | Apply optimized camouflage to 3D object via non-differentiable renderer. | Enables real-world application; allows multiple refinements on surface. | Hard to optimize globally; limited texture quality; less effective in multi-view scenarios; sensitive to surface geometry. |
| Full-Coverage methods | FCA [6] | Combines full-coverage camouflage generation network and deep model deception module. | Generates high-fidelity, visually subtle textures that remain effective across multi-view, partially occluded, and viewpoint-agnostic scenarios, achieving a balance between texture quality and adversarial potency. | Slightly more computationally complex than patch-based methods; requires neural renderer for texture mapping. |

Building upon these improvements, HE-DMDeception integrates high-fidelity texture synthesis with model-specific deception mechanisms to produce stable adversarial textures within a full-coverage framework. Experimental results validate the effectiveness of these enhancements, showing that the generated adversarial samples not only exhibit strong attack capabilities in 3D object detection tasks but also maintain a high degree of visual consistency with the original camouflage patterns.

## 2. Network Architecture

The network architecture, depicted in Figure 1, consists of two main components: human visual deception and deep model deception. The human visual deception module is based on the CycleGAN network, which comprises two generators ($G$,$F$) and two discriminators ($D_X$,$D_Y$). This module operates across two distinct image domains: camouflaged images ($X$) and background images ($Y$). These datasets are used for training, ultimately generating an initial background texture with a camouflage pattern ($\mathbf{T}_0$), which serves as the adversarial texture for the 3D vehicle in the deep model deception module.

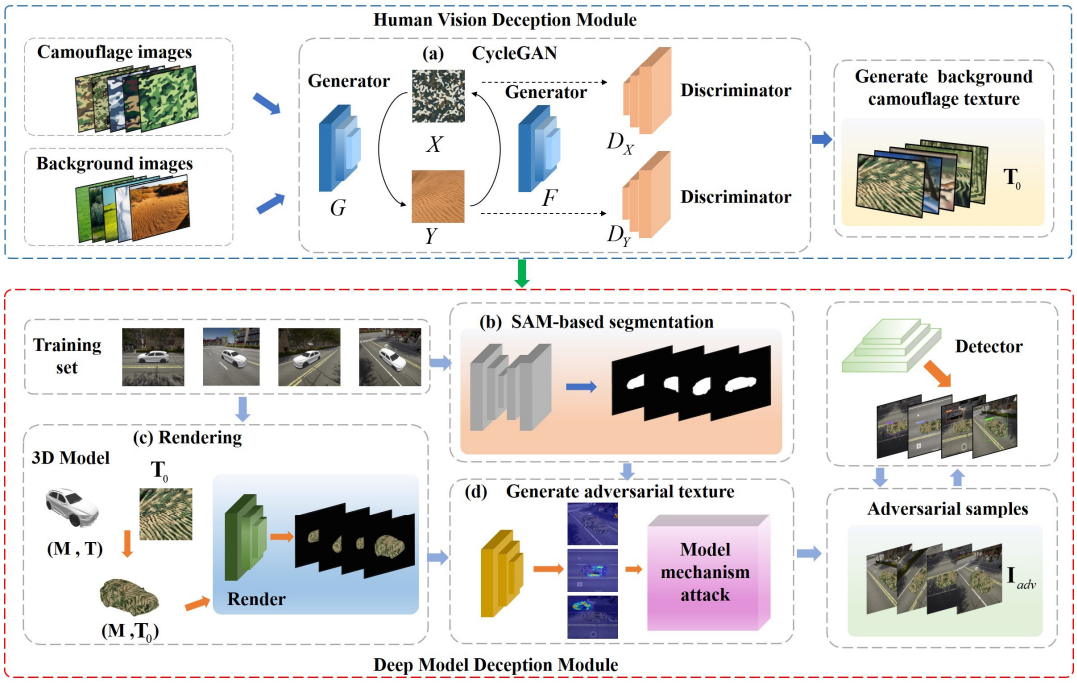



**Figure 1.** Overview of the HE-DMDeception framework. The pipeline includes four modules: (**a**) CycleGAN-based background camouflage texture generation, (**b**) SAM-based segmentation for training data generation, (**c**) Rendering and (**d**) Model mechanism attack based adversarial texture generation Each arrow is annotated with the corresponding data flow and module function.

The deep model deception module employs a neural renderer ($R$) to generate adversarial textures through model-based adversarial training, thereby mitigating significant texture distortion. For a vehicle training set ($\mathbf{B}, \boldsymbol{\omega}$), where $\mathbf{B}$ represents images of the target vehicle with true labels, and $\boldsymbol{\omega}$ represents the corresponding camera parameters, 2D vehicle images are rendered from the 3D vehicle model that includes the mesh $\mathbf{M}$ and texture $\mathbf{T}$, using the camera parameters and the renderer $R$. Finally, the rendered vehicle image $\mathbf{T}_{adv}$ is merged with the background image to produce the final adversarial sample image $\mathbf{I}_{adv}$.

