Efficient Test-time Adaptive Object Detection via Sensitivity-Guided Pruning

Kunyu Wang, Xueyang Fu, Xin Lu, Chengjie Ge, Chengzhi Cao, Wei Zhai, Zheng-Jun Zha*
School of Information Science and Technology and
MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition,
University of Science and Technology of China, Hefei, 230026, China
{kunyuwang, luxion, cjge, chengzhicao}@mail.ustc.edu.cn
{xyfu, wzhai056, zhazj}@ustc.edu.cn

Abstract

Continual test-time adaptive object detection (CTTA-OD) aims to online adapt a source pre-trained detector to ever-changing environments during inference under continuous domain shifts. Most existing CTTA-OD methods prioritize effectiveness while overlooking computational efficiency, which is crucial for resource-constrained scenarios. In this paper, we propose an efficient CTTA-OD method via pruning. Our motivation stems from the observation that not all learned source features are beneficial; certain domain-sensitive feature channels can adversely affect target domain performance. Inspired by this, we introduce a sensitivity-guided channel pruning strategy that quantifies each channel based on its sensitivity to domain discrepancies at both image and instance levels. We apply weighted sparsity regularization to selectively suppress and prune these sensitive channels, focusing adaptation efforts on invariant ones. Additionally, we introduce a stochastic channel reactivation mechanism to restore pruned channels, enabling recovery of potentially useful features and mitigating the risks of early pruning. Extensive experiments on three benchmarks show that our method achieves superior adaptation performance while reducing computational overhead by 12% in FLOPs compared to the recent SOTA method.

1. Introduction

Object detection [39, 40] is a fundamental task in computer vision with numerous downstream applications. However, in real-world scenarios such as autonomous driving [37] or robotics [45, 46], various natural factors, such as adverse weather conditions and lighting changes [14, 15], can lead to substantial domain shifts between training and testing data, resulting in significant performance degradation. Moreover, due to the dynamic nature of real-world environments, the test data distribution is constantly changing and inherently unpredictable, posing additional challenges. To address this issue, continual test-time adaptation [41] has been proposed, which aims to adapt a source pre-trained model during inference to accommodate the evolving test data, providing a promising solution to mitigate the domain shift problem, as shown in Fig 1.

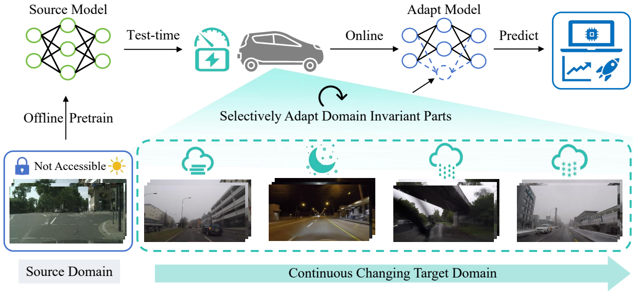

Figure 1. We consider continual test-time adaptive object detection, which adapts a source detector to changing environments during inference. By focusing adaptation efforts on invariant parts while pruning sensitive parts, we improve computational efficiency, which is crucial for resource-constrained scenarios like autonomous driving or UAV.

While existing continual test-time adaptive object detection methods show impressive results [4, 31, 35], enhancing computational efficiency has been relatively underexplored, despite its critical importance for resource-constrained scenarios [1, 28, 36]. Current methods typically adapt all features learned from the source domain indiscriminately during adaptation to the target domain. However, our exploratory experiments reveal that not all learned source features are beneficial for target domain; in fact, certain source feature channels negatively affect the model's performance in the target domain.

In particular, we perform ablation on feature channels of the source pre-trained model, observing the impact of

[*] Corresponding author. This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 62225207, 62436008, 62422609 and 62276243.

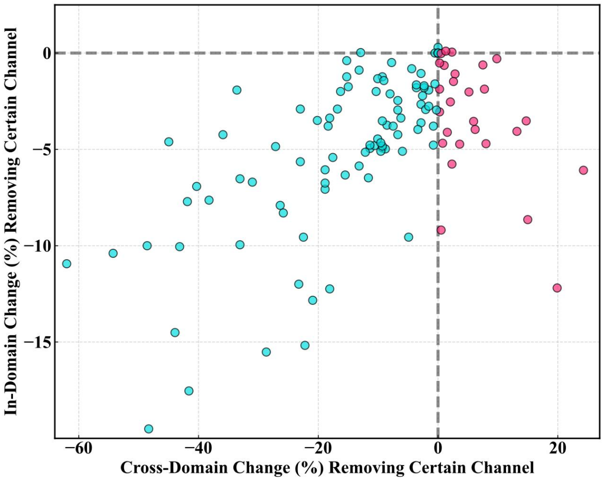

Figure 2. We analyze how removing certain feature channel from the source model affects in-domain and cross-domain performance. The source model is trained on the Cityscapes training set and evaluated in-domain on the Cityscapes validation set and cross-domain on Cityscapes-C. Points on the positive x-axis (or y-axis) indicate that removing certain channel improves cross-domain (or in-domain) performance, while those on the negative x-axis (or y-axis) indicate a decrease in cross-domain (or in-domain) performance when removed. The axis units denote the performance change percentage relative to direct-test.

removing certain channel on both in-domain and cross-domain performance. The results indicate that some feature channels in source model, marked as red points in Fig. 2, contribute positively to in-domain testing, where their removal causes a drop in in-domain performance. Conversely, these channels negatively impact cross-domain testing, as removing them enhances cross-domain performance. This observation reveals the an indiscriminate adaptation strategy is inefficient, as allocating computational resources to such domain-sensitive channels not only increases adaptation difficulty but also reduces efficiency.

