


# Domain Similarity-Perceived Label Assignment for Domain Generalized Underwater Object Detection

Xisheng Li<sup>1</sup>, Wei Li<sup>2</sup>, Pinhao Song<sup>3</sup>, Mingjun Zhang<sup>4</sup> and Guifang Sun<sup>5</sup>

<sup>1</sup> School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China

<sup>2</sup> Robotics Research Group, the Department of Mechanical Engineering, KU Leuven, Belgium

<sup>3</sup> School of Mechanical Engineering, Southeast University, Nanjing, P.R.China  
6213113079@stu.jiangnan.edu.cn

**Abstract.** The inherent characteristics and light fluctuations of water bodies give rise to the huge difference between different layers and regions in underwater environments. When the test set is collected in a different marine area from the training set, the issue of domain shift emerges, significantly compromising the model's ability to generalize. The Domain Adversarial Learning (DAL) training strategy has been previously utilized to tackle such challenges. However, DAL heavily depends on manually one-hot domain labels, which implies no difference among the samples in the same domain. Such an assumption results in the instability of DAL. This paper introduces the concept of Domain Similarity-Perceived Label Assignment (DSP). The domain label for each image is regarded as its similarity to the specified domains. Through domain-specific data augmentation techniques, we achieved state-of-the-art results on the underwater cross-domain object detection benchmark S-UODAC2020. Furthermore, we validated the effectiveness of our method in the Cityscapes dataset.

**Keywords:** Domain adversarial learning, Underwater object detection, Pseudo domain label.

## 1 Introduction

Object detection is a critical task in computer vision that aims to automatically identify specific objects in images or videos and precisely locate them. It has significant applications in intelligent surveillance, autonomous driving, and robot navigation. Traditional object detection algorithms [1-4], assume that the training and testing datasets are sampled from the same distribution, sharing similarities in image features, scene settings, and data collection methods. However, underwater scenarios present unique challenges. The testing dataset often deviates from the training dataset due to variations in lighting conditions, color distortion, light attenuation (coastal, deep oceanic, murky waters), and camera settings [5-6].

Domain generalization (DG) is a concept in machine learning that involves training a model on data from multiple different but related domains so that it can perform

well on unseen domains. The actual application in the real underwater environment matches the DG definition due to light fluctuation and attenuation in different water bodies. However, underwater cross-domain scenarios have received comparatively limited attention. In the scarce studies on cross-domain underwater object detection, researchers have incorporated domain generalization training strategies, leading to significant improvements in cross-domain scenarios [7].

Domain Adversarial Learning (DAL) as proposed by [8], employs domain adversarial learning to align features across underwater cross-domain scenarios, which is widely used in DG. This approach significantly improves the generalization capability of detection models in underwater environments, resulting in enhanced performance. However, DAL faces certain challenges in its application to cross-domain object detection underwater. (i) The existing DG dataset utilizes domain labels annotated manually for applying DAL. The real world includes a mixture of domains that are difficult to explicitly annotate. (ii) Even with a significant investment of human resources in annotating the dataset with discrete domain labels, obtaining favorable detection outcomes proves challenging. This issue arises from a phenomenon emphasized in [9], wherein the high similarity between two domains, when artificially assigned distinct domain labels, can negatively impact the training stability of DAL. Specifically, the backbone may extract highly similar features from these two domains, leading to the domain discriminator overfitting to these inaccurately labeled examples. This, in turn, compromises the model's generalization ability [10]. (iii) Additionally, Domain Data Augmentation (DDA) is a commonly employed technique in DG problems. However, as we expand the number of domains, manually labeling domain labels becomes an obstacle, leading to situations where similar images carry different true labels. This can potentially introduce an over-confidence problem, affecting the stability of the adversarial training. Consequently, it's challenging for existing Domain-Adversarial Learning (DAL) methods to more effectively integrate domain data augmentation.

To address the aforementioned issue, we propose Domain Similarity-Perceived Label Assignment (DSP), which eliminates the need for manual annotations. The central concept of our approach is to perceive a domain as a blend of similarities with various other domains. Each domain is regarded as a sample within a continuous space, enabling the direct generation of distinct pseudo-domain labels for individual images. Inspired by Farthest Point Sampling [11], we leverage Farthest Feature Sampling (FFS) to autonomously curate a set of base domains from the source domain without requiring input from the dataset. Subsequently, the domain label for an image is determined by its similarity to this set of base domains. The objective of designing the DSP module is to train a domain classifier capable of discerning among these base domains. Performing inference using a trained domain classifier, pseudo domain labels can be generated and presented in a soft label format, as opposed to discrete labels. This labeling approach enhances the stability of the DAL training process.

