electronics

MDPI

Article

Denoising Diffusion Implicit Model for Camouflaged Object Detection

Wei Cai [1], Weijie Gao [1,*], Xinhao Jiang [2], Xin Wang [1] and Xingyu Di [1]

[1] Xi'an Research Institute of High Technology, Xi'an 710064, China; xhtu807@outlook.com (W.C.); wangxin9550@outlook.com (X.W.); dixy1998@outlook.com (X.D.)
[2] High-Tech Institute, Fan Gong-ting South Street on the 12th, Qingzhou 262500, China; jiangxinhao2020@outlook.com
* Correspondence: gaoweijie331@outlook.com

Abstract: Camouflaged object detection (COD) is a challenging task that involves identifying objects that closely resemble their background. In order to detect camouflaged objects more accurately, we propose a diffusion model for the COD network called DMNet. DMNet formulates COD as a denoising diffusion process from noisy boxes to prediction boxes. During the training stage, random boxes diffuse from ground-truth boxes, and DMNet learns to reverse this process. In the sampling stage, DMNet progressively refines random boxes to prediction boxes. In addition, due to the camouflaged object's blurred appearance and the low contrast between it and the background, the feature extraction stage of the network is challenging. Firstly, we proposed a parallel fusion module (PFM) to enhance the information extracted from the backbone. Then, we designed a progressive feature pyramid network (PFPN) for feature fusion, in which the upsample adaptive spatial fusion module (UAF) balances the different feature information by assigning weights to different layers. Finally, a location refinement module (LRM) is constructed to make DMNet pay attention to the boundary details. We compared DMNet with other classical object-detection models on the COD10K dataset. Experimental results indicated that DMNet outperformed others, achieving optimal effects across six evaluation metrics and significantly enhancing detection accuracy.

Keywords: camouflaged object detection; diffusion model; computer vision; feature fusion; location refinement

Citation: Cai, W.; Gao, W.; Jiang, X.; Wang, X.; Di, X. Denoising Diffusion Implicit Model for Camouflaged Object Detection. *Electronics* **2024**, *13*, 3690. https://doi.org/10.3390/electronics13183690

Academic Editors: Silvia Liberata Ullo and Li Zhang

Received: 1 August 2024
Revised: 12 September 2024
Accepted: 15 September 2024
Published: 17 September 2024

1. Introduction

In the military field, camouflage technology ensures the survivability of weapons, equipment, and personnel on the battlefield by reducing the differences between objects and their surrounding environment. With the development of camouflage technology, the environments in which military camouflaged objects are located have become intricate and complex, with widespread overlapping and obstruction. These objects exhibit high concealment and low recognizability [1]. The task of camouflaged object detection (COD) faces significant challenges.

Camouflaged object detection can be classified into two categories: visible light and non-visible light. In military applications, non-visible-light technologies such as hyperspectral imaging, laser, radar, and infrared are often used to detect camouflaged objects [2–4]. Visible-light detection employs traditional camouflaged object-detection algorithms such as the Deformable Part Model (DPM), the Histogram of Oriented Gradients method, and Support Vector Machines [5]. However, due to the weak feature expression capabilities of these traditional methods, it is difficult to detect camouflaged objects in complex backgrounds.

With the rapid development of deep learning technology, object-detection algorithms under visible-light conditions [6–11] have seen significant improvements both in accuracy and speed. However, when detecting camouflaged objects, these deep learning-based algorithms still suffer from missed detections, false detections, and poor accuracy. Therefore, researching

how to enhance the detection accuracy of camouflaged objects based on deep learning in the visible-light spectrum is particularly crucial.

Addressing the practical needs of military applications, we propose a Diffusion Model for Camouflaged Object-Detection Network (DMNet) under visible-light conditions. DMNet treats the object-prediction bounding box as a generation task based on a diffusion model. It adds Gaussian noise to the ground-truth bounding box in order to obtain noisy random bounding boxes, and the model iteratively refines these noisy bounding boxes to obtain the final prediction results. This is shown in Figure 1, which illustrates the difference between the proposal bounding box-generation approach guided by the diffusion model and that of common models. To enhance the features extracted by the backbone network, we design a parallel fusion module (PFM) and a progressive feature pyramid network (PFPN). These modules enhance the representation of object features by parallelly integrating asymmetric convolutions and gradually fusing feature layer information. The PFPN includes an upsample adaptive spatial feature fusion (UAF) module that balances information from different feature layers. Furthermore, a location refine module (LRM) is constructed to enable the network to effectively focus on local detail features and produce clear localization information. Finally, by improving the diffusion model head (DiffHead) that is more suitable for the COD task, it can fully extract and analyze the features of camouflaged objects, thereby enhancing the detection accuracy. To summarize, the main contributions of this paper are as follows:

1. We construct a camouflaged object-detection network, DMNet, based on the diffusion model to improve the accuracy of COD detection;
2. We propose a progressive feature pyramid network-connection method and design an upsampling adaptive spatial feature-fusion module to achieve the gradual fusion of adjacent feature layers, thereby enhancing the expressive power of the features;
3. We propose that PFM, LRM, and Diffhead, respectively, increase the receptive field, enhance the representation learning of low-level boundary detail information in camouflaged objects, and become more sensitive to spatial information, thereby enhancing detection accuracy.
4. The constructed DMNet exhibits exceptional performance on the COD10K dataset, surpassing the classic algorithm in six evaluation metrics, thereby demonstrating its effectiveness in enhancing COD accuracy.

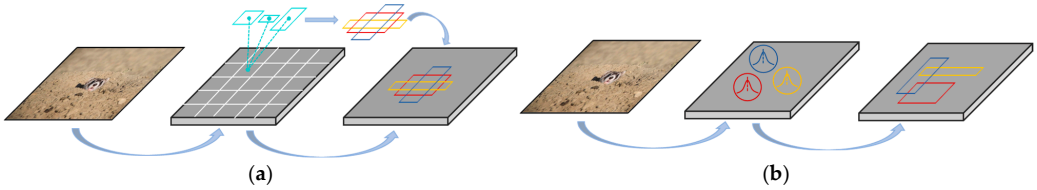


Figure 1. Different ways of generating suggestion boxes. (**a**) Normal generation method of proposal boxes; (**b**) The diffusion model leads to produce proposal boxes. (The different colors in the figure represent different proposal boxes).

The rest of this paper is structured as follows: In Section 2, we introduce the related work. Section 3 describes the network structure of DMNet in detail, as well as its mathematical derivation process, training and inference strategies. Section 4 provides a discussion of the experimental results. Finally, we summarize the research content in Section 5.

2. Related Work

2.1. Deep Learning-Based Object Detection

Object detection is one of the most important tasks in the field of computer vision, which aims to identify objects of interest in images and locate their positions in the images through bounding boxes. Deep learning-based object-detection algorithms can mainly be classified into two categories: two-stage object detection and one-stage object detection. One-stage object-detection algorithms directly predict the target positions and categories in

images through neural networks in an end-to-end manner, without the need to generate candidate regions for further classification and localization. These one-stage object-detection algorithms are simpler and more efficient in design, usually with faster inference speeds. Common one-stage object-detection algorithms include YOLO [6], SSD [7], EfficientDet [8], etc. Two-stage object-detection algorithms divide the object-detection task into two stages: firstly, generating candidate regions, and secondly, classifying and refining the localization of these candidate regions. Common two-stage object-detection algorithms include Sparse RCNN [9], Swin-RCNN [10], SPP-Net [11], etc. Due to the high similarity between camouflaged objects and their surrounding environments, it is difficult to accurately identify camouflaged objects. Therefore, we adopt the idea of two-stage object-detection algorithms. In recent years, researchers have proposed many methods to improve the accuracy and speed of object-detection algorithms, which has promoted the wide application of deep learning-based object-detection technology in fields such as autonomous driving [12], industrial quality inspection [13], medical image analysis [14], etc.

2.2. Deep Learning-Based Camouflaged Object Recognition

Camouflaged object recognition (COR) aims to accurately identify targets hidden in complex backgrounds within images. The difficulty of COR lies mainly in two aspects: firstly, the contrast between the foreground and background is not distinct, making it challenging to capture the edges of camouflaged objects; secondly, there are interfering factors such as obstacles and shadows in the environment that hinder the recognition of camouflaged objects. Figure 2 introduces the tasks related to camouflaged object recognition, including camouflaged object detection, camouflaged object segmentation (COS), and camouflaged instance segmentation (CIS).

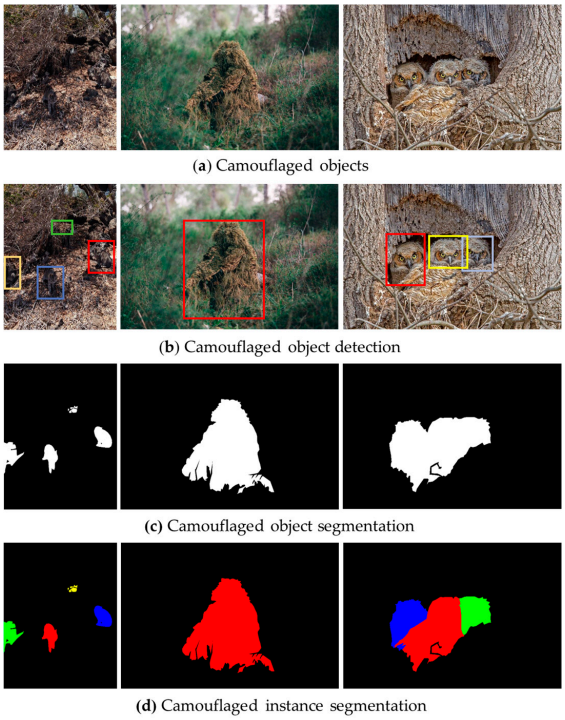


Figure 2. The illustration of camouflaged objects and COR. (The different colors in the figure represent different objects that have been identified).