Inspired by the success of network pruning in removing ineffective subsets to improve efficiency, we explore the potential of pruning similar "ineffective subsets", namely feature channels sensitive to domain shifts, while focusing adaptation efforts on invariant ones to enhance adaptation efficiency. To this end, we introduce a sensitivity-guided channel pruning that quantifies the weight of each feature channel based on its sensitivity to domain discrepancies. Specifically, we compute channel sensitivity weights at both the image level and instance level by measuring the discrepancy between target domain features and pre-collected source domain statistics. Subsequently, we apply a weighted sparsity regularization to the learnable parameters, encouraging the pruning of channels with higher sensitivity weights according to predefined threshold and ratio. Moreover, to prevent the loss of potentially useful channels due to early pruning, we design a stochastic channel reactivation mechanism. This mechanism utilizes Bernoulli sampling to randomly restore pruned channels, allowing the model to reassess their utility. By jointly optimizing the adaptation loss and the channel pruning loss, our method focuses on invariant features, thereby ensuring superior adaptation performance on the changing target domain while significantly reducing computational overhead.

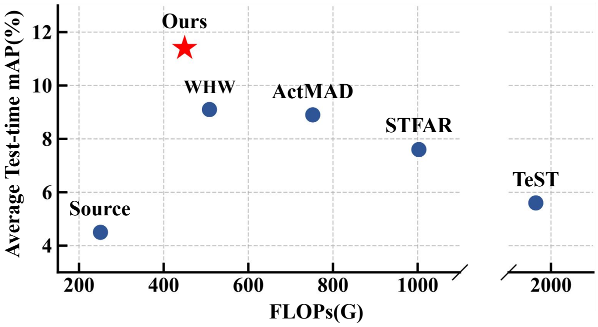

Figure 3. We perform continual online adaptation on Cityscapes → Cityscapes-C. The x-axis and y-axis denote the average test-time mAP across all corruptions and the total FLOPs consumption (including both forward and backward passes). Our method achieves superior results with the least FLOPs consumption.

In summary, our contributions are listed as follows:

- We unveil a novel perspective for enhancing CTTA-OD efficiency from the model structure viewpoint, inspired by the observation that certain source sensitive feature channels negatively impact target domain performance.
- We propose a novel efficient CTTA-OD framework that suppresses and prunes sensitive feature channels and selectively adapts invariant ones, reducing adaptation difficulty and enhancing computational efficiency.
- We propose a sensitivity-guided channel pruning strategy that quantifies each channel's sensitivity to domain discrepancies at both image and instance levels, along with a stochastic reactivation mechanism to prevent early pruning of useful channels.

Extensive experiments on three benchmarks show that our method achieves superior adaptation results while reducing 12% FLOPs compared to the SOTA method, as shown in Fig 3.

2. Related Work

Test-time Adaptive Object Detection (TTA-OD) [5, 7, 10, 23, 25, 27, 38, 43, 47] aims to adapt a source pre-trained detector to an unlabeled target domain without access to the original source data, addressing privacy concerns. To mitigate the absence of source data, most TTA-OD approaches adopt a self-training paradigm. For instance, SED [25] is the first approach to introduce pseudo-label self-training via self-entropy descent. A^2SFOD [7] incorporates an adversarial module into a mean teacher framework to align feature spaces between source-similar and source-dissimilar

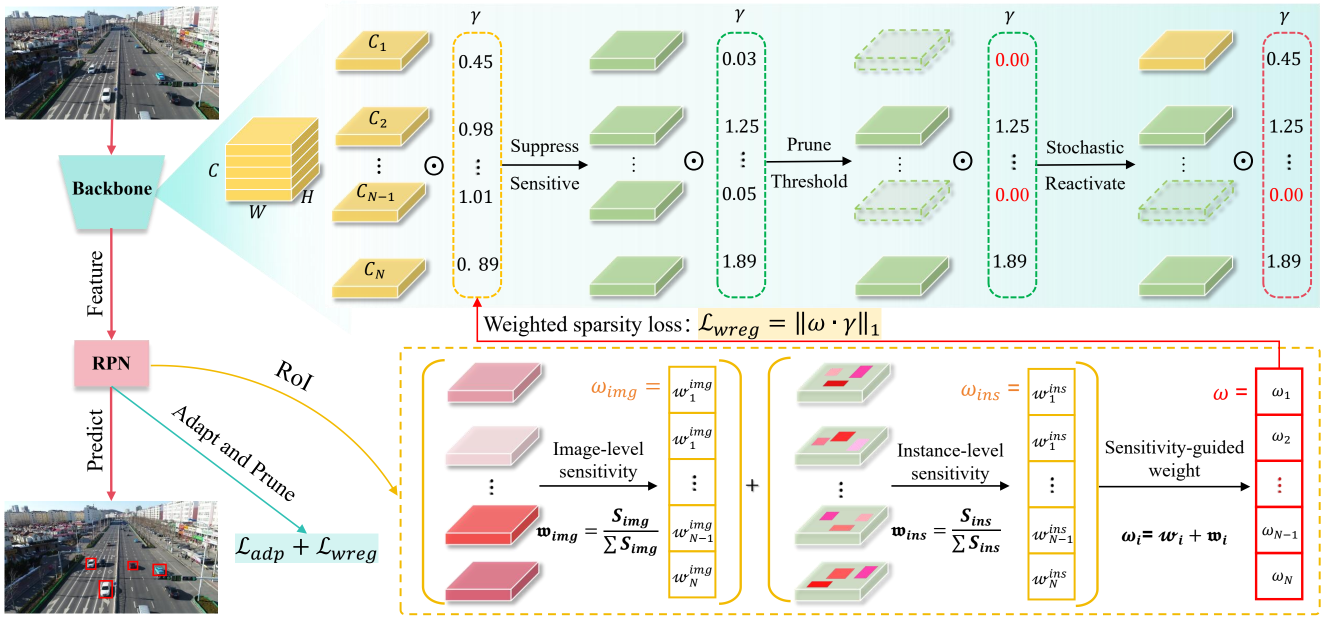

Figure 4. Overview of our method. Motivated by the observation that certain feature channels learned from the source domain negatively impact cross-domain performance, we propose suppressing and pruning sensitive channels while adapting invariant ones. We introduce sensitivity-guided channel pruning, which quantifies the importance of each feature channel based on its image-level and instance-level sensitivity to domain discrepancies. We then apply a weighted sparsity loss to the learnable parameters, promoting the pruning of channels with higher sensitivity based on the predefined threshold and ratio. To prevent the loss of potentially useful channels from early pruning, we introduce a stochastic channel reactivation mechanism.

images. TTA-OD primarily addresses adaptation to the static target domain in an offline setting, assuming that test data follows an i.i.d. distribution with full access to the entire test set. However, in practice, test data is often streamed from continuously changing environments. Thus, we focus on the online continual setting, which is more aligned with real-world scenarios.

