

What, How, and When Should Object Detectors Update in Continually Changing Test Domains?

Jayeon Yoo¹ Dongkwan Lee¹ Inseop Chung¹ Donghyun Kim^{2*} Nojun Kwak^{1*}

¹Seoul National University ²Korea University

¹{jayeon.yoo, biancco, jis3613, nojunk}@snu.ac.kr, ²d.kim@korea.ac.kr

Abstract

It is a well-known fact that the performance of deep learning models deteriorates when they encounter a distribution shift at test time. Test-time adaptation (TTA) algorithms have been proposed to adapt the model online while inferring test data. However, existing research predominantly focuses on classification tasks through the optimization of batch normalization layers or classification heads, but this approach limits its applicability to various model architectures like Transformers and makes it challenging to apply to other tasks, such as object detection. In this paper, we propose a novel online adaptation approach for object detection in continually changing domains, considering which part of the model to update, how to update it, and when to perform the update. By introducing architecture-agnostic and lightweight adaptor modules and only updating these while leaving the pre-trained backbone unchanged, we can rapidly adapt to new test domains in an efficient way and prevent catastrophic forgetting. Furthermore, we present a practical and straightforward class-wise feature aligning method for object detection to resolve domain shifts. Additionally, we enhance efficiency by determining when the model is sufficiently adapted or when additional adaptation is needed due to changes in the test distribution. Our approach surpasses baselines on widely used benchmarks, achieving improvements of up to 4.9%p and 7.9%p in mAP for COCO \rightarrow COCO-corrupted and SHIFT, respectively, while maintaining about 20 FPS or higher. The implementation code is available at https://github.com/natureyoo/ContinualTTA_ObjectDetection.

1. Introduction

Deep learning models excel in many vision tasks but struggle with domain shifts when test data distributions diverge from training data [3, 27, 42]. These shifts, common in real-world settings, arise from natural variations, weather

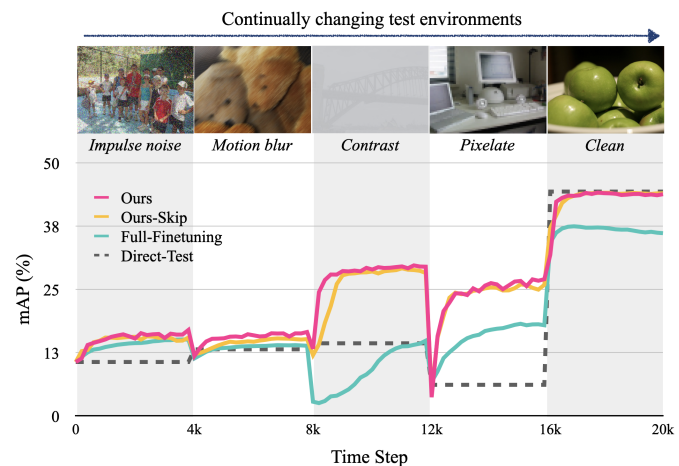


Figure 1. We propose an online adaptation method for object detection in continually changing test domains. Object detectors trained with clean images suffer from performance degradation due to various corruption, such as camera sensor degradation or environmental changes (*Direct-Test*). Updating full parameters for online adaptation require a large number of test samples and vulnerable to drastic domain changes (*Full-Finetuning*), while using only our lightweight adaptor is robust and quickly adapts within a few time steps (*Ours*). We can further improve efficiency by skipping unnecessary adaptation steps (*Ours-Skip*).

changes (e.g., fog, rain), camera differences (e.g., pixelate, defocus blur), and various other factors. Test-Time Adaptation (TTA) [3, 27, 32, 42, 45, 49] addresses these shifts by adapting models to the target (test) distribution online. Considering the dynamic nature of real-world applications, it becomes crucial to extend this approach to Continual Test-Time Adaptation (CTA), which accounts for the ongoing evolution of test distributions. This is particularly vital in environments like autonomous driving, where conditions can change continuously (e.g., from clear to rainy, or day to night). Despite its significance, continual test-time adaptation for object detection remains underexplored.