2.3. Diffusion Model

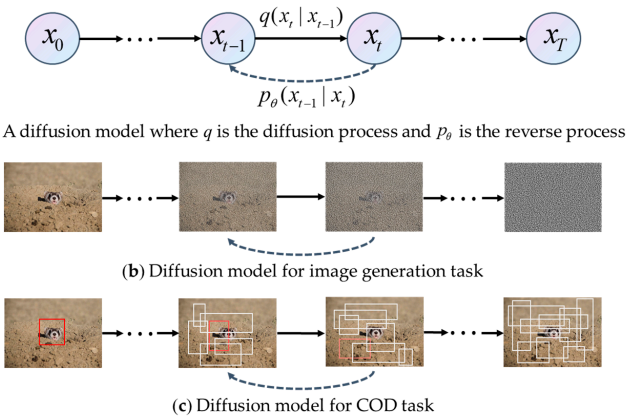
A diffusion model is a type of probabilistic generative model aimed at learning the sampling distribution over the sample space and gradually generating samples from the distribution by removing noise. This model does not require adversarial training in content generation and is trained using the two-step process. First, during the training

process, the forward diffusion process gradually adds noise to a reference (usually an image) according to a predefined schedule. In inference process, the reverse diffusion process generates random initial noise, and the model gradually eliminates the noise until it forms data points of the training distribution, converting them into samples of the object distribution being modeled. Currently, research on diffusion models is primarily based on three types: Denoising Diffusion Probabilistic Models (DDPMs), Score-based Generative Models (SGMs), and Stochastic Differential Equations (SDEs).

Diffusion models have been applied to many fields, such as text-to-image guided synthesis [15], 3D-shape generation [16], molecular prediction [17], video generation [18] and image restoration [19]. In 2022, Wu et al. proposed MedSegDiff, the first medical image segmentation framework based on DDPM, which aims to achieve precise results during the adaptive calibration process [20]. In 2023, Zhao Peiang et al. from the University of Science and Technology of China [21] proposed a general lesion-detection algorithm based on diffusion model, named Diffusive Universal Lesion Detection (DiffULD), to address the issue of insufficient targets. This strategy provides additional high-quality training objects while avoiding significant performance degradation. In 2024, Lv et al. [22] introduced the first model that incorporates a diffusion model into the multi-object tracking task to address nonlinear motion prediction (Diffusion-based Multiple Object Tracker (DiffMOT)), optimizing the diffusion process with fewer sampling steps. This paper aims to construct a diffusion model for the camouflage object-detection algorithm.

3. Methods

As illustrated in Figure 3, we undertake the network architecture design for DMNet, which is predicated on denoising diffusion implicit models (DDIMs). This encompasses both forward and backward processes.



(**a**) A diffusion model where q is the diffusion process and p_θ is the reverse process

(**b**) Diffusion model for image generation task

(**c**) Diffusion model for COD task

Figure 3. The diffusion process of COD. From left to right is the forward process of adding noise, and from right to left is the backward denoising process.

3.1. Mathematical Derivation

We propose DMNet in this paper, which is designed based on DDIMs. A DDIM can be seen as a model that mimics the diffusion process in nature, with its core idea being to gradually remove noise from images and ultimately generate clear and realistic images. This method can be applied to generating high-quality images, text, or other types of data.

Set the input for object detection as (x, b, c), where x is the input image, and b and c are, respectively, a set of bounding boxes and category labels for image x. The ith box in the set is formalized as $box_i = (x_i, y_i, w_i, h_i)$, where (x_i, y_i) represents the center coordinates of the bounding box, and (w_i, h_i) represents the width and height of the bounding box. Unlike DDPMs, which use Markov properties, DDIMs can define a forward process with fewer steps and achieve sampling acceleration by directly defining $q(x_{t-1}|x_t, x_0)$. According to the data distribution $x_0 \sim q(x_0)$, by gradually adding Gaussian noise with variance

$\beta_t \in (0,1)$ at each time step t, a forward Markov noise process $x_1, x_2, \ldots, x_T$ is generated for the data sample q. This allows for the direct sampling of data x_t at any time step t without repeatedly applying q.

DMNet generates bounding boxes through the application of reverse learning in diffusion models—a process that modifies the prior distribution of noise to align with the learned distribution of the bounding boxes. It gradually adds noise boxes box_t that satisfy Gaussian noise distribution to the training images, where box_0 represents the ground-truth box of the original image, with 0 indicating $T = 0$; box_t is the noise box added at time t.

The inference distribution is defined as

$$q_\sigma(box_{1:T}|box_0) = q_\sigma(box_T|box_0)\prod_{t=2}^{T} q_\sigma(box_{t-1}|box_t, box_0) \tag{1}$$

$$q_\sigma(box_T|box_0) = \mathrm{N}(\sqrt{\alpha_T}box_0, (1-\alpha_T)\mathrm{I}) \tag{2}$$

When, for $t \geq 2$,

$$q_\sigma(box_{t-1}|box_t, box_0) = \mathrm{N}(box_{t-1}; \sqrt{\alpha_{t-1}}box_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\frac{box_t - \sqrt{\alpha_t}box_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\mathrm{I}) \tag{3}$$

In the formula, a is a real number, and different settings will result in different distributions, so $q_\sigma(box_{1:T}|box_0)$ represents a series of inference distributions. Among them, $\overline{\alpha_t} := \prod_{s=0}^{t} \alpha_s$; $\overline{\alpha_t} := \prod_{s=0}^{t} \alpha_s$.

It is proved by mathematical induction that for all t, it satisfies

$$q_\sigma(box_T|box_0) = \mathrm{N}(box_t; \sqrt{\alpha_t}box_0, (1-\alpha_T)\mathrm{I}) \tag{4}$$

Calculate box_0 by sampling from $q(box_T)$ and running the inversion process $q(box_{t-1}|box_t)$.

In the generation stage, Formula (5) can be used to generate box_{t-1} from box_t:

$$box_{t-1} = \sqrt{\alpha_{t-1}}(\frac{box_t - \sqrt{1-\alpha_t}\varepsilon_\theta(box_t, t)}{\sqrt{\alpha_t}}) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\varepsilon_\theta(box_t, t) + \sigma_t\varepsilon_t \tag{5}$$

where σ_t^2 is defined as

$$\sigma_t^2 = \eta \cdot \widetilde{\beta}_t = \eta \cdot \sqrt{(1-\alpha_{t-1})/(1-\alpha_t)}\sqrt{(1-\alpha_t/\alpha_{t-1})} \tag{6}$$

When $\eta = 1$, at this time, $\sigma_t^2 = \widetilde{\beta}_t$, and the generation process is the same as the DDPM; when $\eta = 0$, the generation stage is a deterministic process. Once the initial random noise box_t is determined, the sample generation of DDIM is a deterministic process.

3.2. Proposed DMNet

As illustrated in Figure 4, we proposed the overall architecture design of the DMNet network based on a DDIM. Diffusion models require multiple runs of the model during the inference stage. Due to high complexity and difficulty in directly applying the model to the computation process of the original image in each iteration step, the model is divided into two parts: an image encoder and a detection decoder. The image encoder takes the original image as input and extracts image features. Then, the detection decoder takes the features extracted by the encoder as input, iteratively denoises the noise boxes, and finally refines the camouflaged object prediction boxes by calculating the loss between the predicted boxes and the true boxes.