Continual Test-time Adaptive Object Detection (CTTA-OD) [2, 4, 30, 31, 33, 35, 42] has gained increasing attention due to its practicality. DUA [30] continuously adapts batch normalization statistics to refine feature representations. STFAR [4] follows a self-training paradigm, generating pseudo-labeled objects and incorporating feature alignment regularization to improve the robustness of self-training. ActMAD [31] focuses on fine-grained alignment of activation statistics between test and training data. Most CTTA-OD methods primarily focus on improving effectiveness, while rarely exploring ways to enhance computational efficiency, despite the task's online requirement. In this paper, we offer a new perspective on enhancing computational efficiency from the model's viewpoint. By suppressing and pruning sensitive feature channels while adapting invariant ones, we achieve both effectiveness and efficiency.

Network Pruning [6, 13, 19, 22, 24, 26, 29, 44] aims to remove relatively ineffective subsets of parameters or structures from a neural network, improving efficiency while minimizing the degradation in model performance. Pruning methods are broadly categorized into unstructured and structured types. Unstructured pruning removes individual weights, creating sparsity but often requiring specialized hardware for acceleration. Structured pruning [18], by removing entire filters or layers, offers practical acceleration without such dependencies. Channel pruning, a key structured method, boosts computational efficiency by selectively removing feature channels. Motivated by the success of network pruning in reducing computational costs, we are prompted to explore whether similar "ineffective subsets" exist in source pre-trained model. Our exploratory experiments reveal that certain sensitive feature channels in the source pre-trained network negatively impact target domain performance. This finding underscores the feasibility and potential of incorporating pruning to reduce computational overhead.

3. Methodology

In this section, we begin by introducing the task setup. We then describe our sensitivity-guided channel suppression and pruning. Following this, we present the optimization objectives and the pruning process, analyzing their effects on model computational efficiency. Finally, we introduce the stochastic channel reactivation strategy, which enables the model to reassess the utility of pruned channels. Fig 4 provides an overview of our method.

3.1. Problem Definition

Assume we have an object detector pre-trained on the source domain $D_s = \{x_s, y_s\}$, where x_s denotes the source images and y_s denotes the corresponding bounding boxes and category labels. Our goal is to adapt the detector to a sequence of continuously changing target domains $\{D_t^1, D_t^2, ..., D_t^N\}$ using only the target data while making predictions. The target domain at time period n is denoted as $D_t^n = \{x_t^n\}$, where x_t^n denotes the target images at time period n and $P_{test}^n \neq P_{test}^{n-1}$. Following prior work [4, 31, 42], as the source domain is inaccessible during adaptation, pre-computed source feature statistics, such as the mean and variance, are available.

3.2. Network Pruning

To achieve channel-level pruning, we utilize the learnable scaling factors in batch normalization (BN) layers to effectively identify and prune sensitive channels in the network, following prior work [29]. This approach offers two main advantages: First, the ResNet backbone is composed of convolutional layers, BN layers, and activation functions. The BN layers normalize the features extracted by the convolutional layers and apply learnable scaling and shifting parameters to each channel:

$$\hat{z} = \frac{z_{in} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \quad z_{out} = \gamma \hat{z} + \beta, \tag{1}$$

where z_{in} and z_{out} denote the input convolutional feature and output, respectively. $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ are the mean and standard deviation values over $\mathcal{B}$. The parameters γ and β, which are trainable affine transformation factors, allow for the linear transformation of the normalized activations back to any scale. By adjusting the scaling factors γ in the BN layers, we can directly control the magnitude of each feature channel, enabling effective suppression and pruning of specific channels by reducing their scaling factors.

Furthermore, we apply $\mathcal{L}_1$ regularization as the sparsity loss $\mathcal{L}_{\text{reg}}$, pushing all scaling factors γ_i to approach zero for channel suppression:

$$\mathcal{L}_{\text{reg}} = \sum_{i=1}^{n} \|\gamma_i\|_1, \tag{2}$$

where n denotes the total number of BN layers considered.

3.3. Sensitivity-guided Channel Pruning

Evidently, the above sparsity regularization loss suppresses all channels uniformly. However, based on our earlier analysis, we need to suppress domain-sensitive channels while preserving and adapting domain-invariant ones. To achieve this, we introduce a sensitivity-guided weighted sparsity loss $\mathcal{L}_{\text{wreg}}$, where the ω_i reflects the domain sensitivity of each channel γ_i:

$$\mathcal{L}_{\text{wreg}} = \sum_{i=1}^{n} \|\omega_i \cdot \gamma_i\|_1, \tag{3}$$

where $\cdot$ denotes element-wise multiplication. Specifically, omitting the subscript i for simplicity, ω consists of two components: image-level channel sensitivity ω_{img} and instance-level channel sensitivity ω_{ins}:

$$\omega = \omega_{\text{img}} + \omega_{\text{ins}}. \tag{4}$$

We compute ω_{img} to capture image-level channel variation under domain shift as follows:

$$S_{\text{img}} = \frac{\sum_{n=1}^{N} \left\| F_t^n - \overline{F_s} \right\|_1}{ND} \in \mathbb{R}^C, \tag{5}$$

$$w_{\text{img}} = C \times \frac{S_{\text{img}}}{\sum S_{\text{img}}}, \tag{6}$$

where F_t^n denotes the feature map extracted by the convolutional layer before the specified BN layer for the n-th image in the target domain mini-batch, and $\overline{F_s}$ denotes the pre-computed average feature map at the same layer position from the source domain. Both F_t^n and $\overline{F_s} \in \mathbb{R}^{\text{C}\times\text{D}}$, where N is the batch size, C is the number of channels, and D is the product of the H and W of the feature map. To stabilize training, we further normalize S_{img} to balance the magnitude of variance across all channels, yielding w_{img}.