It can be concluded that a detector trained across a wide range of domains demonstrates domain invariance. Therefore, increasing sampling across the domain distribution contributes to enhanced robustness against domain shifts.[7] Consequently, Domain Data Augmentation (DDA) has emerged as a crucial technique in DG. However, since DDA generates images belonging to various domains, annotating domain labels

becomes impractical. By training DSP, we can uncover similarities within the newly generated domains. Leveraging the similarity among a few domains with maximum style differences allows us to effectively represent the remaining domains. We employ the Spurious Correlations Generator (SCG) [12] to generate a significantly larger number of domains compared to the original set, and then apply our DSP to label these domains. By combining SCG, DSP, and DAL, we achieved state-of-the-art results in underwater cross-domain object detection benchmark S-UODAC2020. Furthermore, we validated the effectiveness of our approach on the more general scenario of Cityscapes.

## 2 Methods

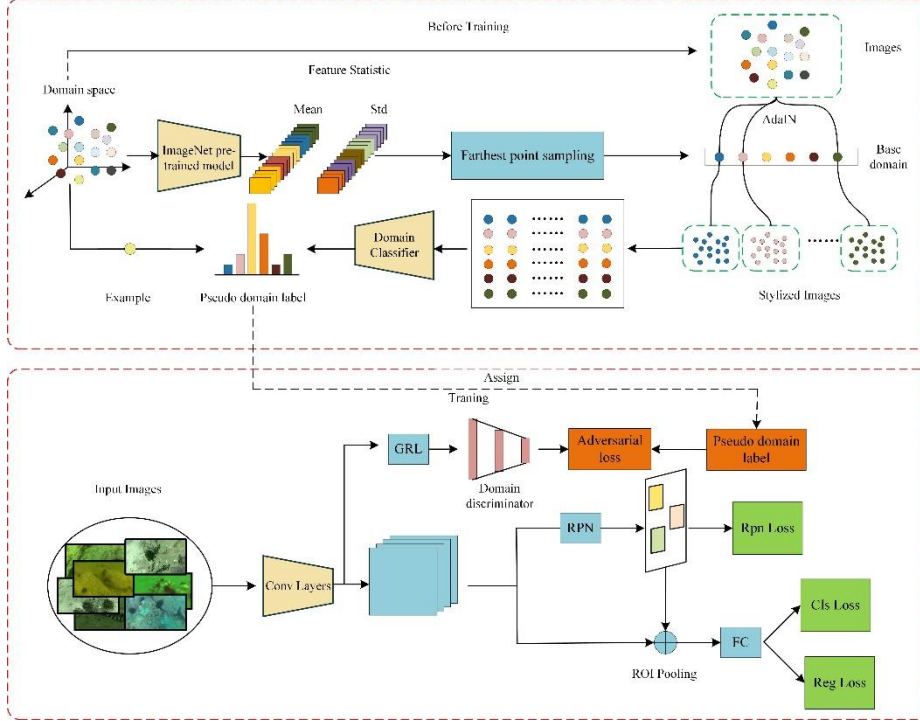
### 2.1 Overview

Following the usual terms for domain generalization, we define  $\{x_i\}_{i=1}^N \in \mathcal{D}_S$  where  $x_i$  are the samples,  $\mathcal{D}_S$  is the source domain. In the typical DG problem, the source domain can be manually divided into several subsets to obtain domain labels. However, this assumption is not practical in the actual application because the real-world data is a mixture of domains that is difficult to annotate. Therefore, we introduce Domain Similarity-Perceived Label Assignment (DSP) to construct  $K$  most dissimilar source domains within the dataset  $\mathcal{D}_S = \{\mathcal{D}_{S_1}, \dots, \mathcal{D}_{S_K}\}$  and use them as a reference to generate pseudo-domain labels for each image. Consequently, DAL can be trained even without the domain labels in the training dataset.

### 2.2 Domain similarity-perceived label assignment (DSP)

We aim to obtain pseudo domain labels for every image  $\bar{y}_i = f(x_i) \in \mathbb{R}^K$ .  $f(\cdot)$  is the proposed label assignment function. As illustrated in Fig. 1, DSP utilizes Farthest Feature Sampling (FFS) to obtain this set of base domains. AdaIN is employed to expand the number of base domains for training the domain classifier. Inference with this domain classifier allows us to obtain a pseudo-domain label for each image.