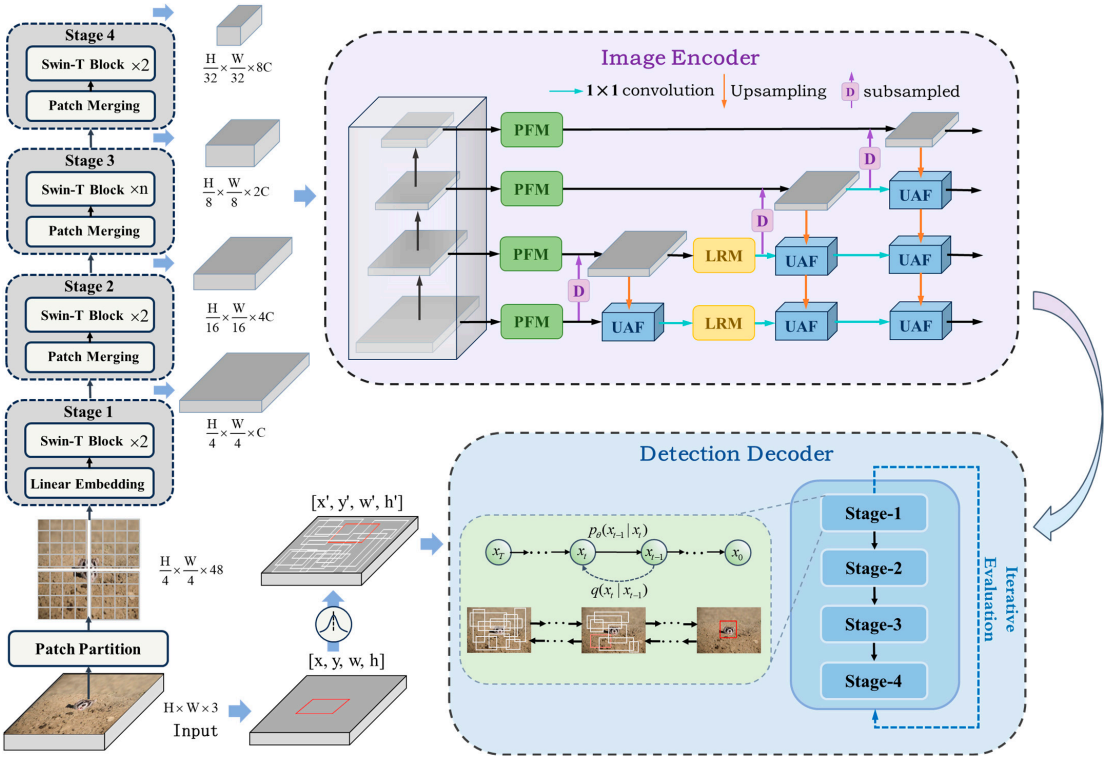


Figure 4. The overall architecture of DMNet.

3.2.1. Feature Extraction Backbone

Due to the multiple iterations required by the diffusion model to generate data samples, using the original image as input in each iteration directly would increase the computational load and complexity of the network. Therefore, DMNet first extracts camouflaged object features through the image encoder and then uses these features as input to the detection decoder for predicting detection boxes.

In the image encoder, the DMNet algorithm takes the original image as input and uses a model based on Swin Transformer to extract camouflaged object features. Similar to the hierarchical structure of convolutional neural networks (CNNs), Swin Transformer's hierarchical structure is not only flexible but is also capable of providing feature information at various scales. The specific computation process is as follows: First, input an image of size $H \times W \times 3$. Then, the input image is divided into blocks through the Patch Partition layer, with each block consisting of 4×4 adjacent pixels. Next, these blocks are concatenated in the channel dimension. Shown in Figure 5 is the structural composition of Swin Transformer (Swin-T) Block, which is capable of extracting multi-scale features of camouflaged objects.

Before entering Stage 1, each pixel's channel data undergo a linear transformation through the linear embedding layer, which employs a fully connected layer. After processing by the linear embedding layer, each image patch is converted into a feature vector of a fixed length, projecting the tensor of dimension $H/4 \times W/4 \times 48$ into an arbitrary dimension C. In DMNet, after passing through Stage 1, the dimension $H/4 \times W/4 \times 48$ is transformed into $H/4 \times W/4 \times 128$.

Four stages are used to construct feature maps of different sizes, and each time, when the sampling is downscaled by two times, the number of channels is doubled. Swin-B is a variant of the Swin Transformer model, usually referring to the Swin Transformer model with specific parameter settings. Swin-B has more parameters, which can capture more complex features, thereby enhancing the model's representational capacity.

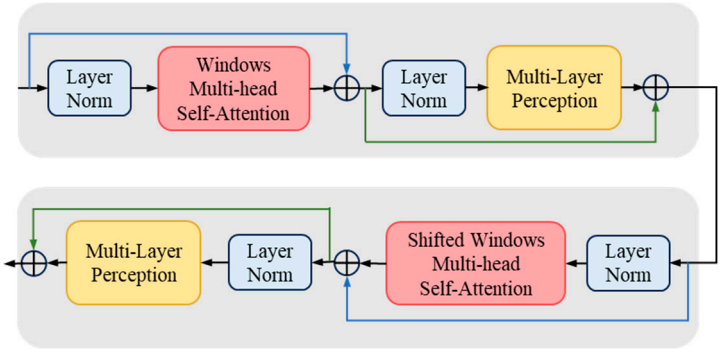


Figure 5. Swin Transformer (Swin-T) Block.

3.2.2. PFPN

We need to input the extracted camouflaged object features into the DMNet for box detection. However, due to the significant differences in features between non-adjacent stages, it is unreasonable to directly perform feature fusion on the features of the four stages. Moreover, detecting camouflaged objects often has a greater demand for low-level features such as edges and textures.

To address the above issues, in this section, we propose a progressive feature pyramid network (PFPN) that gradually fuses features from different layers to extract more information, allowing semantic information with larger differences between different layers to be approximated through progressive fusion. As shown in Figure 6, the framework extracts the last layer of features from each feature layer of the backbone, resulting in a set of features of different scales, denoted as {Z1, Z2, Z3, Z4}. Initially, we input the low-level features Z1 and Z2 into the network, then add the intermediate-level feature Z3. Finally, the high-level feature Z4 is fused. After feature fusion, a set of multi-scale features {P1, P2, P3, P4} is generated.

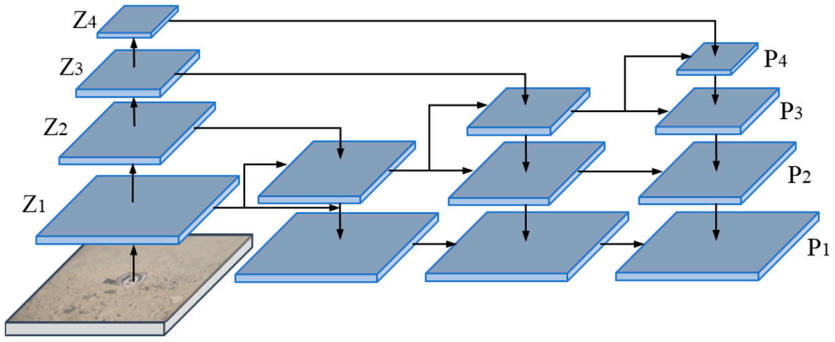


Figure 6. Progressive Feature Pyramid Network (PFPN) structure.

In order to prevent the loss of feature information during fusion between distant layers and to alleviate the conflicts between information from different levels, we fuse features from adjacent stages in the PFPN to retain useful information for features fusion. We use the PFPN to reduce this semantic gap by fusing adjacent feature layers separately, enhancing the weight of camouflaged objects, and mitigating the interference from background information that is highly similar to camouflaged objects. During the process of adjacent feature fusion, it utilizes the idea of adaptive spatial fusion operation for filtering. As shown in Figure 7, in order to balance the conflicting information between levels, we design an upsample adaptive spatial feature fusion (UAF) module to assign spatial weights for different level features.

$$f_{mn} = \alpha_{mn} \cdot x_{mn} + \beta_{mn} \cdot y_{mn} \tag{7}$$

$$\alpha_{mn} + \beta_{mn} = 1 \tag{8}$$

Here, α_{mn} and β_{mn} represent the spatial weight parameters of two different input layers, while x_{mn} and y_{mn} represent the input features of the two levels.

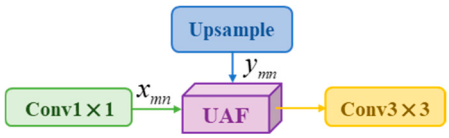


Figure 7. Upsample Adaptive Spatial Feature Fusion (UAF) module structure.

Low-level networks focus more on local detail information such as the edges and textures of camouflaged objects. However, in some cases, they may not be sufficient for completing complex tasks, such as a semantic understanding task or object-detection task. At the same time, they are prone to false alarms due to the influence of fine structures or textures in image, thereby introducing misleading information from non-camouflaged objects that can mislead detection results. On the other hand, high-level features focus more on the semantic information of objects.

To achieve multi-scale feature fusion, the high-level features are upsampled and the features of different levels are fused to retain more details of original features and enhance the spatial information of feature maps. We applied upsampling to low-level features to pass through high-level semantic features, facilitating the exploration of edge information related to camouflaged objects.

While ensuring detection accuracy, we reduce PFPN redundant-feature connections and retain the fusion of the high-level semantic information and low-level detail information of camouflaged objects through upsampling and partial downsampling processes. Lateral connections fuse the results of upsampling or are downsampled with feature maps of the same size generated in the previous stage. After fusion, we use a 3 × 3 convolution kernel to convolve the results, eliminating the aliasing effect caused by upsampling.

The PFPN utilizes gradual fusion between adjacent levels to reduce feature discrepancies in different levels. While preserving low-level features, it repeatedly fuses low-level and high-level features to generate richer feature information. By fusing low-level features with high-level features layer by layer, it avoids information loss or degradation during cross-level transmission.

3.2.3. PFM

Due to the large amount of similar structural information between the camouflaged object and its surrounding environment, it becomes challenging for the model to distinguish minute differences. In order to enlarge the receptive field and extract rich feature information from the output of the backbone network, we design a parallel fusion module (PFM). Given an input feature *f*, the PFM undergoes parallel convolution and then superposition. The parallel convolution can expand the receptive field, further enhancing the model to learn features of different scales.