To capture fine-grained channel variation, we compute the instance-level ω_{ins} by leveraging the Regions of Interest (RoIs) predicted by the object detector on the target domain data. Specifically, we focus on RoIs with a background confidence less than 0.5, indicating potential foreground objects. For each selected RoI in the n-th image of the target domain mini-batch, we extract the corresponding feature map region from the convolutional feature map F_t^n. For all obtained RoI features $\{f_t^m\}_{m=1}^M$, we utilize the RoI-Alignment operation [17] to align the spatial size of all instance-level features. We then calculate ω_{ins} as follows:

$$S_{\text{ins}} = \frac{\sum_{m=1}^{M} \left\| f_t^m - \overline{f_s} \right\|_1}{M D_{\text{RoI}}} \in \mathbb{R}^C, \tag{7}$$

$$w_{\text{ins}} = C \times \frac{S_{\text{ins}}}{\sum S_{\text{ins}}}, \tag{8}$$

where M denote the number of selected RoIs within the target domain mini-batch, and D_{RoI} represent the spatial dimensions of the RoI features after alignment, $\overline{f_s} \in \mathbb{R}^{\text{C}\times\text{D}_{\text{RoI}}}$ refers to the pre-computed average RoI feature at the same layer in the source domain, based on predicted RoIs and their confidence scores. To ensure numerical stability, we also apply channel-wise normalization, yielding w_{ins}.

Algorithm 1: Efficient CTTA-OD via pruning

```
Input: Target data $x_t$, pruning threshold $t$, pruning
       ratio threshold $p$, reactivation probability $r$
for each batch of target data $x_t^n$ do
    Prune channels where the BN scaling factor
      $\gamma < t$ and define the computation graph;
    Forward propagate for prediction;
    Compute pruning ratio $\rho$ according to Eq. (12);
    if $\rho < p$ then
        Compute loss as in Eq. (3) and Eq. (11);
    else
        Compute loss as in Eq. (11);
        Stochastic pruned channels reactivation with
          $r$ according to Eq. (14);
    Backward propagate for updating;
```

3.4. Optimization

For the adaptation loss, we follow prior work [4, 31, 42] and employ feature distribution alignment. Specifically, we use both image-level and instance-level distribution alignment to minimize the KL divergence between source and target feature distributions. Let μ_{s} and Σ_{s} denote the mean and variance of the source features. To capture distribution shifts in the test domain, we estimate the mean of the test features, μ_{t}, using an exponentially moving average (EMA). The image-level alignment loss is defined as the KL divergence between the source and target distributions:

$$\mathcal{L}_{\text{img}} = D_{\text{KL}}\left(\mathcal{N}(\mu_{\text{s}}, \Sigma_{\text{s}}), \mathcal{N}(\mu_{\text{t}}, \Sigma_{\text{s}})\right). \tag{9}$$

For instance-level alignment, we introduce an intra-class feature alignment loss that minimizes the KL divergence between the source and target distributions for each category:

$$\mathcal{L}_{\text{ins}} = \sum_{k} w_k \cdot D_{\text{KL}}\left(\mathcal{N}(\mu_{\text{s}}^k, \Sigma_{\text{s}}^k), \mathcal{N}(\mu^k\text{t}, \Sigma_{\text{s}}^k)\right), \tag{10}$$

where μ_{s}^k and Σ_{s}^k are the mean and variance for category k in the source domain, and μ_{t}^k denotes the EMA-updated mean for category k in the target domain. To further address class imbalance, we dynamically adjust the weight w_k for each category based on its frequency in the target domain, assigning higher weights to rare classes to enhance alignment quality. Overall, the adaptation loss is as follows:

$$\mathcal{L}_{\text{adp}} = \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{ins}}. \tag{11}$$

During adapting the continuously incoming target data, our method dynamically prunes channels for all the considered BN layers before each optimization step, constructing a sub-network for the current step. Specifically, with a predefined pruning threshold t, for each scaling parameter γ_i in the BN layers, if $\gamma_i < t$, the corresponding channel and the associated filter in the preceding convolutional layer are removed, defining the forward computation graph for the current step. We define the pruning ratio ρ as the ratio of pruned channels (those satisfying $\gamma_i < t$) to the total number of channels across considered BN layers:

$$\rho = \frac{\sum_i |\{\gamma_i \mid \gamma_i < t\}|}{\sum_i |\{\gamma_i\}|}. \tag{12}$$

During the backward optimization process, the loss function $\mathcal{L}_{total}$ is defined based on the pruning ratio ρ as follows:

$$\mathcal{L}_{total} = \begin{cases} \mathcal{L}_{\text{adp}} + \lambda \mathcal{L}_{\text{wreg}}, & \text{if } \rho < p \\ \mathcal{L}_{\text{adp}}, & \text{if } \rho \geq p \end{cases} \tag{13}$$

where λ is a balancing coefficient between the adaptation loss and the pruning loss, p is the predefined pruning ratio threshold. By jointly optimizing with the adaptation loss and regulating the pruning ratio through the threshold p, we effectively prevent over-pruning, balancing adaptability and computational efficiency.

Discussion about efficiency. The ResNet backbone comprises stacked modules like BasicBlock and Bottleneck, each with multiple Conv-BN-ReLU groupings. Except for the down-sampling group in the residual branch and the first group in the main branch, other convolution layers exhibit both input and output sparsity. Positioned between two BN layers, these layers support efficient pruning by removing channels within each filter as well as eliminating entire filters. This setup results in a quadratic gain in computational efficiency relative to the pruning ratio; for instance, retaining only half of the channels in both adjacent BN layers reduces the computational load of the intermediate convolution layer to one-fourth of its original cost.

Our method reduces computational costs in both forward and backward propagation. In forward propagation, fewer nodes lead to reduced computational requirements. Backpropagation costs primarily arise from gradient propagation through the computation graph and the calculation of gradients for updatable parameters. By shortening gradient paths and reducing node count, the simplified computation graph lowers the computational burden of gradient propagation. Furthermore, with fewer parameters involved, the cost of computing their gradients is also minimized.