This framework attacks the deep model through non-maximum suppression and attention mechanisms, specifically by dispersing attention weights, which reduces the model's focus on the target.

### *2.1. Human Vision Deception Module*

The camouflage generation network used for human vision deception is built upon a generative adversarial network (GAN). Like a GAN, it requires a large dataset for training the network model, which simultaneously improves the performance of both the generator and discriminator sub-networks. Ultimately, this allows the generator to learn a vast amount of background feature information, enabling it to generate highly realistic patterns.

#### 2.1.1. Camouflage-Style Background Dataset

Background images were collected from both field photography and computer generation, encompassing diverse environments such as snow, forest, desert, sand, and grassland. The real background dataset contains a total of 518 images, including 117 snow, 103 forest, 98 desert, 90 sand, and 110 grassland samples. In addition, 200 camouflage images were gathered from multiple camouflage patterns. To further enrich the dataset, CycleGAN-based style transfer was applied to generate camouflage-style backgrounds from these real and camouflage images, followed by a series of data augmentation techniques. This process produced 1600 synthetic images across the five environments. Such integration of real

and generated data provides a viable solution for implementing high-fusion camouflage disguise. All datasets were split into 80% training, 10% validation, and 10% testing, with proportional representation of each class in every split. Representative examples of the background dataset and camouflage styles are shown in Figure 2.

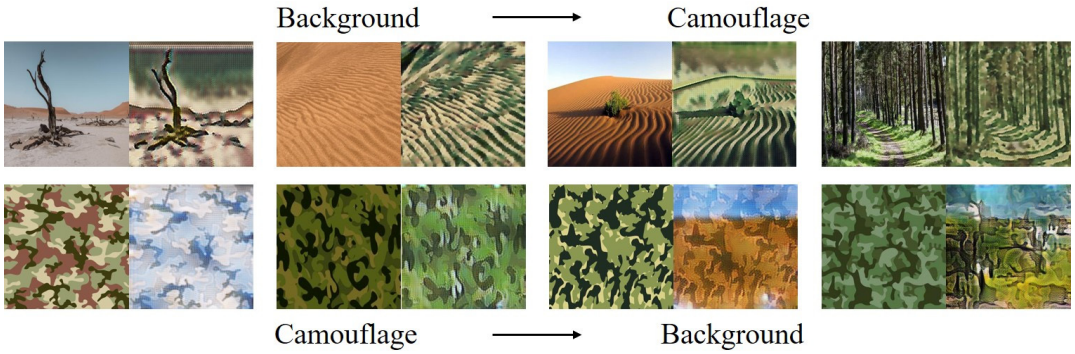



**Figure 2.** Schematic diagram of the background dataset.

#### 2.1.2. Cycle-Consistent Generative Adversarial Network

CycleGAN is an unsupervised image-to-image translation model that generates high-fidelity camouflage patterns without requiring paired training data [7]. It utilizes unpaired background and camouflage samples, simplifying data collection in scenarios where matched pairs are unavailable. The cycle consistency loss ensures faithful color and texture transformation, mitigating common GAN artifacts such as color distortion and structural instability. Consequently, it produces seamless camouflage patterns that blend effectively into background environments, significantly enhancing concealment.

CycleGAN involves two generators and two discriminators for two image domains, $X$ and $Y$. Generator $G(\cdot)$ transforms camouflage images from $X$ to background images in $Y$, aiming to fool discriminator $D_Y(\cdot)$. Generator $F(\cdot)$ does the reverse, converting background images from $Y$ to $X$ to fool discriminator $D_X(\cdot)$.

$$\mathcal{L}_{\text{cyc}}(G, F, D_X, D_Y) = E_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + E_{y \sim p_{\text{data}}(y)}[\|[\|G(F(y)) - y\|_1]\|] \quad (1)$$

The Formula (1) consists of two cycle consistency losses: forward and backward. The forward loss $E_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1]$ ensures that after transforming an image $x$ in domain $X$ to $G(x)$ in domain $Y$ and then back to $F(G(x))$ in domain $X$, the $L_1$ distance between $F(G(x))$ and $x$ is minimized. Similarly, the backward cycle consistency loss $\mathrm{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]$ minimizes the distance between $G(F(y))$ and $y$ in domain $Y$. Together, these losses guarantee that the image can be approximately restored after two-way transformation, preventing unreasonable mappings in the generator.