**Farthest Feature Sampling (FFS).** To construct a set of the base domains, we aim to find images that are most different from each other in style. Inspired by Farthest Point Sampling which is used to downsample the point cloud, we propose Farthest Feature Sampling (FFS). The entire process can be found in Algorithm 1. The goal of FFS is to select  $K$  images from the dataset that exhibit the farthest style distances from each other, as  $\{\bar{x}_k\}_{k=1}^K = \text{FFS}(\{x_i\}_{i=1}^N)$ . Various works [13-15] suggest that The convolutional feature statistics encode the style in an image, which can be used to calculate the style distance. In detail, we first passed all data through the pre-trained model, took the low-level features, then calculated their mean and standard deviation  $\mathcal{F} = \{(\mu(\phi(x_1)), \sigma(\phi(x_2))), \dots, (\mu(\phi(x_N)), \sigma(\phi(x_N)))\}$ ,  $\phi$  denote the low-level layer in backbone. In this way, each image can correspond to a set of feature statistics. We aim to use these statistical features to find the  $K$  domains that best represent the training set. Secondly, select a random image as the starting point, denoted as  $\hat{d}_0$ , and add it to the



**Fig. 1.** DSP is a preprocessing module used before model training. It utilizes a pre-trained model from ImageNet to extract low-level semantic information from images and aggregate their feature statistics. These statistics are then stacked together, and base domains are selected from them using the Farthest Feature Sampling method. Subsequently, each base domain's image quantity is augmented using Adaptive Instance Normalization (AdaIN). These augmented images are fed into the Domain Classifier. Finally, the Input Images are passed through the Domain Classifier for inference, yielding domain labels.

set  $C$ , which means including already selected domains for storage. We denote  $K$  as the intended number of base domains. Calculate the style distances from  $\mathcal{F}$  to the selected set  $C$  and add the image with the farthest distance to  $C$ . After  $K$  steps, we have  $K$  images as base domains.

**Real-time Arbitrary Style Transfer (AdaIN).** The pseudo domain label for each image is determined by its similarity to the set of  $K$  images. Hence, we need to train a classifier capable of distinguishing these  $K$  base domains. We have only one image for each base domain, which is insufficient for training a domain classifier. Therefore, we need to transform our existing images through style transfer to match the style of the base domain images. The goal of style transfer is to create a new image that is based on the content of one image but rendered in the style of another. AdaIN combines Instance Normalization with style transfer and it allows us to adaptively adjust the features of an input image based on the style of a reference image. It computes the mean and variance of the input features and then rescales these features using the mean and variance of the reference image to match its style. In this way, the input image will be stylized into the

---

**Algorithm 1: Farthest Feature Sampling**


---

**Given:**  $\{\mathbf{x}_i\}_{i=1}^N \in \mathcal{D}_S$  : all images in source domain.  $\phi$  is a shallow layer feature extractor.

**Result:** Obtain K images with the farthest style distances

**Initialization:**

1. Compute feature statistics

$$\mu_n = \mu(\phi(\mathbf{x}_N)), \sigma_n = \sigma(\phi(\mathbf{x}_N))$$

$$\mathcal{F} = \{(\mu_1, \sigma_1), \dots, (\mu_2, \sigma_2), \dots, (\mu_N, \sigma_N)\}$$

2.  $S \in \mathbb{R}^N$  stores distances from selected set  $C$  to set  $\mathcal{D}_S$ , which is initialized to  $\infty$ .

3. Select a random image  $\tilde{\mathbf{x}} \in \mathcal{D}_S, \tilde{\mathbf{x}} \rightarrow C$

**for**  $i \leftarrow 0$  **to**  $N - 1$  **do**

    Calculate style distances  $d$  between  $\mathcal{F}$  and  $C$  using

**for**  $j \leftarrow 0$  **to**  $N - 1$  **do**

$$d_j = \sum_{\mathbf{x}_l \in C} \|\mathcal{F}_j - (\mu(\phi(\mathbf{x}_l)), \sigma(\phi(\mathbf{x}_l)))\|^2$$

$$S_j = d_j \text{ if } S_j > d_j$$

**end**

$$x_k \rightarrow C, \text{ where } k = \operatorname{argmax}_j(S_j)$$

**end**

---

input image based on the style of a reference image. It computes the mean and variance of the input features and then rescales these features using the mean and variance of the reference image to match its style. In this way, the input image will be stylized into the style of the reference image. After obtaining the image of K base domains using FFS, we intend to train a domain classifier by transforming the images of the dataset into representations of these styles using AdaIN, as:

$$\tilde{f}_i^k = \sigma(\phi(\bar{x}_k)) \frac{\phi(x_i) - \mu(\phi(x_i))}{\sigma(\phi(x_i))} + \mu(\phi(\bar{x}_k)) \quad (1)$$