3.5. Stochastic Channel Reactivation

During optimization, some channels may be subject to early pruning or mis-pruning due to initial scaling parameter values or the dynamic nature of the target domain. Once a channel is pruned, it is excluded from subsequent forward and backward passes, eliminating any chance to correct pruned channels. To this end, we propose a stochastic channel reactivation strategy. Specifically, when the pruning ratio ρ exceeds the predefined pruning ratio threshold

Table 1. Comparisons on Cityscapes → Cityscapes-C. We report continual test-time adaptive detection results along with FLOPs consumption. 'Avg' denotes the average results over all conditions across 10 rounds. 'Fwd' indicates FLOPs during forward propagation, 'Bwd' indicates FLOPs during backward propagation, and 'Total' denotes the overall FLOPs.

Round	1				5				10				Avg	FLOPs		
Condition	Motion	Snow	Defocus	Contrast	Motion	Snow	Defocus	Contrast	Motion	Snow	Defocus	Contrast		Fwd	Bwd	Total
Direct Test	5.6	0.6	10.8	1.0	5.6	0.6	10.8	1.0	5.6	0.6	10.8	1.0	4.5	250.7	0.0	250.7
TeST [35]	6.7	1.9	11.8	2.2	6.9	1.1	12.1	2.1	6.2	1.6	11.6	2.4	5.6	1011.1	953.2	1964.2
DUA [30]	6.0	0.8	11.3	1.4	6.1	0.9	11.1	1.2	5.2	0.2	10.5	0.5	4.6	250.7	0.0	250.7
STFAR [4]	7.5	1.9	12.4	2.4	10.2	2.7	12.4	5.0	10.6	2.7	11.7	4.1	7.6	501.4	501.4	1002.9
ActMAD [31]	12.4	1.8	15.1	5.5	14.0	3.3	13.9	7.7	12.1	2.2	13.7	5.0	8.9	250.7	501.4	752.2
WHW [42]	13.3	2.8	16.5	6.7	14.7	3.5	14.7	9.9	13.6	3.1	13.7	6.6	9.1	253.0	255.2	508.2
Ours	13.2	2.6	17.1	7.4	15.6	4.6	15.1	10.3	15.2	4.8	15.1	10.4	11.4	224.6	225.0	449.6

p, we employ Bernoulli sampling to determine whether each pruned channel should be reactivated. For each pruned channel c_i^j, a random variable b_i^j is sampled from a Bernoulli distribution:

$$b_i^j \sim \text{Bernoulli}(r), \tag{14}$$

where r is a small reactivation probability. If $b_i^j = 1$, the channel's BN scaling parameter γ_i^j is reset to its initial pre-trained value from the source domain. This reactivation allows the channel to rejoin the model's forward and backward passes, giving the model an opportunity to reassess the utility of the reactivated channels, mitigating losses caused by early pruning or mis-pruning. For clarity, the workflow of our method is illustrated in Algorithm 1.

4. Experiments

4.1. Experimental Setup

Datasets. We conduct experiments in both autonomous driving and UAV scenarios, including **Cityscapes-C**, **ACDC**, and **UAVDT-C**. The **Cityscapes** dataset [8] consists of 2,975 training images and 500 validation images with 8 categories of objects. We construct **Cityscapes-C** based on benchmark robustness tasks [20], selecting four common corruption types: snow, contrast, motion blur, and defocus blur. These corruptions are applied to the validation set at the maximum severity level 5, with each corruption type forming an individual target domain consisting of 500 images. The **ACDC** dataset [34] shares the same class categories as Cityscapes but includes four different adverse visual conditions: fog, night, rain, and snow, with each condition containing 400 unlabeled images. The **UAVDT** dataset [12] can be separated into three parts based on the weather annotations: daylight, nighttime, and fog. We use 17.5k images from the daylight part as the source domain and extract 500 images each from the nighttime and fog parts as target domains. Additionally, 500 images from the remaining daylight images are selected and applied with motion blur and defocus blur based on robustness benchmark [20]. Thus, the **UAVDT-C** include nighttime, fog, motion blur, and defocus blur. For all experiment settings, we repeat 10 times adaptation to target domain group to evaluate performance.

Implementation Details. We use Faster R-CNN [32] with a ResNet18 [16] backbone pre-trained on ImageNet [9] as the detector. During test-time adaptation, we adapt the learnable scaling factors in the BN layers while freezing all other parameters pre-trained on the source domain. The batch size is set to 4, and the learning rate for the Adam optimizer is set to 0.005. For hyper-parameter settings, the pruning threshold t is set to 0.05, the pruning ratio threshold p to 0.1, the reactivation probability r to 0.01, and the loss balancing coefficient λ to 0.05. Following prior work [41], we simulate the continual changing target domains via 10 rounds adaptation.

Evaluation Metrics. We use mAP@50 (%) and FLOPs (G) as the evaluation metrics for detection performance and computational efficiency. FLOPs during adaptation mainly consist of two components: Forward propagation (Fwd) and Backward propagation (Bwd). Fwd FLOPs can be calculated based on the model structure or using auxiliary tools. In models where all parameters are learnable, Bwd typically requires twice the FLOPs of Fwd [21], mainly consisting of two parts: gradient propagation across layers and gradient computation for the learnable parameters. Given that the Fwd of $f_{i+1} = f_i \mathcal{W} + b$, the Bwd from the $i + 1^{th}$ layer to the i^{th} layer, and the weight gradient are formulated as:

$$\frac{\partial \mathcal{L}}{\partial f_i} = \frac{\partial \mathcal{L}}{\partial f_{i+1}} \mathcal{W}^T, \tag{15}$$

$$\frac{\partial \mathcal{L}}{\partial \mathcal{W}} = f_i^T \frac{\partial \mathcal{L}}{\partial f_{i+1}}, \tag{16}$$

where Eq. (15) describes gradient propagation across layers, which must be computed regardless of whether the parameters in the current layer are learnable, and Eq. (16) describes the gradient computation for learnable parameters, which is only necessary if the parameters of the current

Table 2. Comparisons on UAVDT → UAVDT-C. We report continual test-time adaptive detection results along with FLOPs consumption.