We designed the PFM's structure, which is shown in Figure 8. The module architecture leverages the advantages of asymmetric convolution [23] by decomposing a standard n × n convolutional kernel into two smaller ones (typically 1 × n and n × 1). This approach significantly reduces the number of parameters while maintaining the receptive field, allowing for efficient feature extraction. In contrast to traditional convolutional operations, where the elements in the kernel are symmetric in both horizontal and vertical directions, asymmetric features in images are crucial for model performance in complex scenarios. Asymmetric convolution achieves directional feature extraction by assigning different weights, tailored to capture salient information from different orientations. The PFM process can be represented by Formulas (9) and (10): Firstly, we use a 3 × 3 convolution to enhance the features from the backbone network; then, we apply a 1 × 1 convolution to reduce the number of channels and decrease model complexity. In order to learn stronger feature information with a more pronounced receptive field, the targeted extraction of

features in different directions is performed through 1 × n and n × 1 convolutional layers. Subsequently, the features from the two branches are superimposed to capture features in different directions. Finally, a 3 × 3 convolution, normalization, and ReLU activation function (CBR) are applied to enrich contextual features. This module is capable of understanding input feature information better and enhancing local key features in different directions.

$$f_i = \mathcal{AM}(convi(f)) \tag{9}$$

$$\widetilde{f}_i = CBR(conv3(conv1(f_5)) \oplus (conv1(f_7)))) \tag{10}$$

In the formula, *conv*1 and *conv*3, respectively, represent 1 × 1 convolution and 3 × 3 convolution. *CRB* is an abbreviation for conv3 × 3 + ReLU activation function + normalization process. $\mathcal{AM}$ represents asymmetric convolution with 1 × n and n × 1 kernels, and $\oplus$ denotes element-wise addition.

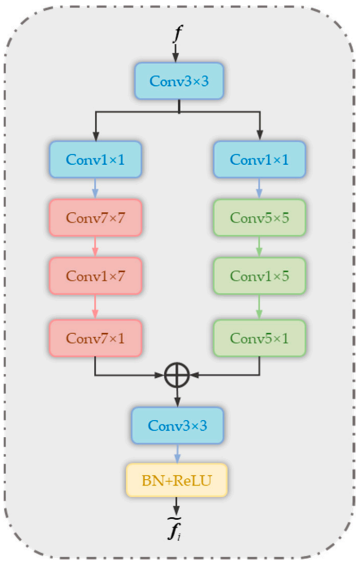


Figure 8. Parallel Fusion Module (PFM) structure.

3.2.4. LRM

We designed a location refine module (LRM) to enhance the representation learning of low-level feature details in camouflaged objects. In convolutional neural networks, as data propagate through DMNet, the features extracted by each convolutional layer can be regarded as feature representations at different levels. Low-level detail features typically refer to the feature representations extracted closer to the input layer, which are characterized by high-resolution feature maps and strong spatial geometric information-representation capabilities, making it easier to locate camouflaged objects. For example, edge cues can enhance the feature representation of the structural semantics of camouflaged objects.

The detail enhancement module is shown in Figure 9. It utilizes a 3 × 3 convolution to further capture spatial information from shallow local detail features, obtaining the initial enhanced features. Then, we apply global average pooling to aggregate spatial information. Next, channel attention is obtained through a 1D convolution operation and σ function, which reduces the number of parameters and mitigates overfitting. Finally, the channel attention is multiplied with the input features, and a 1 × 1 convolution layer is used to reduce the channels, resulting in the final output. This process can be represented by Formulas (11) and (12) as follows:

$$f_{conv3} = \mathcal{F}_{conv3}(f) \tag{11}$$

$$f_i^g = \mathcal{F}_{conv1}(\sigma(\mathcal{F}_{Conv1d}(GAP(f_{conv3}))) \otimes f_{conv3}) \tag{12}$$

In the formula, GAP stands for global average pooling, which is a pooling operation that calculates the average of each feature map across its spatial dimensions. σ represents the Sigmoid activation function. $\otimes$ represents the multiplication of elements, $\mathcal{F}_{conv1}$ is a 1×1 convolution, $\mathcal{F}_{conv3}$ is a 3×3 convolution, and $\mathcal{F}_{conv1d}$ represents a one-dimensional convolution operation.

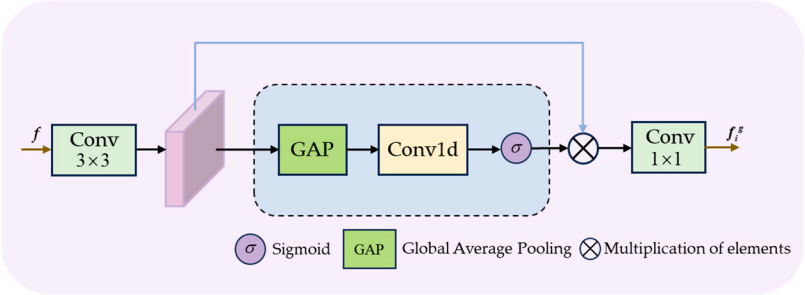


Figure 9. Illustration of the Location Refine Module (LRM) structure.

3.2.5. Detection Encoder

Figure 10 illustrates the improved detection-head structure based on the diffusion model, named DiffHead, which is more suitable for the specific task of COD. The detection decoder takes proposal boxes as input, crops Region-of-Interest (RoI) features from the feature maps generated by the image encoder, and feeds these RoI features into the detection head to identify the camouflaged objects and their locations in the image.

During the training phase, the proposal boxes are perturbed with noise from the ground-truth boxes, and during the evaluation phase, they are directly sampled from a Gaussian distribution. Given n random noisy boxes, the final prediction results are generated by extracting features from each box. The detection-head module combines a fully connected head and a convolutional head to locate camouflaged objects. The fully connected head has higher spatial sensitivity than the convolutional head, making it more capable of distinguishing the complete objects. Since the COD task focuses more on the location information of targets, using the convolutional head for object prediction can better locate these camouflaged objects.

Utilizing the idea of optimization regression, DMNet iteratively runs through multiple stages where the detection head is re-executed, and parameters are shared across different stages. The objective of a proposal box remains consistent throughout, allowing the object features from the previous stage to be reused in the next stage, providing it with richer and more informative features. This iterative process enhances the detection accuracy by refining the predictions based on the progressively enriched feature representations of the object.

When the number of stages in the detection head is excessive, although it can provide more comprehensive feature information and facilitate the fusion of multi-scale features of camouflaged objects, the model may become overly focused on detailed features while neglecting the overall structure. This can lead to the over-extraction of camouflaged object features and simultaneously increase the complexity and computational cost of the model. On the other hand, when the detection head contains fewer stages, the model structure is simpler, but its performance in complex scenarios may suffer, as it may inadequately extract target features and lose crucial information. Therefore, by comprehensively considering the model's performance and the specific requirements of the COD task, the detection head is designed to include four stages. Experimental validation is shown in Table 1, which demonstrates that setting the number of stages in the DMNet detection decoder to four yields optimal results.

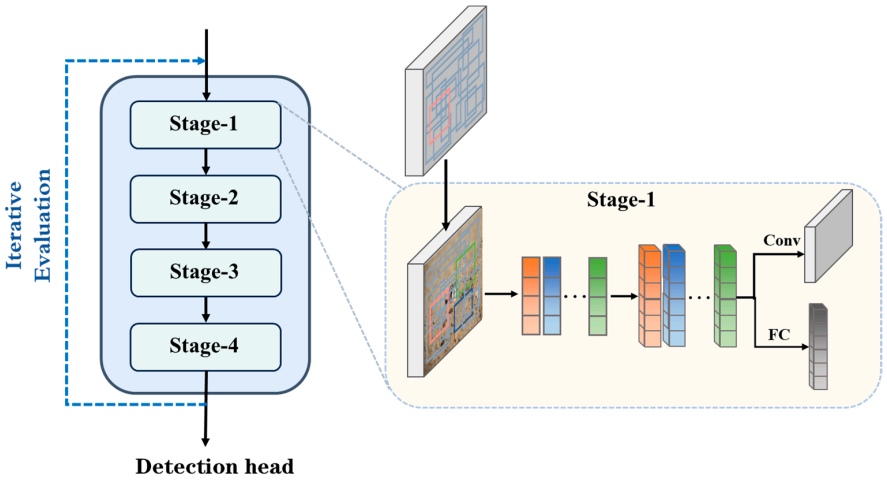


Figure 10. Architecture of the detection decoder DiffHead.

Table 1. Effect of number of decoder stages.

Stage	$AP_{50–95}$	AP_{50}	AP_{75}	APS	APM	APL
1	48.0	81.8	50.2	5.5	30.3	53.0
3	57.6	85.3	62.7	16.7	38.8	62.8
4	58.2	86.1	63.8	17.5	40.2	63.3
5	57.8	85.1	62.7	20.3	40.9	62.7
6	58.1	86.2	62.4	16.2	39.6	63.2

The values in background color are the optimal experiment results.