$$\tilde{x}_i^k = \phi^{-1}(\tilde{f}_i^k) \quad (2)$$

where  $\phi(\cdot)$  and  $\phi^{-1}(\cdot)$  are the ImageNet pre-trained model and the inverse decoding model, respectively.  $\tilde{x}_i^k$  is the stylized  $x_i$  by style  $\bar{x}_k$ . With the stylized dataset, the domain classifier can be trained with the goal as:

$$f = \operatorname{argmin} \sum_{i=1}^N \sum_{k=1}^K \log \tilde{y}_i^k(k) \quad (3)$$

$$\tilde{y}_i^k = f(\tilde{x}_i^k) \in \mathbb{R}^K \quad (4)$$

With the trained domain classifier (label assignment function)  $f(\cdot)$ , a unique domain label for each image can be obtained for DAL.

**Domain Adversarial Learning (DAL).** DAL is a conventional approach for capturing shared characteristics among diverse domains. It achieves this by maximizing the cost of the domain discriminator. Our approach differs from traditional DAL in that we do not aim to confuse artificially defined discrete domains. Instead, we aim to perturb the entire domain space represented by all images in the training set. The domain adversarial loss can be written as:

$$\mathcal{L}_{dal} = \max_h \sum_{i=1}^N C E \left( h(g(x_i)), f(x_i) \right) \quad (5)$$

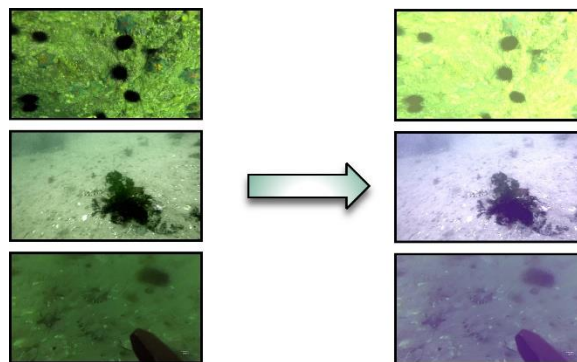
where  $N$  denotes the total number of images,  $g(\cdot)$  for backbone,  $h(\cdot)$  for domain discriminator. The domain discriminator aims to maximize the loss with pseudo-domain labels, while the backbone aims to confuse the domain discriminator. We add this loss in the object detection task, and obtain the total loss as:

$$\mathcal{L}_{tot} = \mathcal{L}_{rpn} + \mathcal{L}_{cls} + \mathcal{L}_{loc} + \lambda * \mathcal{L}_{dal} \quad (6)$$

where  $\mathcal{L}_{rpn}$  denoting the Region Proposal Network (RPN) loss, and  $\mathcal{L}_{cls}$  and  $\mathcal{L}_{loc}$  stand for the classification loss and bounding-box regression loss, respectively. The parameter  $\lambda$  represents a hyper-parameter that necessitates fine-tuning to achieve optimal model performance.

**Data Augmentation.** Data augmentation helps alleviate domain shift by enriching the diversity of image styles. It can effectively expand the variety of styles in cases where the training set is either monotonous or limited in style. However, as the number of styles increases, manual labeling becomes more challenging, making it difficult to leverage DAL techniques. In contrast, with DSP, we could represent a greater number of domains using a smaller subset of domains. This implies that data augmentation methods could seamlessly integrate with DAL, even when dealing with a broader range of styles. To achieve this, we employed SCG[12] to generate a greater variety of stylistic

images, see in fig.2. Specifically, SCG applied the Discrete Cosine Transform (DCT) to the input images, transforming them into the frequency domain, and then blended them with randomly generated reference images in the frequency domain. We employed SCG\* as our method, and what sets it apart from the original SCG in the research paper is that it exclusively manipulates the low-frequency information to obtain new styles, without any modifications to the high-frequency details. This was accomplished by adjusting the blending parameters to enhance stylistic diversity. As we acquired more domains, resulting in a more domain-invariant representation. This approach leverages the introduction of more stylistic diversity, thereby enhancing the model's adaptability to various domains and features.



**Fig. 2.** SCG\* is a method that generates variations of each image in the dataset by solely altering the low-frequency information while preserving the core content but introducing different styles.