Round	1				5				10				Avg	FLOPs		
Condition	Fog	Defocus	Motion	Night	Fog	Defocus	Motion	Night	Fog	Defocus	Motion	Night		Fwd	Bwd	Total
Direct Test	4.8	2.5	6.8	11.4	4.8	2.5	6.8	11.4	4.8	2.5	6.8	11.4	6.4	250.7	0.0	250.7
TeST [35]	6.4	4.5	10.5	12.0	8.2	6.9	10.2	12.1	8.8	6.6	10.5	12.2	9.2	1011.1	953.2	1964.2
DUA [30]	5.2	2.8	7.1	11.7	5.0	2.8	6.9	11.8	3.6	1.1	5.5	10.3	6.1	250.7	0.0	250.7
STFAR [4]	6.9	5.7	11.3	12.3	9.5	6.9	10.5	12.0	9.0	6.6	10.8	12.2	9.6	501.4	501.4	1002.9
ActMAD [31]	9.5	6.4	15.3	15.0	13.2	9.1	13.4	14.0	13.1	7.5	12.3	13.9	10.8	250.7	501.4	752.2
WHW [42]	9.9	7.8	17.0	15.6	16.1	10.1	14.6	14.9	13.9	7.1	13.6	13.4	12.0	253.0	255.2	508.2
Ours	10.2	8.9	17.2	16.9	16.6	11.1	16.0	15.4	18.0	10.1	16.3	15.5	14.3	221.0	221.4	442.4

Table 3. Comparisons on Cityscapes → ACDC. We report continual test-time adaptive detection results along with FLOPs consumption.

Round	1				5				10				Avg	FLOPs		
Condition	Snow	Rain	Night	Fog	Snow	Rain	Night	Fog	Snow	Rain	Night	Fog		Fwd	Bwd	Total
Direct Test	21.8	22.2	9.8	34.4	21.8	22.2	9.8	34.4	21.8	22.2	9.8	34.4	22.0	250.7	0	250.7
TeST [35]	22.0	22.1	9.6	34.4	21.8	22.3	9.6	34.1	22.0	22.2	9.5	34.4	22.1	1011.1	953.2	1964.2
DUA [30]	20.9	21.5	9.1	33.8	21.1	21.3	9.0	33.7	20.4	20.8	8.4	33.1	21.1	250.7	0	250.7
STFAR [4]	21.9	22.5	9.5	34.7	21.7	22.2	9.5	34.2	21.6	22.8	9.6	34.7	22.1	501.4	501.4	1002.9
ActMAD [31]	22.8	22.6	10.7	33.5	23.4	22.7	11.5	34.5	22.2	22.9	10.9	34.9	23.3	250.7	501.4	752.2
WHW [42]	23.2	22.6	11.2	34.7	24.0	24.1	12.4	34.2	24.0	23.6	12.7	34.7	23.6	253.0	255.2	508.2
Ours	23.4	22.6	11.7	34.7	24.7	24.1	13.0	35.1	24.5	24.1	12.9	35.0	24.2	226.2	226.6	452.8

Table 4. Ablation analysis of the framework components, where 'SCR' represents Stochastic Channel Reactivation.

$\mathcal{L}_{\text{adp}}$	$\mathcal{L}_{\text{reg}}$	ω_{img}	ω_{ins}	SCR	Avg	ΔFLOPs↓
					4.5	0.0%
✓					10.9	0.0%
✓	✓				8.8	10.2%
✓	✓	✓			10.5	10.5%
✓	✓		✓		9.9	10.9%
✓	✓	✓	✓		11.2	10.7%
✓	✓	✓	✓	✓	11.4	10.4%

layer are learnable. Therefore, the FLOPs for Fwd can be calculated based on the model's computation graph, and the FLOPs for Bwd can be calculated based on the FLOPs for Fwd and the distribution of learnable parameters, enabling calculation of the total FLOPs. FLOPs computation for convolutional layers is similar, so we skip this discussion.

4.2. Benchmark Results

We compare our approach with the baseline (directly testing the source model on target domains) and five competing CTTA-OD methods [4, 30, 31, 35, 42] across three benchmarks. For open-source methods, we strictly follow the provided hyper-parameters, and for others, we replicate based on paper details. The results are shown in Tables 1, 2, 3. Our method achieves an increase of 2.3, 2.3, and 0.6 in ten-round average mAP over the recent SOTA method, while reducing FLOPs by 11.6%, 12.9%, and 10.9%, respectively. Compared to the baseline, our method boosts average mAP by 6.9, 7.9, and 2.2 with the lowest additional FLOPs among all CTTA-OD methods. In DUA, multi-round adaptation leads to severe disruption of BN layer statistics by domain shifts, which does not contribute positively to target domain performance. TeST and STFAR, based on the mean teacher framework, achieve stable improvements during adaptation but incur substantial additional computation due to the inference and updates required for a larger set of parameters. Compared to alignment-based methods such as ActMAD and WHW, our approach prunes sensitive channels, reducing adaptation difficulty. Additionally, the smaller number of optimized parameters allows for faster adaptation to changing target domains and minimizes error accumulation, resulting in the highest performance gains.

4.3. Ablation Analysis

Framework Components. We conduct an ablation study to verify the effectiveness of each component in our proposed framework, as detailed in Table 4. Indiscriminate pruning leads to notable performance degradation, whereas channel sensitivity guidance at both image and instance levels enables the model to make informed pruning decisions, thereby preserving performance. Stochastic channel reactivation provides additional performance gain, albeit with a minor increase in the FLOPs consumption.