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \alpha \mathcal{L}_{\text{cyc}}(G, F) \quad (2)$$

$$G^*, F^*, D_X, D_Y = \text{argmin ma}\mathcal{L}(G, F, D_X, D_Y) \quad (3)$$

The objective function Equation (2) combines adversarial loss and cycle consistency loss, with $\alpha$ controlling the importance of the cycle consistency. The goal is to optimize generators $G$ and $F$ such that, when confronted with discriminators $D_X$ and $D_Y$, the generated images are both similar to target domain images (via adversarial loss) and maintain cycle consistency (via cycle consistency loss). The optimization problem shown in Equation (3) seeks to balance adversarial training by minimizing with respect to the generators and maximizing with respect to the discriminators. Note that the adversarial

loss contains two terms: $\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$ and $\mathcal{L}_{\text{GAN}}(F, D_X, Y, X)$, which can be written as follows:

$$\begin{cases} \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = E_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + E_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x)))] \\ \mathcal{L}_{\text{GAN}}(F, D_X, X, Y) = E_{x \sim p_{\text{data}}(x)}[\log D_X(x)] + E_{y \sim p_{\text{data}}(y)}[\log(1 - D_X(F(y)))] \end{cases} \tag{4}$$

#### 2.1.3. Neural Renderer

The Neural 3D Mesh Renderer (NMR) is a deep learning-based method designed to generate high-quality 2D images from 3D mesh data [8]. It emulates the traditional rendering pipeline using a neural network comprising a rendering network and a view transformation module. The rendering network produces images based on factors such as lighting, viewpoint, and material properties, while the view transformation module simulates variations in perspective. During training, NMR fine-tunes the network by comparing the generated images with real ones, thereby enhancing realism.

A key advantage of NMR is its capacity to produce 2D images directly from 3D models, effectively overcoming challenges in lighting, viewpoint, and material properties.

In the context of physical world attacks, neural renderers are used to convert 3D objects into input images required by deep learning systems. A 3D object is represented by a mesh tensor $\mathbf{M}$, a texture tensor $\mathbf{T}$, and a ground truth label $\mathbf{y}$, denoted as $(\mathbf{M}, \mathbf{T})$. Given environmental conditions $\boldsymbol{\omega}$ (such as camera view, object distance, lighting, etc.), the neural renderer $R$ can generate the input image $\mathbf{I} \in R^{H \times W \times 3}$, i.e., $\mathbf{I} = R((\mathbf{M}, \mathbf{T}), \boldsymbol{\omega})$. This process indicates that the neural renderer plays an important role in physical world attacks, connecting the real object with the input image of the deep learning system, and providing the necessary image data for subsequent generation of adversarial camouflage.

In generating adversarial camouflage in the physical world, the original texture tensor $\mathbf{T}$ is replaced with the adversarial texture tensor $\mathbf{T}_{adv}$, which is then processed by the neural renderer to produce the adversarial image $\mathbf{I}_{adv}$, i.e., $\mathbf{I}_{adv} = R((\mathbf{M}, \mathbf{T}_{adv}), \boldsymbol{\omega})$. This adversarial image is then used to attack the deep learning model.

### *2.2. Deep Model Deception Module*

Adversarial attacks can be categorized into black-box and white-box attacks. Black-box attacks exhibit better transferability but weaker performance, while white-box attacks demonstrate stronger performance but may suffer from overfitting. To balance adversarial effectiveness and transferability, white-box attacks are designed to target the non-maximum suppression (NMS) and attention mechanisms of deep learning models during training, thereby generating adversarial samples with strong transferability. NMS is a critical step in object detection, used to eliminate redundant candidate boxes, whereas attention mechanisms are common feature extraction methods shared across object detectors. By attacking these shared features, the effectiveness and transferability of adversarial samples can be significantly enhanced.

#### 2.2.1. NMS Mechanism Attack

The NMS mechanism attack disrupts the process by increasing the confidence scores of candidate boxes and reducing the intersection-over-union (IoU) values between boxes, which results in retaining redundant boxes and causing the mechanism to fail, leading to model detection errors. The attack strategy involves raising confidence scores, decreasing IoU values, and reducing the size of candidate boxes to lower computational resource demands, thereby achieving an effective attack on the NMS mechanism. The loss function for this attack is as follows:

$$\mathcal{L}_{adv}^{\text{IoU}} = \sum_{i}^{N} \text{IoU}\left(b^i, b_{gt}^i\right) \tag{5}$$

where $N$ represents the multi-scale outputs of the object detection model, while $b^i$ and $b^i_{gt}$ denote the predicted result at the $i$-th scale and the corresponding ground truth box, respectively.

#### 2.2.2. Attention Mechanism Attack

The attention mechanism attack disperses the model's focus on target regions, weakening its feature extraction capability and resulting in erroneous detection and classification. To achieve this, we reduce pixel values in the attention map and expand the edges of zero-pixel regions to shift the hotspot areas. An attention dispersion loss function is designed to target the model's feature extraction capability. The loss function is as follows:

$$\mathcal{L}_a = \|(\gamma \cdot \mathbf{E} + \mathbf{1}) \odot (\mathbf{T}_{adv} - \mathbf{T}_0)\|_2^2 \tag{6}$$

where $\gamma \cdot \mathbf{E} + \mathbf{1}$ represents the attention tensor; $\mathbf{1}$ is a tensor where all elements are 1, with the same dimensions as $\mathbf{E}$; $\odot$ denotes element-wise multiplication; $\mathbf{T}_0$ and $\mathbf{T}_{adv}$ represent the initial and generated adversarial texture tensors, respectively.