### 3 Experimental Results And Analysis

#### 3.1 Experiments on the S-UODAC2020 benchmark

**Experimental setup.** The experiments were conducted with NVIDIA GeForce RTX 3080Ti GPU. Following benchmark S-UODAC2020, the training set consists of six different domains, labeled as type1 to type6, while the evaluation and testing are conducted on the seventh domain, referred to as type7, as depicted in Fig.3. We have chosen the classic Faster R-CNN model as our detector, augmented with a Feature Pyramid Network (FPN) to enhance its detection capabilities. For the implementation of Faster R-CNN, we have used the mmdetection [16] version 2.24.1 framework. The backbone chosen for this model is ResNet-50. We set the batch size to 1 and trained the model for 12 epochs. The optimizer used is SGD (Stochastic Gradient Descent) with a learning rate of 0.005, weight decay of 0.0001, and momentum of 0.9. We did not employ multi-scale training, and all images were resized to a consistent size of (1333, 800) pixels during training. Only the horizontal-flip data augmentation method is employed unless specified otherwise. We extract 64-channel feature maps from the backbone for feature statistics. For the training of the domain classifier in the DSP module, we opt for the

utilization of 128 base domains. We iterate through 800 training iterations, then inferring the domain classifier for each image, ultimately obtaining a 128-dimensional domain label for each image. As for the trade-off parameter  $\lambda$  in domain adversarial learning, we have selected a value of 0.7.

**Comparison with other domain generalization methods.** The comparison of domain generalization methods is shown in Table 1. Faster R-CNN + FPN approach exhibits limited generalization capabilities, while Mixup, a commonly employed data augmentation technique, has demonstrated limited effectiveness in underwater scenarios and can even lead to adverse effects. In contrast, DANN showcases exceptional performance, holding significant advantages over other methods. However because of the constraints posed by discrete, our approach consistently outperforms them, outperforms them even with a ResNet101 backbone. In the S-UODAC2020 benchmark, our method outperforms all others, establishing itself as state-of-the-art in this benchmark.



**Fig. 3.** S-UODAC2020 is a dataset for underwater cross-domain object detection, comprising four marine species: echinus, holothurian, scallop, and starfish. The training set consists of 4,745 images sourced from six distinct domains, while the test set comprises 797 images from domains distinct from those in the training set.

### 3.2 Experiments on the Cityscapes benchmark

We utilized the Cityscapes [17] for our experiments. The training dataset consisted of 19,395 daytime-sunny images sourced from BDD100K. Our test dataset comprised 26,158 night-sunny images from BDD100K, along with an additional 3,775 images collected from the Foggy Cityscapes and Adverse-Weather datasets. This approach allowed us to assess whether our method, trained primarily on readily available data, could perform effectively under more challenging conditions. We used the results obtained with [17] as our baseline. To align with its methodology, we employed the Faster R-CNN+FPN object detection network, with ResNet-101 serving as the backbone detector. The image size was set to have a minimum side length of 600 pixels. Our Faster R-CNN implementation was based on mmdetection version 2.24, with a learning rate of 0.0025, weight decay set at 0.0001, and a momentum of 0.9. For the DSP, the hyperparameters were maintained identical to those used in the S-UODAC2020 dataset, with the solitary exception being the training iterations for the domain classifier, which were set to 200. In total, our model underwent 10 epochs of training to achieve the final results. Due to variations in the implementation of Faster R-CNN, there may be differences in the values of FPN. In the table, we use "FPN\*" to denote this. As we can observe in Table 2, in the night-sunny environment, FPN outperforms all other domain generalization methods.



**Table 1.** The performance of various domain generalization methods on the benchmark S-UODAC2020, where 'Ave' represents mAP50.

Method	Backbone	Epochs	Input Size	Ave(%)
DeepAll	ResNet50	12	1333*800	48.86%
DG-YOLO	DarkNet53	300	416*416	39.24%
DMCL	ResNet50	12	512*512	61.36%
DANN	ResNet50	12	1333*800	53.87%
DANN(r101)	ResNet101	24	1333*800	56.18%
CIDDG	ResNet101	12	1333*800	54.60%
JiGEN	ResNet101	12	768*768	55.20%
Ours	ResNet50	12	1333*800	<b>61.88%</b>
方法	Echinus(%)	Starfish(%)	Holo (%)	Scallop(%)
DeepAll	74.79%	36.59%	43.12%	34.54%
DG-YOLO	62.74%	26.83%	32.84%	18.86%
DMCL	78.44%	54.62%	53.15%	59.23%
DANN	<b>78.62%</b>	42.76%	50.60%	43.48%
DANN(r101)	73.23%	49.92%	50.96%	50.61%
CIDDG	74.04%	48.98%	49.71%	45.67%
JiGEN	75.92%	47.01%	51.37%	46.50%
Ours	76.27%	<b>57.23%</b>	<b>53.59%</b>	<b>60.41%</b>

except for our own. Despite the differences in the values between our re-implemented FPN\* and FPN, our approach still surpasses the best-performing methods. In the day-time-foggy scenario as in Table 3, our method exhibits overwhelming superiority.