DiffHead replaces prediction boxes from the previous stage that are below a certain threshold with randomly generated noise boxes. It then inputs both the boxes above the threshold and the new random boxes sampled from a Gaussian distribution into the sampling process of the next stage. Filtering out more prediction boxes that are close to camouflaged objects means adding more uncertain random boxes, which increases the difficulty of prediction. However, setting the threshold too low can result in more boxes that are far from the target being input into the sampling process of the next stage, misleading the prediction results. Due to the characteristic that camouflaged objects are harder to detect than ordinary objects, setting the IoU threshold slightly below 0.5 helps retain more prediction boxes, which is beneficial for the detection of camouflaged objects. Table 2 shows the impact of different score thresholds on average precision (AP). The experimental results verify that, with other parameters being the same, setting the threshold to 0.4 yields better performance than other thresholds.

Table 2. Threshold setting.

Threshold	$AP_{50–95}$	AP_{50}	AP_{75}	APS	APM	APL
0.30	56.3	83.7	61.6	16.4	39.2	61.2
0.35	56.6	84.8	61.2	17.1	39.3	61.5
0.40	57.1	85.3	61.9	14.8	39.0	62.2
0.50	51.0	80.8	54.1	18.5	34.4	55.7

The values in background color are the optimal experiment results.

In the training phase, neural networks $f_\theta(box_t, t)$ utilize the mean-squared error (MSE) to calculate the average squared difference between predicted values and actual observations, serving as a measure of the discrepancy between them. This evaluation assesses the model's fit on the given data, ensuring that feature weights do not become excessively

large. The training objective is minimized through the loss function, enabling the model to make predictions. The formula for the loss function is as follows:

$$\mathcal{L}_{train} = \frac{1}{2}||f_\theta(box_t, t) - box_0||^2 \tag{13}$$

In the inference stage, the noise box box_T reconstructs the object prediction box box_0 through the model f_θ and iterative updating by using $box_T \rightarrow box_{T-\Delta} \rightarrow \cdots \rightarrow box_0$.

3.3. Training and Inference Strategies

3.3.1. Training Strategy

Algorithm 1 represents the pseudo-code for the training process of DMNet. During the training process, the diffusion process from the ground-truth bounding boxes to the noisy bounding boxes is first constructed, and then the model is trained in the reverse process. Due to the variable number of camouflaged objects in the images, this algorithm fills additional random bounding boxes following a Gaussian distribution into the images, so that each image has the same fixed number of bounding boxes. The noise scale is controlled by α_t, which decreases monotonically with the cosine function for different time steps t. Finally, the detection decoder takes these noisy bounding boxes as input and performs bounding-box detection on the camouflaged objects.

$$\alpha_t = \frac{f(t)}{f(0)} \tag{14}$$

$$f(t) = \cos(\frac{t/T + s}{1 + s} \cdot \frac{\pi}{2}) \tag{15}$$

In the formula, T represents the total duration; t represents the current time; and s represents the offset.

DMNet applies the set prediction loss to the recognition of camouflaged objects and the prediction of bounding box coordinates, positions, and sizes. The formula is as follows:

$$\mathcal{L} = \lambda_{cls} \cdot \mathcal{L}_{cls} + \lambda_{L1} \cdot \mathcal{L}_{L1} + \lambda_{giou} \cdot \mathcal{L}_{giou} \tag{16}$$

In the formula, $\mathcal{L}_{cls}$ is used to determine whether the target is a camouflaged object by calculating the loss between the predicted value and the ground-truth map. $\mathcal{L}_{\mathrm{L1}}$ and $\mathcal{L}_{giou}$ represent the normalized center coordinates and the loss between the predicted bounding box and the ground-truth bounding box, respectively. λ_{cls}, λ_{L1} and λ_{giou} are the weights for each component, and the optimal values are set as $\lambda_{cls} = 1.5$, $\lambda_{L1} = 5$ and $\lambda_{giou} = 2$, respectively.

Algorithm 1: Training Stage

Input : Image, box_0
ddim_training_loss (images, box_0):
# Images = (B, H, W, 3); box_0 = (B, N, 4)
# B: batch # N: number of proposal boxes
features = image_encoder (Images)
pb = padding (box_0)
for step, t in [T, . . ., 0]:
  eps = normal (mean = 0, std = 1)
  pb_crpt = sqrt(α_{cum}(t)) $*$ pb + sqrt(1 $-$ α_{cum}(t)) $*$ eps
#$\alpha_{cum} = \prod_{i=1}^{t} \alpha_i$
pb_pred = detection_decoder(pb_crpt, features, t)
loss = pre_loss (pb_pred, GT_boxes)
return loss

3.3.2. Inference Strategy

The inference process of DMNet is a denoising sampling process from noisy bounding boxes to object-prediction bounding boxes. The noisy bounding boxes are gradually refined until they match the prediction results of camouflaged objects. The pseudo-code for the sampling process is shown in Algorithm 2. After obtaining the prediction bounding boxes at the current stage, a DDIM is used to optimize and estimate the prediction bounding boxes for the next stage. At each sampling stage, random bounding boxes and prediction bounding boxes from the previous sampling stage are sent to the detection decoder to identify camouflaged objects and bounding-box coordinates. Using a DDIM can avoid the degradation of AP with more iteration steps when directly using the output prediction of the current step as input for the next step.

Algorithm 2: Inference Stage

Input: images, steps, T
ddim_sampling (images, steps, T):
steps: number of sampling steps
T: time steps
box_t = normal (mean = 0, std = 1)
times = reversed (linespace(-1, T, steps))
time_pairs = list (zip (times [: -1], times [1:]))
#box_{pre}:prediction boxes
for t_now, t_next in zip(time_pairs):
  box_{pre} = detection_decoder (eps, features, t)
  box_t = ddim_sampling (box_t, box_{pre}, t_now, t_next)
  box_t= boxes_renewal (box_t)
return box_{pre}

4. Results

4.1. Experimental Platform Configuration

The hardware platform configuration used in the experimental training and testing phase is shown in Table 3, which is as follows: The graphics card is an NVIDIA GeForce RTX 3090 with 24 GB of video memory, and we used the PyTorch deep learning development framework. The CPU is Intel Xeon Gold 6148, the operating system is Windows 10, and the Adam optimizer is used for network optimization during training.

Table 3. The hardware Platforms for model training.

Names	Related Configurations
GPU	NVIDIA GeForce RTX 3090
CPU	Xeon Gold 6148/128 G
GPU memory size	24 G
Operating system	Win 10
Computer platform	CUDA 12.2
Deep learning framework	Pytorch

4.2. Datasets and Evaluation Metrics

4.2.1. Dataset Settings

The COD10K dataset [1] was selected for the experiment. COD10K is currently the largest COS dataset, consisting of 10,000 images from openly accessible photography websites. It covers 10 super categories and 78 subcategories, including camouflaged marine creatures, flying creatures, amphibians, etc. These images include 5066 processed camouflaged images, 3000 background images, and 1934 unprocessed non-camouflaged images. The dataset divides these images into a training set and a testing set at a ratio of 6:4 and marks all subcategories as foreground objects.

4.2.2. Evaluation Metrics

The commonly used average precision (AP) is selected as the evaluation metric for testing camouflaged object detection. Here, $AP_{50–95}$ denotes the calculation of AP values within the Intersection-over-Union (IoU) threshold range set from 0.5 to 0.95, with a step size of 0.05, taking the average of these AP values; AP_{50} represents the AP value obtained with an IoU threshold of 0.5; and AP_{75} is the AP value calculated with an IoU threshold of 0.75. A larger IoU threshold indicates a higher degree of overlap between the network's predicted bounding box and the ground-truth bounding box, resulting in more accurate object localization. These three metrics can be used to evaluate the detection performance of the COD network for objects of different scales, where APS measures the average precision for small objects, representing objects with a pixel area less than 32^2; APM denotes the average precision for medium-sized objects with a pixel area between 32^2 and 96^2; and APL is the average precision for large objects with a pixel area greater than 96^2.

4.2.3. Training Settings

We set the initial learning rate to 2.5×10^{-5} and the weight decay to 10^{-4}. The data augmentation strategies include random horizontal flips, random cropping, and scale jittering by adjusting the input image size, ensuring the shortest side is at least 480 pixels and at most 800 pixels, while the longest side is at most 1333 pixels. The training iteration is set to 60 K, and the learning rate is divided by 10 at iterations of 40 K and 50 K, respectively. The number of anchor boxes, iterations, and the threshold for prediction boxes are set to 300, 4, and 0.4, respectively. The experimental data are detailed as follows:

The training process requires multiple iterations, with the number of iterations depending on factors such as the complexity of the model and the size of the dataset. Generally, a larger number of iterations leads to the better performance of the model on the training set. However, as shown in Table 4, more iterations are not always better; excessive iterations can result in overfitting when iterations = 450,000. Through experimentation, it was found that setting the number of iterations to 60,000 results in better model performance. This is shown in Table 5, and the detection accuracy is optimal when the inference step size of DMNet is set to four. To investigate how the number of training anchor boxes affects inference performance, Table 6 shows the results of experiments investigating how fixing the number of anchor boxes in DMNet affects model accuracy and training duration. By setting the number of anchor boxes to 200, 300, 500, and 2000, it was found that, with other variables held constant, DMNet performs best, and the training duration is shortest when the number of anchor boxes is set to 300.