Recently, several TTA methods [6, 31, 38] tailored for object detection have been proposed. ActMAD [31] aligns all the output feature maps ($\mathbb{R}^{C \times H \times W}$) after Batch Nor-

*Co-corresponding authors

malization (BN) layers [15] to adapt the test domain to be similar to that of the training domain. However, this approach requires significant memory during adaptation and does not explicitly consider the objects present in the image. TeST [38] and STFAR [6] adapt to a test domain by utilizing weak and strong augmented test samples with a teacher-student network [39], but they significantly increase inference costs since they require additional forward passes and update steps. Also, these methods update all network parameters, making them highly inefficient in online adaptation and vulnerable to losing task-specific knowledge when the test domain experiences continual or drastic changes.

In this paper, we aim to develop an efficient continual test-time adaptation (CTA) method for object detection. We investigate the following three key aspects to improve efficiency; *what to update*: while previous TTA methods for object detection [6, 31, 38] use full fine-tuning, updating all parameters at test time, they are inefficient and prone to losing task-specific knowledge in relatively complex object detection tasks. Updating BN layers, as done in many TTA methods for classification [18, 32, 45, 49], is not as effective for object detection, given its smaller batch size compared to classification and the limitation in applying various backbones, such as Transformer [28, 43]. *how to update*: several previous TTA methods for object detection [6, 38] adapt the model by using teacher-student networks, resulting in a significant decrease in inference speed, which is detrimental during test time. While another existing method [31] aligns feature distributions for adaptation, it does not consider each object individually, focusing only on image features, making it less effective for object detection. *when to update*: most TTA or CTA methods update models using all incoming test samples. However, it is inefficient to update continuously the model if it is already sufficiently adapted when the change of the test domain is not significant.

To this end, (1) we propose an efficient continual test-time adaptation method for object detectors to adapt to continually changing test domains through the use of lightweight adaptors which require only 0.54%~0.89% additional parameters compared to the full model. It exhibits efficiency in parameters, memory usage, and adaptation time, along with robustness to continuous domain shifts without catastrophic forgetting. Additionally, it demonstrates wide applicability to various backbone types compared to BN-based TTA methods [18, 24, 32, 45, 49, 50]. (2) To enhance the adaptation effectiveness in the object detection task, we align the feature distribution of the test domain with that of the training domain at both the image-level and object-level using only the mean and variance of features. For estimating the mean of the test domain features, we employ Exponentially Moving Average (EMA) as we can leverage only the current incoming test samples, not the entire test domain data. Due to the unavailability of training

data access, we utilize only the mean and variance of the features from a few training samples. (3) We also introduce two novel criteria that do not require additional resources to determine when the model needs adaptation to enhance efficiency in a continually changing test domain environment. As illustrated in Fig. 1, our approach *Ours*, employing adaptors, tends to adapt much faster to domain changes compared to full parameter updates. This enables efficient TTA by using only a few test samples to update the adaptor and skipping the rest of the updates as shown in *Ours-Skip*.

Our main contributions are summarized as follows:

- We introduce an architecture-agnostic lightweight adaptor, constituting only a maximum of 0.89% of the total model parameters, into the backbone of the object detector to adapt the model in a continually changing test domain. This approach ensures efficiency in parameters, memory usage, and adaptation speed, demonstrating the robust preservation of task-specific knowledge owing to its inherent structural characteristics.
- We propose a straightforward and effective adaptation loss for CTA in object detection tasks. This is achieved by aligning the distribution of training and test domain features at both the image and object levels, utilizing only the mean and variance of a few training samples and EMA-updated mean features of the test domain.
- We also propose two criteria to determine when the model requires adaptation, enabling dynamic skipping or resuming adaptation as needed. This enhancement significantly boosts inference speed by up to about 2 times while maintaining adaptation performance.
- Our adaptation method proves effective for diverse types of domain shifts, including weather changes and sensor variations, regardless of whether the domain shift is drastic or continuous. In particular, our approach consistently improves the mAP by up to 7.9% in COCO→COCO-C and SHIFT-Discrete/Continuous with higher than 20 FPS.