#### 2.2.3. Adversarial Camouflage Texture Constraints

The adversarial camouflage texture constraints aim to avoid conspicuous visual features, ensuring that the generated textures remain difficult to detect in real-world scenarios. By constraining the Euclidean distance between the generated texture and the original texture, the adversarial patterns are prevented from excessive distortion during training. A similarity loss function is designed to minimize image differences, preserving the original camouflage characteristics. Additionally, smoothing techniques are introduced to reduce pixel-to-pixel gaps, enhancing the stealthiness of the camouflage. The loss function is as follows:

$$\mathcal{L}_s = \sum_{i,j} \left(x_{i,j} - x_{i+1,j}\right)^2 + \left(x_{i,j} - x_{i,j+1}\right)^2 \tag{7}$$

where $x_{i,j}$ represents the pixel value at coordinate $(i, j)$ of the rendered image covered with the adversarial texture.

#### 2.2.4. The Overall Loss Function

In addition to the three loss functions mentioned above, the classification loss in object detection (representing the classification probability of the target class) also needs to be considered. Specifically, for the detection results, the probability of the target class $t$ at the $i$-th scale is denoted as $b^i_{cls^t}$. The classification loss is given by the following formula:

$$\mathcal{L}^{cls}_{adv} = \sum_i^N b^i_{cls^t} \tag{8}$$

The overall loss function of the deep model deception module can be expressed as follow:

$$\mathcal{L}_{adv} = \lambda_1 \mathcal{L}^{iou}_{adv} + \lambda_2 \mathcal{L}_a + \lambda_3 \mathcal{L}^{cls}_{adv} + \lambda_4 \mathcal{L}_s \tag{9}$$

where the fourth term denotes the traditional classification loss. Note that the first three loss terms $\mathcal{L}^{\text{IoU}}_{adv}$, $\mathcal{L}_a$ and $\mathcal{L}^{cls}_{adv}$ are given equal weights. Table 2 summarizes each loss term, including its purpose and relative weight.

**Table 2.** Summary of each of the four individual loss terms in the overall loss function.

| Loss Term | Purpose/Mechanism | Weight |
| --- | --- | --- |
| NMS Mechanism Attack Loss $\mathcal{L}_{adv}^{iou}$ | This attack disrupts NMS by inflating confidence scores and lowering IoU to prevent proper duplicate filtering, resulting in cluttered false positives. | $\lambda_1 = 1$ |
| Attention Mechanism Attack Loss $\mathcal{L}_a$ | This attack weakens the model's feature extraction by dispersing its attention, leading to misclassification and missed detections. | $\lambda_2 = 1$ |
| Camouflage Similarity Loss $\mathcal{L}_{adv}^{cls}$ | This constraint ensures the adversarial texture remains stealthy and visually plausible in real-world scenarios by minimizing its distortion from the original pattern. | $\lambda_3 = 1$ |
| Classification Loss $\mathcal{L}_s$ | This core loss function minimizes the model's probability score for the correct class, driving the adversarial misclassification. | $\lambda_4 = 0.5$ |

### *2.3. Image Segmentation Module*

Acquiring large-scale annotated data for automotive image segmentation is challenging, which limits model performance. To address this, we propose a two-stage training strategy utilizing the pre-trained Segment Anything (SAM) model, which leverages unlabeled data to improve performance [9]. The process consists of semi-supervised self-training followed by supervised fine-tuning.

In the data processing stage, the automotive image data $\mathbf{X} \in R^{l \times w \times h}$ (where $l$, $w$, and $h$ represent the length, width, and number of channels of the image, respectively) and its corresponding segmentation labels $\mathbf{Y}_{seg} \in R^{l \times w \times h}$ are divided into a series of sub-images $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in R^{c \times w \times h}$ and $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\} \in R^{c \times w \times h}$ according to specific rules (where $c$ is a relevant parameter such as the number of channels set according to the image division requirements). This division process can be represented by the following mathematical formula:

$$\mathbf{x}_i = \mathbf{X}[i, :, :] \text{ and } \mathbf{y}_i = \mathbf{Y}_{\text{seg}}[i, :, :] \text{ for } i = 1, 2, \cdots, n \tag{10}$$

Entering the semi-supervised self-training stage, the unlabeled automotive image dataset $\mathbf{X}_u = \{\mathbf{x}_{n+1}, \mathbf{x}_{n+2}, \ldots, \mathbf{x}_{n+m}\} \in R^{c \times w \times h}$ ($m$ represents the number of unlabeled images) is input into the SAM model $f$. The SAM model will extract features from these unlabeled data, learn their feature representations, and then predict pseudo-labels $\hat{\mathbf{Y}}_u = \{\hat{\mathbf{y}}_{n+1}, \hat{\mathbf{y}}_{n+2}, \ldots, \hat{\mathbf{y}}_{n+m}\}$. The specific processes of feature extraction and pseudo-label generation can be described by the following formulas:

$$\mathbf{z}_j = f_{\text{encoder}}(\mathbf{x}_j; \boldsymbol{\theta}_{\text{encoder}}) \tag{11}$$

$$\hat{\mathbf{y}}_j = f_{\text{decoder}}(\mathbf{z}_j; \boldsymbol{\theta}_{\text{decoder}}) \tag{12}$$

In the formulas, $\mathbf{z}_j$ represents the feature representation of the sub-image, $\hat{\mathbf{y}}_j$ is the pseudo-label generated by the SAM model, and $\boldsymbol{\theta}_{\text{encoder}}$ and $\boldsymbol{\theta}_{\text{decoder}}$ are the model parameters of $f_{\text{encoder}}$ and $f_{\text{encoder}}$, respectively.