**Table 2.** The performance of various domain generalization methods on the night-sunny

Method	bus	bike	car	motor	person	rider	truck	mAP
FPN	37.4	33.1	62.2	21.4	42.5	32.1	40.9	38.6
FPN*	42.1	35.0	65.9	18.0	48.0	31.9	44.8	40.8
SW	35.4	28.6	56.7	18.4	38.2	26.2	39.3	34.7
IBN-Net	40.2	31.4	62.1	19.0	42.9	29.3	44.2	38.4
IterNorm	28.8	29.2	55.7	12.3	35.9	25.4	35.4	31.8
ISW	37.4	32.2	60.4	16.5	41.0	29.2	43.0	37.1
Ours	<b>43.1</b>	<b>37.4</b>	<b>66.0</b>	20.0	<b>50.1</b>	<b>31.9</b>	<b>46.7</b>	<b>42.2</b>

**Table 3.** The performance of various domain generalization methods on the daytime-foggy

Method	bus	bike	car	motor	person	rider	truck	mAP
FPN	30.5	29.7	52.1	28.4	33.9	40.4	21.0	33.7
FPN*	27.7	30.0	56.6	29.0	36.2	38.9	20.7	34.2
SW	32.0	28.4	52.3	28.8	33.5	39.5	21.9	33.8
IBN-Net	32.5	31.4	52.5	31.1	38.0	42.1	23.5	35.9
IterNorm	25.3	27.4	50.4	24.0	32.2	37.4	18.6	30.7
ISW	31.9	30.5	51.9	30.8	37.5	40.9	21.9	35.1
Ours	<b>35.0</b>	<b>31.5</b>	<b>60.1</b>	<b>33.8</b>	<b>40.0</b>	41.3	<b>23.8</b>	<b>37.9</b>

over the remaining methods.

### 3.3 Ablation studies

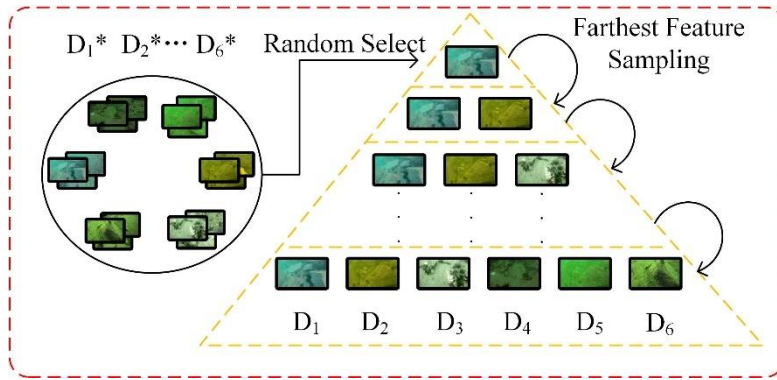
**Pesudo labels analysis.** When the number of the base domains selected matches the artificially partitioned domains in the dataset as shown in Fig.4, the base domains chosen by DSP align with the manually designated domains. For example, the selected  $D_1$  resembles  $D_1^*$ . As illustrated in Fig.5, longer training of DSP results in an over-confident label prediction, while shorter training of DSP can smooth the label and capture more relations between different domains. Table 4 studies the training iterations of DSP. The optimal performance is achieved when DSP reaches 800 iterations. A remarkable finding is that, even without the use of any manual annotations, the pseudo labels generated by DSP surpass the performance of manually annotated one-hot labels and even outperform the results obtained by softening the manual annotations using ELS. This model underscores the capability of DSP to provide more accurate domain labels for domain adversarial training.

**The number of the base domains.** In the previous section, we set the base domains to exactly match the manually partitioned domains. We further tested scenarios with fewer or more domains selected by DSP. As demonstrated in the last row of Table 5, even when the number of base domains is reduced to 2, the proposed method still outperforms the baseline DeepAll ( $K=0$ ). The domain diversity in S-UODAC dataset is limited so we only set  $K$  less than the numbers of source domains.

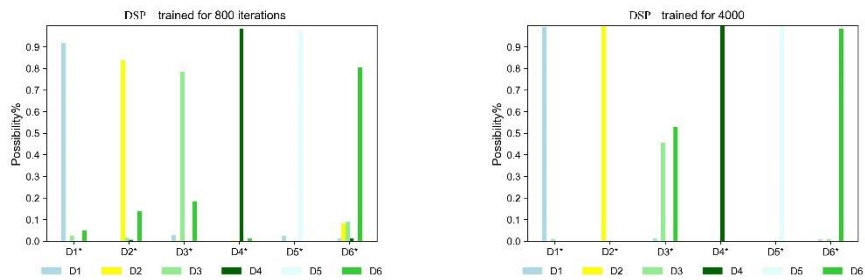
**Combining domain data augmentation with DSP.** Domain data augmentation can largely enrich the domain diversity in the dataset, DSP can leverage that to further improve the performance. We employed SCG\* in different datasets to generate various styles of images. From the results in Table 6, using SCG\* along can improve the performance because it enriches the training data. Using DSP alone can improve the performance in S-UODAC2020, while DSP provides less improvement in both Sunny -> Foggy and Sunny -> Night scenarios of the Cityscapes.