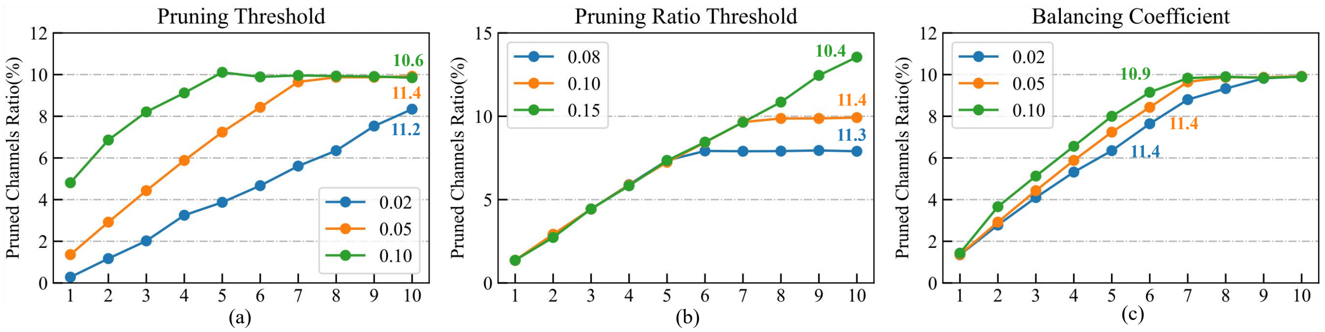

Figure 5. Ablation analysis of three hyper-parameters: (a) Pruning threshold t, (b) Pruning ratio threshold p, and (c) Balancing coefficient λ. We provide the pruned channel ratio for each round, with the average mAP for each setting annotated on the line plot.

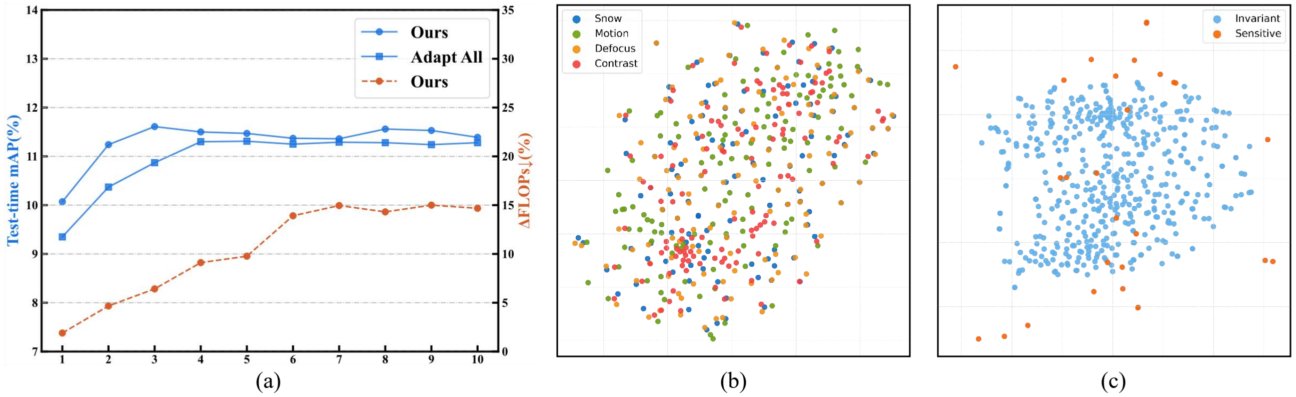

Figure 6. Visualization analysis of the effectiveness of pruning sensitive channels. (a) mAP change curves during test-time adaptation for our method versus the adapt-all-channels approach, with FLOPs reduction for our method. (b) T-SNE visualization of invariant channel distribution changes retained by our method in the final adaptation round across four target domains. (c) T-SNE visualization of feature channel distribution when adapting all channels in the final round, with sensitive and invariant channels color-coded based on our mask.

Hyper-parameters. We investigate how varying the setting of pruning threshold t, pruning ratio threshold p, and balancing coefficient λ. affects the pruning process and adaptation performance, as shown in Fig 5. Setting a high pruning threshold risks early pruning, removing useful channels and causing performance loss, while a low threshold stabilizes performance but slows pruning process, limiting FLOPs reduction. A high pruning ratio threshold may lead to over-pruning, causing substantial performance degradation. A larger balancing coefficient between losses prioritizes pruning loss optimization, accelerating the pruning process; however, an aggressive pruning strategy may also compromise performance.

4.4. Visualization Analysis

To validate the effectiveness of pruning sensitive channels, we conduct visualization analysis. Fig 6 (a) compares mAP change curves during test-time adaptation between adapting all channels and our method, which focuses only on invariant channels. Pruning sensitive channels reduces adaptation difficulty, enabling faster learning of target domain distributions in early rounds and boosting average mAP. It also cuts computational overhead, while adapting all channels does not decrease FLOPs. Fig 6 (b) visualizes the distribution changes of the invariant channels retained by our method through T-SNE in the final round of continual adaptation across four target domains. The mixed distribution across these domains demonstrates the invariance of the feature channels. Fig 6 (c) presents the feature channel distribution when adapting all channels during the final round, with sensitive and invariant channels differentiated by color using our channel mask. Sensitive channels show scattered distributions under changing target domains, while invariant channels remain more stable, validating the necessity of pruning sensitive channels.

5. Conclusion and Future Work

In this paper, we introduce an efficient CTTA-OD framework based on sensitivity-guided pruning, offering a new perspective on enhancing computational efficiency from a model structure standpoint. For future work, one promising direction is to develop a more subtle pruning strategy, as focusing solely on general invariant features may not fully exploit the model's potential. Furthermore, extending our approach to versatile detectors such as YOLO [11], DETR [3], or models without BN layers will also be crucial for broader application.