Table 4. Setting the number of training iterations.

Iterations	$AP_{50–95}$	AP_{50}	AP_{75}	APS	APM	APL
450,000	54.9	78.9	58.2	14.6	36.7	60.6
90,000	51.1	81.1	53.6	15.7	35.3	55.7
75,000	58.0	85.0	62.7	15.5	39.5	63.1
66,000	57.0	85.2	61.7	14.9	39.0	62.0
60,000	58.1	86.2	62.4	16.2	39.6	63.2
54,000	58.1	85.8	63.1	17.8	39.2	63.2
45,000	57.1	85.3	61.9	14.8	39.0	62.2

The values in background color are the optimal experiment results.

Table 5. The sampling step of DDIM.

Step	$AP_{50–95}$	AP_{50}	AP_{75}
1	58.2	85.5	61.3
2	57.9	86.0	62.8
4	58.1	86.2	63.1
8	58.0	86.3	62.8

The values in background color are the optimal experiment results.

Table 6. The fixed number of random boxes setting.

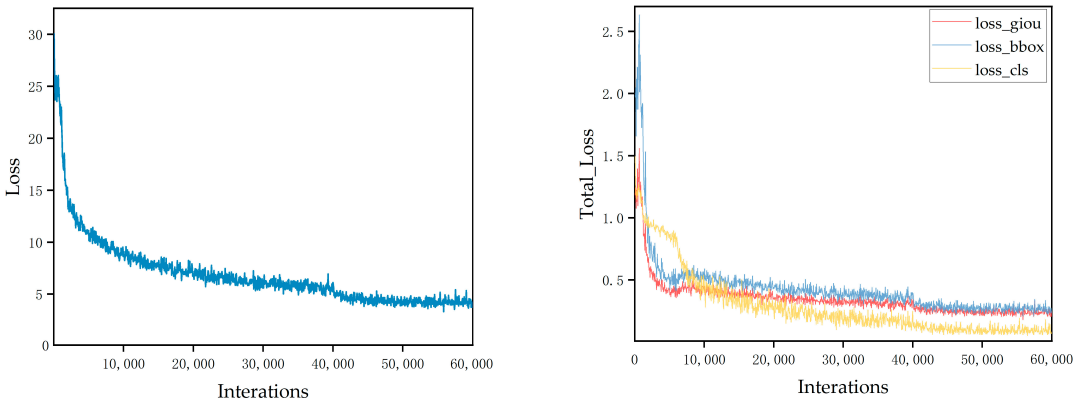
Boxes	$AP_{50–95}$	AP_{50}	AP_{75}	Train-Duration
200	50.9	79.6	54.1	8 h
300	51.0	80.8	54.1	7.5 h
500	50.5	81.0	52.9	12 h
2000	50.4	81.1	54.0	23 h

The values in background color are the optimal experiment results.

4.3. Loss Curves Experiments

Loss functions play a crucial role in object-detection tasks. By closely monitoring their relationship with the number of training iterations, we can optimize training process, effectively improving the performance and stability of DMNet.

As can be observed from Figure 11, the evolution of the loss function is visualized using data from each training iteration, the loss values of DMNet converged at 60,000 training iterations. When the number of iterations increases, the value of the loss function tends to decrease gradually. This is because during the continuous learning and optimization process, the predicted information gradually approaches the true labels, resulting in a gradual reduction in the loss value.



(**a**) (**b**)

Figure 11. Loss function graph for experiments. (**a**) Loss curve. (**b**) The loss curves group consist of classification loss (loss_cls), bounding box loss (loss_bbox) and generalized intersection over union Loss(loss_giou).

4.4. Comparison Algorithms

To demonstrate the effectiveness of the proposed DMNet in this paper, in this section, we compare DMNet with 15 classic and advanced object-detection algorithms. To ensure a fair comparison of the detection performance of the algorithms, all algorithms are evaluated using the experimental configuration platform proposed in Section 3.1, with experimental parameters following the default settings.

4.5. Analysis of the Comparative Experimental Results

4.5.1. Quantitative Comparison

Table 7 presents the experimental comparison results of the DMNet algorithm proposed in this paper with other 15 algorithms on the COD10K dataset, mainly demonstrating the feature extraction capability of this network and the accuracy of detecting camouflaged objects. As can be seen from Table 7, DMNet achieves the best performance in six evaluation metrics, with the best overall ability to detect camouflaged objects, where the $AP_{50–95}$ reaches 58.2%—an improvement of 8% compared to the Swin-RCNN algorithm. By observing the last two columns in Table 6, it can be determined that DMNet has significant advantages in detecting medium and large objects. Among them, the metrics AP_{75} and

APM exceed the Swin-RCNN algorithm by about 10%, which again reflects the effectiveness and accuracy of DMNet in detecting camouflaged objects.

Table 7. Comparative experiment result on the COD10K dataset.

Methods	Pub. Year	$AP_{50–95}$	AP_{50}	AP_{75}	APS	APM	APL
YOLACT [24]	ICCV2019	36.5	69.8	34.6	6.1	19.6	41.1
Cascade RCNN [25]	TPAMI2019	46.3	73.5	48.4	8.4	27.9	51.4
BlendMask [26]	CVPR2020	43.6	68.6	45.5	7.6	25.2	48.9
ATSS [27]	ECCV2020	45.0	73.6	45.3	11.9	30.4	49.1
CondInst [28]	ECCV2020	42.8	69.4	44.6	5.8	24.7	47.9
SparseRCNN [9]	CVPR2021	46.5	74.9	47.9	12.4	34.3	50.4
SCNet [29]	AAAI2021	47.1	75.5	48.2	13.8	29.0	52.0
Swin-RCNN [10]	ICCV2021	50.2	79.8	54.6	11.3	31.9	55.3
Tood [30]	ICCV2021	47.9	73.9	49.0	12.0	32.0	52.4
VFNet [31]	CVPR2021	46.6	73.9	48.0	7.9	31.2	51.1
MaskTrans [32]	CVPR2022	46.3	70.0	48.8	3.5	27.7	51.4
Centernet [33]	Arxiv2022	42.3	72.3	42.2	11.0	24.7	47.3
MPVIT-RCNN [34]	CVPR2022	57.8	82.3	63.3	17.2	38.1	62.9
CO-DETR [35]	ECCV2023	44.5	63.5	32.3	8.3	20.9	37.4
DINO [36]	ICLR2023	37.1	62.1	30.6	10.5	19.6	30.0
DiffCOD	**ours**	**58.2**	**86.1**	**63.8**	**17.5**	**40.2**	**63.3**

The values in bold are the optimal detection results

4.5.2. Qualitative Comparison

DMNet utilizes a diffusion model-based detection head named DiffHead, specifically designed for the COD task, to guide the iterative denoising of noisy random boxes, thereby obtaining camouflaged object-prediction boxes and enhancing detection accuracy. The PFM module designed in DMNet incorporates asymmetric convolution in parallel, which can enhance the features extracted by backbone. The LRM enables DMNet to more effectively focus on difficult-to-recognize detailed features in the image, resulting in higher precision when locating camouflaged objects. Figure 12 shows the visualization results of the comparative experiments on the COD10K dataset in this section. It can be observed that DMNet can accurately detect camouflaged small objects that are obscured. The iterative feature adaptive fusion construction of the PFPN and UAF can better extract and fuse multi-scale features in images, obtain spatial information of camouflaged objects, and thereby improve the problem of missed detection in multi-target detection in COD tasks to enhance the accuracy of camouflaged object detection. As can be seen from Figure 12, in situations where multiple camouflaged objects exist simultaneously, DMNet is able to detect the camouflaged objects in the image completely and accurately. Qualitative analysis shows that the DMNet algorithm proposed in this paper is more suitable for COD tasks compared to other detection algorithms.