By introducing a large number of unlabeled data, the generalization ability of the model is significantly enhanced, and the dependence on labeled data is reduced accordingly. Next, the pseudo-labeled dataset $(\mathbf{X}_u, \hat{\mathbf{Y}}_u) = \left\{(\mathbf{x}_j, \hat{\mathbf{y}}_j)\right\} (j = n+1, \ldots, n+m)$ is combined with the existing actual labeled dataset $(\mathbf{X}_l, \mathbf{Y}_l) = \left\{(\mathbf{x}_i, \mathbf{y}_{i)}\right\} (i = 1, \ldots, n)$ to construct an extended training set. The Unet model is trained using this extended training set. During the training process, the Binary Cross Entropy with Logits Loss is used as the loss function, and the AMAD optimizer is used to adjust the model parameters.

Following the semi-supervised self-training phase, the Unet model is fine-tuned with labeled data to correct deviations from pseudo-labels and further enhance model performance. This two-stage approach effectively exploits the feature extraction capabilities of the SAM model, leading to improved segmentation performance in automotive image tasks. It offers a viable solution to the challenge of limited labeled data. The detailed process is illustrated in Figure 3.

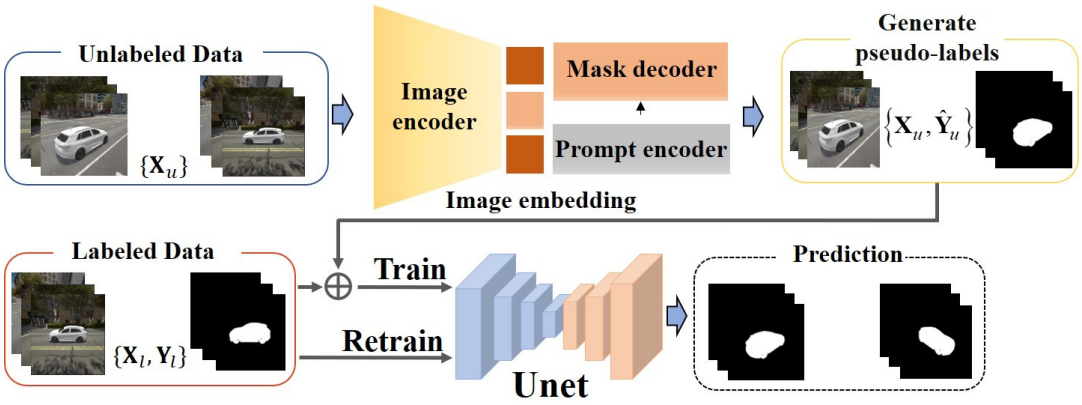



**Figure 3.** Two-stage training method.

## 3. Experimental Setup

### *3.1. Datasets*

This study employs the CARLA platform, an established open-source simulator grounded in Unreal Engine 4, to replicate physical world attacks within a 3D virtual environment [10,11]. CARLA is equipped with a diverse set of high-resolution digital assets, enabling the simulation of realistic urban settings for autonomous driving research. A range of high-fidelity digital environments, such as modern urban layouts, are provided by the CARLA (v0.9.14) simulator, which is developed using Unreal Engine 4. For fair comparison with previous work, the datasets from DAS [1] and FCA [6] are directly employed. The training set consists of 12,500 images at 1920 $\times$ 1080 resolution, with an additional 3000 images retained for testing. Within the simulation environment, 155 locations were randomly chosen for vehicle positioning. At each location, 100 images were acquired using a virtual camera configured with varying distances (5, 10, 15, 20 m), pitch angles (22.5°, 45°, 67.5°, 90°), and yaw angle (south, north, east, west, southeast, southwest, northeast, and northwest).

### *3.2. Evaluation Metrics*

To evaluate adversarial attack performance, four image classification models (ResNet-152, DenseNet-201, Inception-V3, VGG-19) and five object detection models (YOLOv8, RT-DETR, SSD, Faster R-CNN, Mask R-CNN) were tested [12–20]. Accuracy was used for image classification, and P@0.5 (percentage of correct detections with an IOU threshold of 0.5) for object detection [21,22].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \tag{13}$$

where TP (True Positive) denotes the number of positive samples correctly identified; TN (True Negative) refers to the number of negative samples correctly identified; FN (False Negative) indicates the number of positive samples incorrectly classified as negative; and FP (False Positive) represents the number of negative samples incorrectly classified as positive.

This study evaluates the impact of various adversarial camouflage methods on the accuracy of four classification models: Inception-V3, VGG-19, ResNet-152, and DenseNet.