2. Related Work

Test-time adaptation. Recently, there has been a surge of interest in research that adapts models online using unlabeled test samples while simultaneously inferring the test sample to address the domain shift problem, where the test data distribution differs from that of the training data. There are two lines for online adaptation to the test domain, *Test-time Training (TTT)* and *Test-time Adaptation (TTA)*. TTT [1, 2, 27, 42] involves modifying the model architecture during training to train it with self-supervised loss, allowing adaptation to the test domain in the test time by applying this self-supervised loss to the unlabeled test samples. On the other hand, TTA aims to adapt the trained model directly to the test domain without specifically tailored model architectures or losses during training time. NORM [37] and DUA [30] address the domain shifts by

adjusting the statistics of batch normalization (BN) layers using the current test samples, without updating other parameters, inspired by [23]. Following this, [24, 32, 45, 50] and [18] update the affine parameters of BN layers using unsupervised loss, entropy minimization loss to enhance the confidence of test data predictions, and feature distribution alignments loss, respectively. Several studies [16, 17] update the classifier head using the pseudo-prototypes from the test domain. However, these methods limit their applicability to architectures without BN layers or to object detection tasks that involve multiple objects in a single image. Others [31, 40, 49] update full parameters for online adaptation to the test domain in an online manner, but this approach is inefficient and susceptible to the noisy signal from the unsupervised loss. While existing TTA methods are oriented towards classification tasks, we aim to propose an effective and efficient method for online adaptation in the object detection task.

Continual test-time adaptation. Recent studies [33, 46] point out that existing TTA methods have primarily focused on adapting to test domains following an i.i.d assumption and may not perform well when the test data distribution deviates from this assumption. [46] introduces a *Continual TTA (CTA)* method designed for scenarios where the test domain continuously changes over time. This poses challenges in preventing the model from over-adapting to a particular domain shift and preserving the knowledge of the pre-trained model to avoid catastrophic forgetting. In the field of CTA, the self-training strategy adopting an Exponentially Moving Average (EMA) teacher-student structure is attracting interest as an effective algorithm enabling robust representation to be learned through self-knowledge distillation. In many studies, the EMA teacher-student structure and catastrophic restoration of source model weights have been proposed as a solution to achieve the goal of CTA [4, 46, 47]. Approaches using source replay [34], and anti-forgetting regularization [32] have also achieved good performances in robust continuous adaptation. Furthermore, there is growing attention on methods that mitigate the computational and memory challenges associated with CTA, such as [13], which relies on updates to batch normalization statistics.

Test-time adaptive object detection. Research on *TTA for Object Detection (TTAOD)* is progressively emerging [6, 31, 38, 44]. Most existing TTAOD methods [6, 38, 44] exploit a teacher-student network to adapt to the test domain, following the self-training approach commonly employed in Unsupervised Domain Adaptation for object detection [7, 19, 21, 36]. However, it is inefficient for TTA due to data augmentation requirements and additional forward and backward steps, resulting in slower inference speeds and higher memory usage. Another approach, ActMAD [31], aligns the distributions of output feature maps

after all BN layers along the height, width, and channel axes to adapt to the test domain. However, this location-aware feature alignment is limited to datasets with fixed location priors, such as driving datasets, and is less effective for natural images like COCO. Additionally, *CTA for Object Detection (CTAOD)* have not been thoroughly explored. Therefore, there is a need for an effective CTAOD method considering memory and time efficiency.

3. Method

To enable the efficient and effective Continual Test-time Adaptation of Object Detectors (CTAOD), we introduce an approach that specifies *which part of the model should be updated*, describes *how to update* those using unlabeled test data, and determines *whether we perform model updates or not* to improve efficiency.