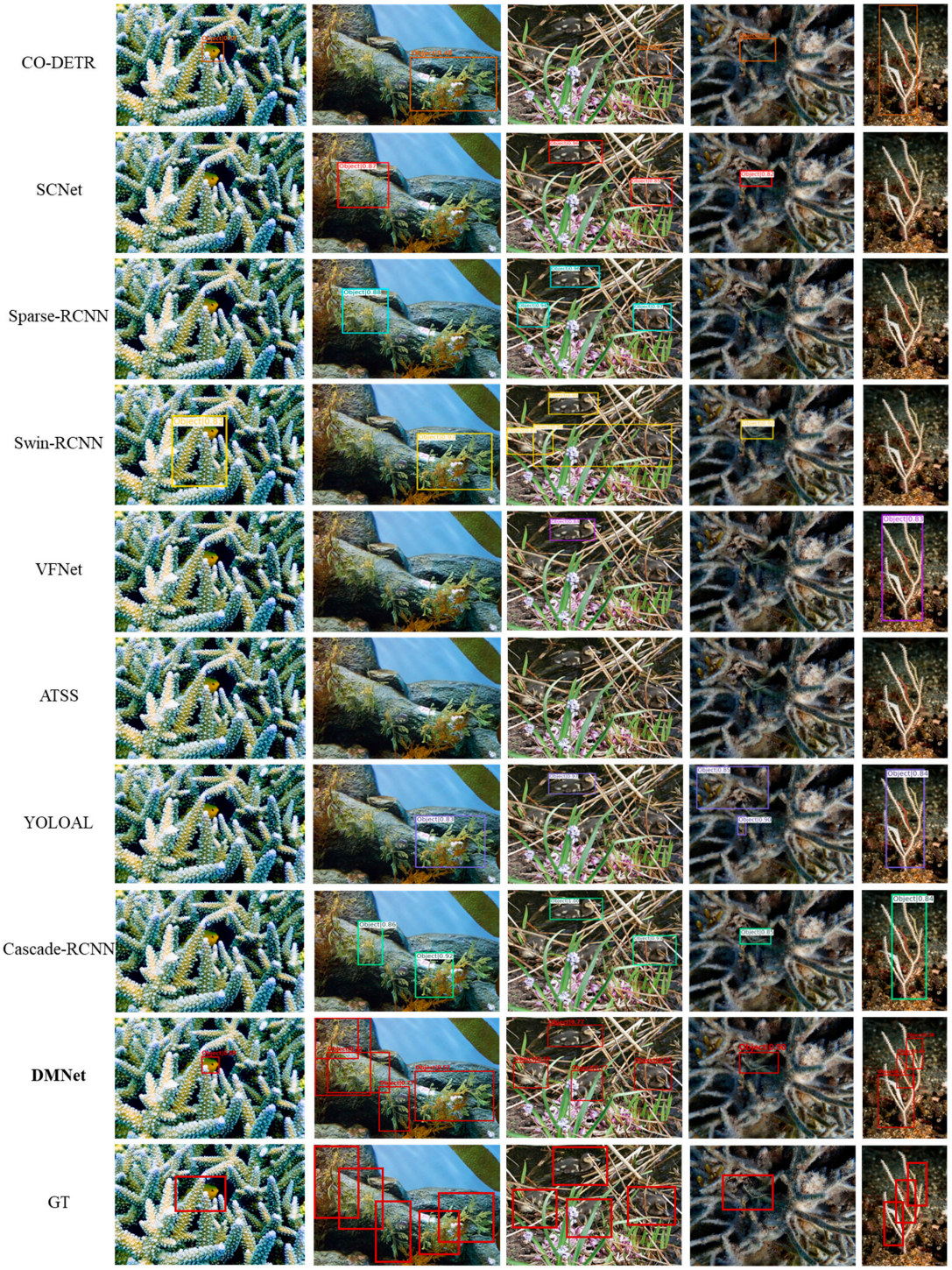


Figure 12. The visualization results obtained by different algorithms comparison experiments.

4.6. Ablation Experiments

In this section, ablation experiments are designed to verify the effectiveness of the key components of DMNet, including the PFPN, LRM, PFM, and DiffHead. All experiments are conducted on the COD10K dataset, and the experimental results are shown in Table 8, where the baseline network is the DiffusionNet algorithm.

Table 8. Ablation experiment.

Baseline	PFPN	PFM	LRM	DiffHead	$AP_{50–95}$	AP_{50}	AP_{75}	APS	APM	APL
√					51.8	80.8	54.1	13.7	31.8	57.2
√	√				57.5	83.4	62.1	10.5	39.5	61.6
√	√	√			58.1	84.6	61.9	16.0	39.3	63.3
√	√	√	√		58.1	84.2	62.4	16.2	39.6	62.4
√	√	√	√	√	58.2	86.1	63.8	17.5	40.2	63.3

Table 8 comprehensively displays the vertical comparison results of the ablation experiments conducted using DMNet. The effectiveness of each submodule in improving model performance is verified by testing the PFPN, LRM, PFM, and DiffHead separately. By adding the LRM and PFM to the network, the detection accuracy is improved to some extent. However, since the LRM focuses on detailed information in the image and neglects global features, it results in slight information loss, leading to a slight reduction in detection accuracy for large objects. Finally, after adding the improved DiffHead, all AP values are increased by approximately 1%. Compared with the baseline network, the addition of designed components improves the accuracy of DMNet.

4.7. Extended Experiments

The research on COD is of great significance. To prove that the proposed DMNet in this paper is still effective when extended to generalized COD, in this section, we present extended experiments on DMNet, aiming to verify the effectiveness and applicability of the algorithm in practical military domains.

Figure 13 showcases the visualization results of DMNet detecting military camouflaged objects, where the images are sourced from the MiCOD dataset [37] and the NC4K dataset [38]. Through observation and analysis, it can be concluded that DMNet possesses high capability in detecting military camouflaged objects, thereby demonstrating the accuracy and practicality of DMNet.

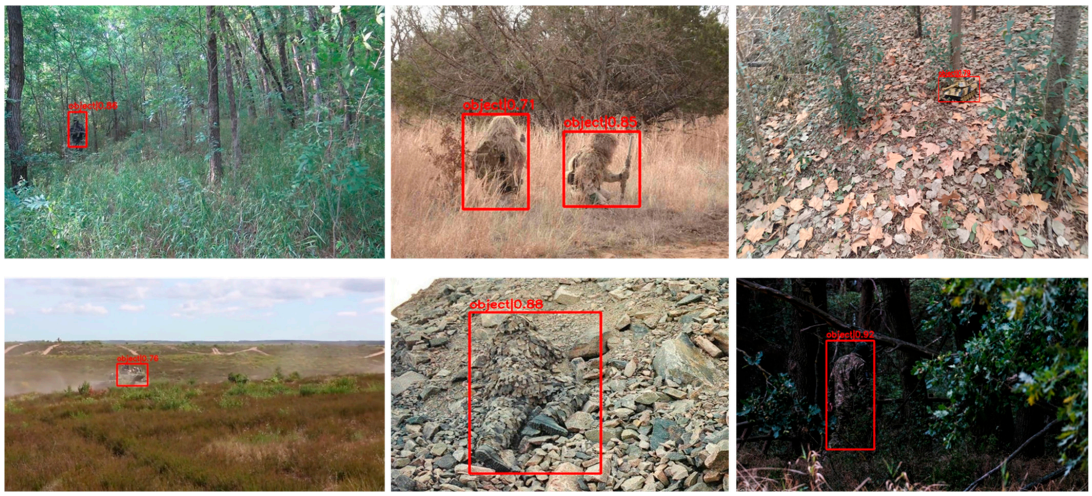


Figure 13. Detection results of military camouflaged objects.

5. Discussions

The results from the benchmark tests discussed in Section 4 indicate that our method has achieved excellent performance in the domain of camouflaged object detection. Although it advances beyond previous studies, there are still multiple aspects that warrant further exploration and discussion.

5.1. Advantages of DMNet

While mainstream object-detection models currently exhibit excellent performance, the task of camouflaged object detection differs significantly from conventional object detection. Due to the high similarity between camouflaged objects and their backgrounds, as well as the characteristics of blurred boundaries and confusing coloration inherent in camouflaged objects, they are more difficult to detect compared to conventional objects. Therefore, this paper proposes a camouflaged object-detection method based on a DDIM, constructs a DMNet network, and introduces a diffusion model to assist the network in better capturing image details, thereby solving the problem of low detection accuracy.

The diffusion model-guided proposal boxes differ significantly from those generated by conventional models in that they do not require heuristic object priors or learnable queries, which further simplifies target-candidate generation and furthers the development of the detection pipeline. Traditional object-detection methods mostly rely on a fixed set of learnable queries or predefined anchors, whereas the camouflage object-detection network based on the diffusion model directly detects objects from random boxes, minimizing the reliance on prior knowledge. Through experimental analysis, DMNet is able to reduce the missed detection rate of camouflaged objects in complex environments and improve detection accuracy. In summary, DMNet exhibits superior performance and is more suitable for detecting camouflaged objects in complex scenes.

5.2. Limitations and Challenge

First and foremost, the experimental results indicate that while our current model outperforms most existing object-detection models in terms of detection performance, there are still some issues that need to be addressed. We have observed that our model may occasionally produce false alarms, and when the camouflaged objects are relatively small, the detection accuracy of DMNet remains unsatisfactory.

Secondly, the inference speed of detection tasks based on diffusion models is relatively slow due to the fact that diffusion models encompass two primary processes: the forward process, which gradually adds noise boxes to the original image, and the reverse process, which progressively removes noise boxes to generate prediction boxes that conform to the target distribution. Diffusion models optimize application results through iterative noise processing, achieving satisfactory high-quality outcomes. However, this process consumes significant time, computation, and storage costs during both training and inference. This is because they rely not only on hundreds or thousands of diffusion steps but also require network evaluations at each step of the sampling process to refine the results. This substantially increases the computational and storage costs during training and inference, thereby limiting the widespread application of our models.

Consequently, we will further research and improve upon enhancing the detection accuracy of small-scale camouflaged objects and lightweighting the diffusion model for camouflaged object detection.