Experimental results indicate that the proposed method significantly outperforms existing approaches.

## 4. Analysis and Discussion

Without adversarial camouflage, as shown in Table 3, Inception-V3 achieved the highest accuracy (58.33%), while VGG-19 performed the worst (40.28%). ResNet-152 and DenseNet yielded accuracies of 41.67% and 46.53%, respectively, suggesting that Inception-V3 possesses the strongest classification capability, whereas VGG-19 is the most susceptible to errors. Upon applying adversarial camouflage, all evaluated methods—MeshAdv, CAMOU, UPC, and Dual Attention Suppression (DAS)—decreased model accuracy, with particularly pronounced effects on VGG-19 and ResNet-152. MeshAdv reduced accuracy by 18.05% for Inception-V3, 6.25% for VGG-19, 2.78% for ResNet-152, and 10.42% for DenseNet. CAMOU exhibited the strongest attack, especially on VGG-19, where accuracy declined by 11.11%. In contrast, UPC and DAS demonstrated weaker effects, with DAS affecting DenseNet the least (a reduction of only 4.86%). The proposed method had the most substantial impact, reducing accuracy by 18.67% for ResNet-152 and 13.86% for DenseNet. While Inception-V3 and VGG-19 retained higher accuracy (53.73% and 24.4%, respectively), the proposed approach still outperformed all other methods. Notably, DenseNet's accuracy dropped to 32%, underscoring the effectiveness of the attack. Overall, despite variations among models, the proposed method consistently reduces classification accuracy, particularly for ResNet-152 and DenseNet, demonstrating its strong potential to impair the recognition capabilities of state-of-the-art classification models.

**Table 3.** The comparison results of adversarial attacks on the classification task.

| Method | Accuracy (%) [1] | | | |
|---|---|---|---|---|
| | Inception-V3 | VGG-19 | ResNet-152 | DenseNet |
| Raw | 58.33 | 40.28 | 41.67 | 46.53 |
| MeshAdv | 40.28 | 34.03 | 38.89 | 36.11 |
| CAMOU | 40.28 | 29.17 | 31.25 | 45.14 |
| UPC | 35.41 | 33.33 | 33.33 | 41.67 |
| DAS | 31.94 | 27.78 | 29.86 | 41.67 |
| HE-DMDeception | 30 | 24.4 | 23 | 32 |

[1] The percentage of correctly classified predictions out of all predictions made.

In this experiment, six adversarial camouflage methods (MeshAdv, CAMOU, UPC, DAS, FCA and HE-DMDeception) were evaluated on five object detection models—YOLOv8, RT-DETR, Faster R-CNN, SSD, and Mask R-CNN—using the P@0.5 metric, as summarized in Table 4. MeshAdv had minimal effects on YOLOv8 and RT-DETR, maintaining P@0.5 at 100%, while causing more notable degradation on Faster R-CNN (71.84%), SSD (66.44%), and Mask R-CNN (80.84%). CAMOU reduced P@0.5 to 69.64% for Faster R-CNN and 76.44% for Mask R-CNN, showing stronger effects on these models. UPC maintained P@0.5 at 100% for YOLOv8 and RT-DETR but caused declines in Faster R-CNN (76.94%), SSD (74.58%), and Mask R-CNN (81.97%). DAS produced the weakest attack, with small decreases for YOLOv8 (92.36%) and RT-DETR (91%) and a larger drop for Faster R-CNN (62.11%). FCA achieved more substantial attacks, reducing P@0.5 to 65.28% for YOLOv8 and 66% for RT-DETR, and further decreasing accuracy for Faster R-CNN (24.31%), SSD (29.17%), and Mask R-CNN (29.17%).

**Table 4.** The comparison results of adversarial attacks on the detection task.

| Method | P@0.5 (%) [1] | | | | |
|---|---|---|---|---|---|
| | **YOLOv8** | **RT-DETR** | **Faster-RCNN** | **SSD** | **MaskRCNN** |
| Raw | 100 | 100 | 86.04 | 81.54 | 89.24 |
| MeshAdv | 100 | 100 | 71.84 | 66.44 | 80.84 |
| CAMOU | 99.31 | 98 | 69.64 | 73.81 | 76.44 |
| UPC | 100 | 100 | 76.94 | 74.58 | 81.97 |
| DAS | 92.36 | 91 | 62.11 | 68.81 | 70.21 |
| FCA | 65.28 | 66 | 24.31 | 29.17 | 29.17 |
| HE-DMDeception | 50 | 55 | 20 | 25 | 25 |

[1] The percentage of detections with an IoU $\geq$ 0.5 relative to ground-truth boxes.