**Table 4.** The performance of various types of domain labels, including one-hot encoding, Environment Label Smoothing (ELS), and labels obtained under various DSP training epochs.

Labeling Method	mAP	Labeling Method	mAP
DeepAll	48.86	DANN(one-hot)	53.87
DANN (ELS)	52.61	DANN(DSP_100)	54.21
DANN (DSP_500)	51.67	DANN(DSP_800)	54.50
DANN (DSP_1000)	51.40	DANN(DSP_4500)	52.31



**Fig. 4.** When the number of base domains equals the manually partitioned domain count, utilizing Farthest Feature Sampling can identify domains  $D_1$  through  $D_6$  within the source domains  $D_1^*, D_2^*, \dots, D_6^*$ .



**Fig. 5.**  $D_i^*$  denotes an image intentionally labeled as the  $i$ -th domain by humans, while  $D_i$  represents the  $i$ -th base domain selected by DSP.  $D_i^*$  is considered as a probability combination of  $D_1$  to  $D_6$ . (left) Pseudo domain labels obtained after 4000 iterations of DSP training. (right) Pseudo domain labels obtained after 800 iterations of DSP training.

**Table 5.** Ablation study of the number of the base domains  $K$  in S-UODAC2020 dataset.  $K=0$  denotes DeepAll.

$K$	0	2	3	4	5
mAP	48.9	51.9	53.0	52.2	52.7

dataset than S-UODAC2020 because they are single-source domain generalization problems. This result shows the dependence of DSP on the domain diversity in the dataset. If we combine SCG\* with DSP, we can further improve the performance in all three datasets because DSP effectively excavates the invariant features from diverse training domains.

**T-SNE visualization.** T-SNE visualization was employed to observe feature distributions after applying different methods, which is shown in Fig.4. We extracted features from the final layer of ResNet50, selecting 300 random images from the source domain and 100 from the target domain for visualization. In the graph, blue dots represent the target domain, while red dots represent the source domain. As depicted in the visualization, our approach brings the feature distances between the two domains closer, establishing a connection between the tasks of detecting the source and target domains. Consequently, this approach yields the best results.

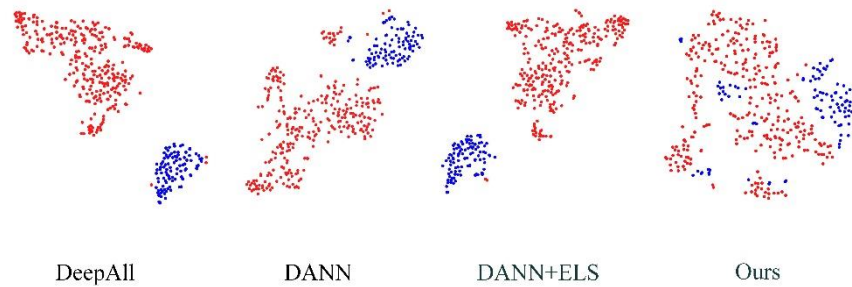
**Visualization of detection results.** We present the most visually compelling results as shown in Fig.5. Both vanilla Faster R-CNN and DANN exhibit false positive detections when recognizing objects in the second image, whereas ELS experiences a significant number of false negatives. In contrast, our method not only identifies a greater number of target objects but also avoids any false positive detections. Consequently, if our model is utilized for underwater exploration in unfamiliar environments, it has the potential to identify a greater number of aquatic organisms present in the water.