References

[1] Muhammad Abrar, Ushna Ajmal, Ziyad M Almohaimeed, Xiang Gui, Rizwan Akram, and Roha Masroor. Energy efficient uav-enabled mobile edge computing for iot devices: A review. *IEEE Access*, 9:127779–127798, 2021. 1

[2] Shilei Cao, Yan Liu, Juepeng Zheng, Weijia Li, Runmin Dong, and Haohuan Fu. Exploring test-time adaptation for object detection in continually changing environments. *arXiv preprint arXiv:2406.16439*, 2024. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 8

[4] Yijin Chen, Xun Xu, Yongyi Su, and Kui Jia. Stfar: Improving object detection robustness at test-time by self-training with feature alignment regularization. *arXiv preprint arXiv:2303.17937*, 2023. 1, 3, 4, 5, 6, 7

[5] Zhihong Chen, Zilei Wang, and Yixin Zhang. Exploiting low-confidence pseudo-labels for source-free object detection. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 5370–5379, 2023. 2

[6] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 3

[7] Qiaosong Chu, Shuyan Li, Guangyi Chen, Kai Li, and Xiu Li. Adversarial alignment for source free object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 452–460, 2023. 2

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 6

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[10] Jinhong Deng, Wen Li, and Lixin Duan. Balanced teacher for source-free object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024. 2

[11] Tausif Diwan, G Anirudh, and Jitendra V Tembhurne. Object detection using yolo: Challenges, architectural successors, datasets and applications. *multimedia Tools and Applications*, 82(6):9243–9275, 2023. 8

[12] Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 370–386, 2018. 6

[13] Xitong Gao, Yiren Zhao, Łukasz Dudziak, Robert Mullins, and Cheng-zhong Xu. Dynamic channel pruning: Feature boosting and suppression. *arXiv preprint arXiv:1810.05331*, 2018. 3

[14] Chengjie Ge, Xueyang Fu, and Zheng-Jun Zha. Learning dual convolutional dictionaries for image de-raining. In *ACM MM*, pages 6636–6644, 2022. 1

[15] Chengjie Ge, Xueyang Fu, Peng He, Kunyu Wang, Chengzhi Cao, and Zheng-Jun Zha. Neuromorphic event signal-driven network for video de-raining. In *AAAI*, pages 1878–1886, 2024. 1

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 4

[18] Yang He and Lingao Xiao. Structured pruning for deep convolutional neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2023. 3

[19] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018. 3

[20] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 6

[21] Marius Hobbhahn and Jaime Sevilla. What's the backward-forward flop ratio for neural networks?, 2021. Accessed: 2024-10-26. 6

[22] Weizhe Hua, Yuan Zhou, Christopher M De Sa, Zhiru Zhang, and G Edward Suh. Channel gating neural networks. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[23] Trinh Le Ba Khanh, Huy-Hung Nguyen, Long Hoang Pham, Duong Nguyen-Ngoc Tran, and Jae Wook Jeon. Dynamic retraining-updating mean teacher for source-free object detection. *arXiv preprint arXiv:2407.16497*, 2024. 2

[24] Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Dynamic slimmable network. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 8607–8617, 2021. 3

[25] Xianfeng Li, Weijie Chen, Di Xie, Shicai Yang, Peng Yuan, Shiliang Pu, and Yueting Zhuang. A free lunch for unsupervised domain adaptive object detection without source data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8474–8481, 2021. 2

[26] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. *Advances in neural information processing systems*, 30, 2017. 3

[27] Qipeng Liu, Luojun Lin, Zhifeng Shen, and Zhifeng Yang. Periodically exchange teacher-student for source-free object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6414–6424, 2023. 2

[28] Shaoshan Liu, Liangkai Liu, Jie Tang, Bo Yu, Yifan Wang, and Weisong Shi. Edge computing for autonomous driving: Opportunities and challenges. *Proceedings of the IEEE*, 107 (8):1697–1716, 2019. 1

[29] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient

convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017. 3, 4

[30] M Jehanzeb Mirza, Jakub Micorek, Horst Possegger, and Horst Bischof. The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14765–14775, 2022. 3, 6, 7

[31] Muhammad Jehanzeb Mirza, Pol Jané Soneira, Wei Lin, Mateusz Kozinski, Horst Possegger, and Horst Bischof. Actmad: Activation matching to align distributions for test-time-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24152–24161, 2023. 1, 3, 4, 5, 6, 7

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 6

[33] Xiaoqian Ruan and Wei Tang. Fully test-time adaptation for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1038–1047, 2024. 3

[34] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Acdc: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10765–10775, 2021. 6

[35] Samarth Sinha, Peter Gehler, Francesco Locatello, and Bernt Schiele. Test: Test-time self-training under distribution shift. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2759–2769, 2023. 1, 3, 6, 7

[36] Khaled Telli, Okba Kraa, Yassine Himeur, Abdelmalik Ouamane, Mohamed Boumehraz, Shadi Atalla, and Wathiq Mansoor. A comprehensive review of recent research trends on unmanned aerial vehicles (uavs). *Systems*, 11(8):400, 2023. 1

[37] Ayşegül Uçar, Yakup Demir, and Cüneyt Güzeliş. Object recognition and detection with deep learning for autonomous driving applications. *Simulation*, 93(9):759–769, 2017. 1

[38] Vibashan VS, Poojan Oza, and Vishal M Patel. Instance relation graph guided source-free domain adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3520–3530, 2023. 2

[39] Kunyu Wang, Xueyang Fu, Yukun Huang, Chengzhi Cao, Gege Shi, and Zheng-Jun Zha. Generalized uav object detection via frequency domain disentanglement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1064–1073, 2023. 1

[40] Kunyu Wang, Xueyang Fu, Chengjie Ge, Chengzhi Cao, and Zheng-Jun Zha. Towards generalized uav object detection: A novel perspective from frequency domain disentanglement. *International Journal of Computer Vision*, pages 1–29, 2024. 1

[41] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7201–7211, 2022. 1, 6

[42] Jayeon Yoo, Dongkwan Lee, Inseop Chung, Donghyun Kim, and Nojun Kwak. What how and when should object detectors update in continually changing test domains? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23354–23363, 2024. 3, 4, 5, 6, 7

[43] Ilhoon Yoon, Hyeongjun Kwon, Jin Kim, Junyoung Park, Hyunsung Jang, and Kwanghoon Sohn. Enhancing source-free domain adaptive object detection with low-confidence pseudo label distillation. *arXiv preprint arXiv:2407.13524*, 2024. 2

[44] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in neural information processing systems*, 32, 2019. 3

[45] Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*, 2024. 1

[46] Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024. 1

[47] Sicheng Zhao, Huizai Yao, Chuang Lin, Yue Gao, and Guiguang Ding. Multi-source-free domain adaptive object detection. *International Journal of Computer Vision*, pages 1–33, 2024. 2