The proposed HE-DMDeception method achieved the most significant attacks overall, particularly on Faster R-CNN, SSD, and Mask R-CNN, where P@0.5 dropped to 20%, 25%, and 25%, respectively. While the attack effects on YOLOv8 and RT-DETR were milder (50% and 55%), detection accuracy was still reduced. This milder effect can be explained by the architectural characteristics of these models. YOLOv8 employs a one-stage detection architecture with strong feature pyramids, extensive multi-scale feature aggregation, and a robust spatial attention mechanism, which collectively enhance its resilience to localized and global texture perturbations. Similarly, RT-DETR integrates transformer-based attention with NMS adaptations, enabling the model to effectively focus on salient object features while mitigating the influence of adversarial textures. These mechanisms reduce the impact of subtle adversarial camouflage, making YOLOv8 and RT-DETR less sensitive to the attacks compared to two-stage detectors like Faster R-CNN or region-based methods in SSD and Mask R-CNN.

Table 4 shows that HE-DMDeception reduces P@0.5 to 20% for Faster R-CNN and to 50% for YOLOv8, indicating variable transferability across architectures. This suggests our method more strongly disrupts region-proposal and multi-stage pipelines; transfer to other unseen detectors is therefore partially effective but not uniform. Future work will explore ensemble-based optimization and feature-space regularization to improve cross-model transferability.

Overall, HE-DMDeception outperformed all other camouflage-based techniques in reducing P@0.5, demonstrating strong adversarial capability across diverse detection architectures and confirming the robustness and generality of the proposed method.

To examine the impact of our generated adversarial samples on the model's feature extraction, we used Grad-CAM to generate attention maps during vehicle detection. Grad-CAM visualizes the model's focus by calculating the gradient information at each network layer, highlighting the regions on which the model relies for decision-making [23,24]. By comparing the attention maps of initial camouflage and adversarial samples, we gain insights into how adversarial camouflage alters the model's attention distribution and affects feature extraction.

In the experiment, we first generated camouflage images of vehicles and obtained the corresponding Grad-CAM attention maps, shown in Figure 4, the model concentrated on key vehicle features, indicating a strong focus on these areas. However, after applying adversarial camouflage, the attention maps revealed a significant shift in attention, with the model focusing on irrelevant background or camouflage textures instead of the vehicle regions. This suggests that adversarial samples successfully interfered with the feature ex-traction process, preventing the model from extracting useful information from the intended regions. Further analysis revealed that, under HE-DMDeception attacks, detectors

such as YOLOv8 and Faster R-CNN exhibited significant suppression of key vehicle features in the attention maps, and in certain test cases failed to localize the vehicle correctly. Compared to traditional features, adversarial samples significantly degraded the model's performance, affecting both accuracy and feature extraction. The Grad-CAM results clearly demonstrate how adversarial camouflage disrupts attention allocation, leading to a reduction in performance for object detection tasks. These findings suggest that generating adversarial samples can effectively weaken the feature extraction ability of the target mod-el, thus achieving the objective of counteracting intelligent reconnaissance.

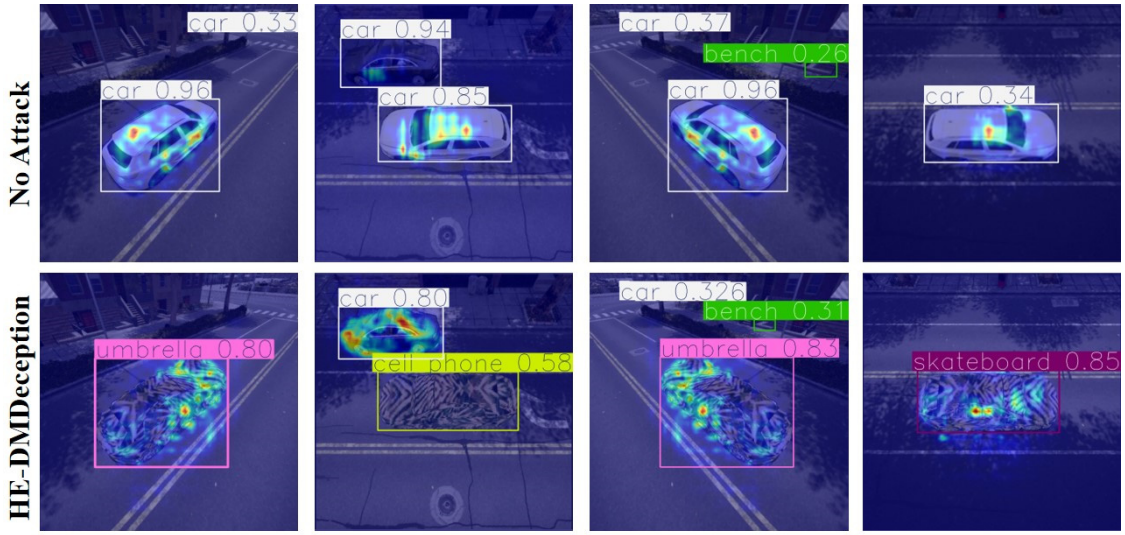



**Figure 4.** Visualization of the attention maps with no attack and the HE-DMDeception attack.

## 5. Conclusions

Despite extensive studies on adversarial samples for 3D object detection, patch-based and camouflage methods still face methodological challenges, often producing visually unnatural textures with limited generalizability across detectors. To address this, we propose HE-DMDeception—an integrated framework combining human-visual and deep model deception via a CycleGAN network and a module that perturbs non-maximum suppression and attention mechanisms under perceptual constraints. A two-stage training strategy using semi-supervised learning and fine-tuning with the SAM model reduces reliance on annotated data. Experiments show the method achieves significantly reduced precision across YOLOv8, RT-DETR, Faster-RCNN, SSD, and Mask R-CNN, while maintaining high visual fidelity. This work bridges perceptual camouflage and adversarial machine learning, offering a dual capability to mislead models and evade human detection, thus enhancing real-world applicability. It represents an advance in generating stealthy adversarial examples, with implications for model reliability evaluation. The core novelty is the unified integration of visual realism and algorithmic deception.