**Table 6.** Ablation study of SCG\* across different datasets.  $K$  is set to 128.

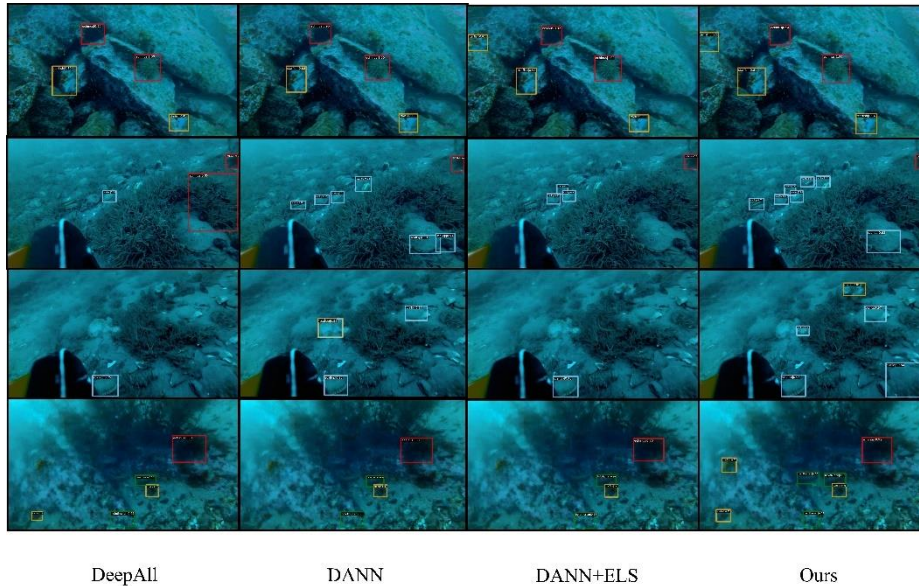
I.S-UODAC2020				
Method	DeepAll	DSP w/o SCG*	SCG*-only	DSP + SCG*
mAP	48.9	54.5	60.2	61.9
II.Sunny->Foggy				
Method	DeepAll	DSP w/o SCG*	SCG*-only	DSP + SCG*
mAP	34.2	35.4	36.5	37.9
III.Sunny->Night				
Method	DeepAll	DSP w/o SCG*	SCG*-only	DSP + SCG*
mAP	40.8	41.4	41.5	42.2

## 4 Conclusion

This paper aims to address challenges faced by domain adversarial training in underwater scenes, where over-confident discrete manual domain labels lead to the instability of adversarial training. We propose the Domain Similarity-Perceived Label Assignment (DSP), representing each image based on its similarity to a set of base domains. The proposed approach demonstrates outstanding performance on the S-UODAC2020 and Cityscape datasets. The results suggest that smooth and continuous label space can effectively improve the performance of domain adversarial training. We believe that the applicability of DSP extends beyond this, as it can be employed in various directions such as cross-domain pedestrian re-identification.



**Fig. 6.** Visualizing different methods using t-SNE. Red points denote data from the source domain, while blue points represent data from the target domain.



**Fig. 7.** Comparison of actual detection results under different methods. Different colored boxes represent the discovery of various underwater creatures. The red box signifies the presence of echinus, the blue box represents scallop, the yellow box indicates starfish, and the green box denotes holothurian.

## References

1. Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016: 21-37.
2. Girshick R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448
3. Cai Z, Vasconcelos N. Cascade r-cnn: Delving into high quality object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6154-6162.
4. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.
5. P. M. Uplavikar, Z. Wu, and Z. Wang, “All-in-one underwater image enhancement using domain-adversarial learning,” in CVPR workshops, 2019, pp. 1–8.
6. H. Liu, P. Song, and R. Ding, “Wqt and dg-yolo: Towards domain generalization in underwater object detection,” arXiv preprint arXiv:2004.06333, 2020.
7. Y. Chen, P. Song, H. Liu, L. Dai, X. Zhang, R. Ding, and S. Li, “Achieving domain generalization for underwater object detection by domain mixup and contrastive learning,” *Neurocomputing*, vol. 528, pp. 20–34, 2023.
8. Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.
9. Zhang Y F, Wang X, Liang J, et al. Free lunch for domain adversarial training: Environment label smoothing[J]. arXiv preprint arXiv:2302.00194, 2023
10. H. Thanh-Tung, T. Tran, and S. Venkatesh, “Improving generalization and stability of generative adversarial networks,” arXiv preprint arXiv:1902.03984, 2019.
11. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *Advances in neural information processing systems*, vol. 30, 2017.
12. Xu M, Qin L, Chen W, et al. Multi-view adversarial discriminator: Mine the non-causal factors for object detection in unseen domains[C]//Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. 2023: 8103-8112.
13. L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.
14. C. Li and M. Wand, “Combining markov random fields and convolutional neural networks for image synthesis,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2479–2486.
15. Y. Li, N. Wang, J. Liu, and X. Hou, “Demystifying neural style transfer,” arXiv preprint arXiv:1701.01036, 2017.
16. K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu et al., “Mmdetection: Open mmlab detection toolbox and benchmark,” arXiv preprint arXiv:1906.07155, 2019.

17. A. Wu and C. Deng, "Single-domain generalized object detection in urban scene via cyclic-disentangled self-distillation," in Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, 2022, pp. 847–856.