6. Conclusions

In this paper, we introduce a diffusion model for camouflage object detection and develop a DMNet to address the issue of low detection accuracy in COD. This network treats the COD task as a denoising diffusion process from the noise box to the detection box. In the training phase, Gaussian noise is added to the true bounding box to obtain a noisy random box, and in the inference phase, the model iteratively refines the noisy random box into a camouflage object-prediction box. A lateral comparison utilizing the challenging COD10K dataset reveals that DMNet possesses substantial performance benefits in the realm of camouflaged object detection (COD). Notably, DMNet's AP_{50} metric surpasses MPVIT by a margin of 4%, while its AP_{75} metric outperforms Swin-RCNN by an impressive 9.2%. Ablation experiments validate the efficacy of each suggested module. Moreover, extended experiments were undertaken to affirm the prowess of DMNet in detecting military camouflaged objects.

In future work, we plan to introduce depth information and frequency information to refine the detection effect and solve the problem of the incomplete detection of camouflaged objects when the occlusion area is large. Finally, we will further investigate the diffusion model for COD to promote the translation of research findings into practical applications.

Author Contributions: Conceptualization, W.C. and W.G.; methodology, W.G.; validation, W.G., X.J.; formal analysis, X.W.; investigation, X.D.; resources, W.C.; data curation, X.J.; writing—original draft preparation, W.G.; writing—review and editing, W.C.; visualization, X.W.; supervision, X.D.; project administration, W.C. All authors have read and agreed to the published version of the manuscript.

Funding: Basic Strengthening Plan Field Fund (Grant No. 2021-JCJQ-JJ-0871) for funding our experiments.

Data Availability Statement: Data related to the current study are available from the corresponding author upon reasonable request.

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

References

1. Fan, D.P.; Ji, G.P.; Sun, G.; Cheng, M.M.; Shen, J.; Shao, L. Camouflaged object detection. In Proceedings of the 2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 2774–2784.
2. MacDonald, D.; Isenman, J.; Roman, J. Radar detection of hidden targets. In Proceedings of the IEEE 1997 National Aerospace and Electronics Conference (NAECON), Dayton, OH, USA, 14–17 July 1997; pp. 846–855.
3. Gautam, A.K.; Preet, P.; Rawat, T.S.; Chowdhury, R.P.; Sinha, L.K. *Detection of Camouflaged Targets in Hyperspectral Images*; Springer: Singapore, 2020; pp. 155–161. [CrossRef]
4. Shen, Y.; Lin, W.; Wang, Z.; Li, J.; Sun, X.; Wu, X.; Wang, S.; Huang, F. Rapid Detection of Camouflaged Artificial Target Based on Polarization Imaging and Deep Learning. *IEEE Photonics J.* **2021**, *13*, 1–9. [CrossRef]
5. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.A.; Ramanan, D. Object Detection with Discriminatively Trained Part Based Models. *IEEE Trans. Pattern Anal. Mach Intell.* **2010**, *32*, 1627–1645. [CrossRef] [PubMed]
6. Redmon, J.; Divvala, S.K.; Girshick, R.B.; Farhadi, A. You Only Look Once: Unified; Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
7. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision–ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
8. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10778–10787.
9. Sun, P.; Zhang, R.; Jiang, Y.; Kong, T.; Xu, C.; Zhan, W.; Tomizuka, M.; Li, L.; Yuan, Z.; Wang, C. Sparse R-CNN: End-to-End Object Detection with Learnable Proposals. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 14449–14458.
10. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Nashville, TN, USA, 20–25 June 2021; pp. 9992–10002.
11. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach Intell.* **2014**, *37*, 1904–1916. [CrossRef] [PubMed]
12. Wang, W.J.; Lu, Y.H.; Zheng, G.C.; Zhan, S.G.; Ye, X.Q.; Tan, Z.C.; Wang, J.D.; Wang, G.A.; Li, X. BEVSpread: Spread Voxel Pooling for Bird's-Eye-View Representation in Vision-based Roadside 3D Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024; pp. 14718–14727.
13. Tu, Y.; Zhang, B.; Liu, L.; Li, Y.; Chen, X.; Zhang, J.; Wang, Y.; Wang, C.; Zhao, C.R. Self-supervised Feature Adaptation for 3D Industrial Anomaly Detection. *arXiv* **2024**, arXiv:2401.03145.
14. la Fuente, R.P.-D.; Delclòs, X.; Peñalver, E.; Speranza, M.; Wierzchos, J.; Ascaso, C.; Engel, M.S. Early evolution and ecology of camouflage in insects. *Proc. Natl. Acad. Sci. USA* **2012**, *109*, 21414–21419. [CrossRef] [PubMed]
15. Avrahami, O.; Lischinski, D.; Fried, O. Blended Diffusion for Text-driven Editing of Natural Images. *arXiv* **2022**, arXiv:2111.14818.
16. Wang, T.F.; Zhang, B.; Zhang, T.; Gu, S.Y.; Bao, J.M.; Baltrusaitis, T.; Shen, J.J.; Chen, D.; Wen, F.; Chen, Q.F.; et al. RODIN: A Generative Model for Sculpting 3D Digital Avatars Using Diffusion. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 4563–4573.
17. Qian, H.; Huang, W.J.; Tu, S.K.; Xu, L. KGDiff: Towards explainable target-aware molecule generation with knowledge guidance. *Brief Bioinform.* **2023**, *25*, 435. [CrossRef] [PubMed]
18. Esser, P.; Chiu, J.; Atighehchian, P.; Granskog, J.; Germanidis, A. Structure and Content-Guided Video Synthesis with Diffusion Models. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 7312–7322.

19. Chung, H.; Sim, B.; Ryu, D.; Ye, J.C. Improving Diffusion Models for Inverse Problems using Manifold Constraints. *arXiv* **2022**, arXiv:2206.00941.
20. Wu, J.; Fang, H.; Zhang, Y.; Yang, Y.; Xu, Y. MedSegDiff: Medical Image Segmentation with Diffusion Probabilistic Model. *arXiv* **2022**, arXiv:2211.00611.
21. Zhao, P.A.; Li, H.; Jin, R.Y.; Zhou, S.K. DiffULD: Diffusive Universal Lesion Detection. In Proceedings of the 26th International Conference on Vancouver, Vancouver, BC, Canada, 8–12 October 2023; pp. 94–105.
22. Lv, W.; Huang, Y.; Zhang, N.; Lin, R.; Han, M.; Zeng, D. DiffMOT: A Real-time Diffusion-based Multiple Object Tracker with Non-linear Prediction. *arXiv* **2024**, arXiv:2403.02075.
23. Ding, X.H.; Guo, Y.C.; Ding, G.G.; Han, J.G. ACNet: Strengthening the Kernel Skeletons for Powerful CNN via Asymmetric Convolution Blocks. *arXiv* **2019**, arXiv:1908.03930.
24. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. YOLACT: Real-Time Instance Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27–28 October 2019; pp. 9156–9165.
25. Cai, Z.; Vasconcelos, N. Cascade R-CNN: Delving Into High Quality Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
26. Chen, H.; Sun, K.; Tian, Z.; Shen, C.; Huang, Y.; Yan, Y. BlendMask: Top-Down Meets Bottom-Up for Instance Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8570–8578.
27. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S. Bridging the Gap Between Anchor-Based and Anchor-Free Detection via Adaptive Training Sample Selection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 9756–9765.
28. Tian, Z.; Shen, C.; Chen, H. Conditional Convolutions for Instance Segmentation. In Proceedings of the Computer Vision–ECCV 2020, Glasgow, UK, 23–28 August 2020; pp. 282–298.
29. Vu, T.; Kang, H.; Yoo, C.D. SCNet: Training Inference Sample Consistency for Instance Segmentation. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 2701–2709. [CrossRef]
30. Feng, C.; Zhong, Y.; Gao, Y.; Scott, M.R.; Huang, W. TOOD: Task-aligned One-stage Object Detection. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 11–17 October 2021; pp. 3490–3499.
31. Zhang, H.; Wang, Y.; Dayoub, F.; Sunderhauf, N. VarifocalNet: An IoU-aware Dense Object Detector. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 8510–8519.
32. Ke, L.; Danelljan, M.; Li, X.; Tai, Y.W.; Tang, C.K.; Yu, F. Mask Transfiner for High-Quality Instance Segmentation. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–20 June 2022.
33. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. CenterNet: Keypoint Triplets for Object Detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27–28 October 2019; pp. 6568–6577.
34. Lee, Y.; Kim, J.; Willette, J.; Hwang, S.J. MPViT: Multi-Path Vision Transformer for Dense Prediction. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–20 June 2022; pp. 7277–7286.
35. Zong, Z.F.; Song, G.L.; Liu, Y. DETRs with Collaborative Hybrid Assignments Training. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 6725–6735.
36. Zhang, H.; Li, F.; Liu, S.L.; Zhang, L.; Su, H.; Zhu, J.; Ni, L.M.; Shum, H.Y. DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection. *arXiv* **2022**, arXiv:2203.03605.
37. Jiang, X.H.; Cai, W.; Zhang, Z.L.; Jiang, B.; Yang, Z.Y.; Wang, X. Camouflaged object segmentation based on COSNet. *Acta Arma.* **2023**, *44*, 1456–1468.
38. Lv, Y.; Zhang, J.; Dai, Y.; Li, A.; Liu, B.; Barnes, N. Simultaneously localize; segment and rank the camouflaged objects. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 11586–11596.