## 6. Limitations and Future Work

Despite its strong performance, HE-DMDeception is limited by poor transferability to unseen architectures, high computational demand—requiring approximately 12 h per vehicle texture on an RTX 3090 GPU—and a reality gap caused by reliance on simulated data. Additionally, segmentation errors can be propagated to the camouflage module. Although advances have been made in texture generation, further improvements in complexity, diversity, and efficiency are still required. Future work will be focused on refining adversarial examples through more efficient optimization, cross-simulator validation, and enhanced texture generation. To bridge the simulation-to-real gap, small-scale physical tests will be

conducted, in which printed camouflage patterns are applied to vehicle prototypes and evaluated under real-world conditions.

**Author Contributions:** Conceptualization, P.Z. and Y.L.; methodology, P.Z. and Y.L.; software, P.Z. and H.L.; validation, Y.T., J.N., Z.X. and H.L.; formal analysis, Y.T.; investigation, J.N.; resources, J.N.; data curation, J.W. and Z.X.; writing—original draft preparation, P.Z., J.W. and Y.L.; writing—review and editing, P.Z. and Y.L.; visualization, Y.T.; supervision, P.Z.; project administration, P.Z.; funding acquisition, P.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Science and Technology Foundation of State Key Laboratory (grant number JCKYS2023LD3).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The dataset can be accessed publicly at https://github.com/carla-simulator/carla (accessed on 15 October 2024).

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

## References

1. Wang, J.; Liu, A.; Yin, Z.; Liu, S.; Tang, S.; Liu, X. Dual Attention Suppression Attack: Generate Adversarial Camouflage in Physical World. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 8561–8570.
2. Zeng, X.; Liu, C.; Wang, Y.S.; Qiu, W.; Xie, L.; Tai, Y.W.; Tang, C.K.; Yuille, A.L. Adversarial Attacks Beyond the Image Space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4302–4311.
3. Xiao, C.; Yang, D.; Li, B.; Deng, J.; Liu, M. MeshAdv: Adversarial Meshes for Visual Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 6898–6907.
4. Zhang, Y.; Foroosh, H.; David, P.; Gong, B. CAMOU: Learning Physical Vehicle Camouflages to Adversarially Attack Detectors in the Wild. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018; pp. 1–20.
5. Huang, L.; Gao, C.; Zhou, Y.; Xie, C.; Yuille, A.L.; Zou, C.; Liu, N. Universal physical camouflage attacks on object detectors. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 717–726.
6. Wang, D.; Jiang, T.; Sun, J.; Zhou, W.; Gong, Z.; Zhang, X.; Yao, W.; Chen, X. FCA: Learning a 3D Full-Coverage Vehicle Camouflage for Multi-View Physical Adversarial Attack. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22–29 February 2022; AAAI Press: Palo Alto, CA, USA, 2022; Volume 36, pp. 2414–2422.
7. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2223–2232.
8. Kato, H.; Ushiku, Y.; Harada, T. Neural 3D Mesh Renderer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 3907–3916.
9. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment Anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 4–11 October 2023; pp. 4015–4026.
10. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. In Proceedings of the 1st Annual Conference on Robot Learning (CoRL), Mountain View, CA, USA, 13–15 November 2017; PMLR: Palo Alto, CA, USA, 2017; pp. 1–16.
11. Wang, Y.; Lv, H.; Kuang, X.; Zhao, G.; Tan, Y.-a.; Zhang, Q.; Hu, J. Towards a Physical-World Adversarial Patch for Blinding Object Detection Models. *Inf. Sci.* **2021**, *556*, 459–471. [CrossRef]
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
13. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269.

14. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
15. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556. [CrossRef]
16. Jocher, G.; Qiu, J.; Chaurasia, A. Ultralytics YOLO (Version 8.0.0) [Computer Software]. 2023. Available online: https://github.com/ultralytics/ultralytics (accessed on 27 August 2025).
17. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs Beat YOLOs on Real-Time Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–23 June 2024; pp. 16965–16974.
18. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Computer Vision—ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer: Cham, Switzerland, 2016; Volume 9905, pp. 21–37.
19. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
20. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
21. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
22. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 586–595.
23. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 618–626.
24. Chattopadhay, A.; Sarkar, A.; Howlader, P.; Balasubramanian, V.N. Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 839–847.


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.