3.1. Preliminary

Assume that we have an object detector $h \circ g_{\Theta}$, here h and g are the RoI head and the backbone, respectively with their parameters being Θ . The training dataset is denoted as $D_{train} = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \sim P_{train}(x)$ and $y_i = (bbox_i, c_i)$, containing information on the bounding box (bbox) and class label $c_i \in \mathcal{C}$. Consider deploying the detector to the test environments where the test data at period T is denoted as $x_j^T \sim P_{test}^T(x)$, $P_{test}^T \neq P_{train}$ and P_{test}^T deviates from the i.i.d. assumption. In addition, the domain of P_{test}^T continually changes according to T (i.e., $P_{test}^T \neq P_{test}^{T-1}$). Our goal is to adapt the detector $h \circ g$ to P_{test}^T using only test data x_j^T while making predictions.

3.2. What to update: Adaptation via an adaptor

Previous methods [6, 31, 38, 44] adapt the model to the test domain by updating all parameters Θ , leading to inefficiency at test time and a high risk of losing task knowledge from the training data. In contrast, we adapt the model by introducing an adaptor with an extremely small set of parameters and updating only this module while freezing Θ . We introduce a shallow adaptor in parallel for each block, inspired by [5, 14], where transformer-based models are fine-tuned for downstream tasks through parameter-efficient adaptors, as shown in Fig. 2. Each adaptor consists of down-projection layers $\mathcal{W}_{down} \in \mathbb{R}^{d \times \frac{d}{r}}$, up-projection layers $\mathcal{W}_{up} \in \mathbb{R}^{\frac{d}{r} \times d}$ and ReLUs, where d denotes the input channel dimension and r is the channel reduction ratio set to 32 for all adaptors. We use MLP layers for the Transformer block (Fig. 2a) and 1×1 convolutional layers for the ResNet block (Fig. 2b) to introduce architecture-agnostic adaptors. The up-projection layer is initialized to 0 values so that the adaptor does not modify the output of the block, but as the adaptor is gradually updated, it adjusts the output of the block to adapt to the test domain. Even as the adaptor

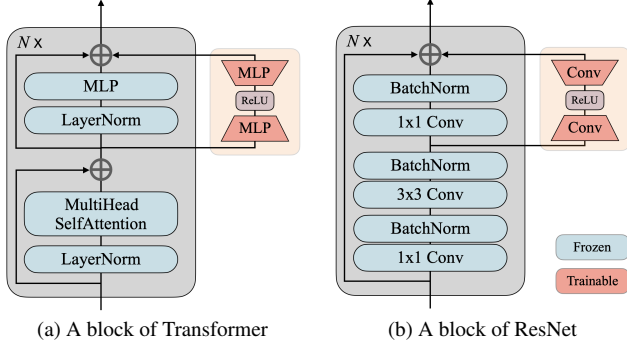


Figure 2. We attach an **adaptor**, which is a shallow and low-rank MLP or CNN, to every N block in parallel. We update only these adaptors while other parameters are frozen. Our approach can be applied to diverse architectures including CNNs and Transformers.

is updated in the test domain, the original backbone parameter Θ remains frozen and fully preserved. This structural preservation, as evident in *Ours* in Fig. 1, enables robust and efficient adaptation to domain changes by maintaining relatively complex task knowledge in object detection and updating very few parameters.

3.3. How to update: EMA feature alignment

To adapt the object detector to the test domain, we align the feature distribution of the test domain with that of the training data, inspired by [18, 31, 40]. In contrast to these methods that solely align image feature distribution, we additionally align object-level features in a class-wise manner, considering class frequency, to enhance its effectiveness for object detection. As the training data is not accessible during test time, we pre-compute the first and second-order statistics, denoted as $\mu_{tr} = \mathbb{E}[F_{tr}]$ and $\Sigma_{tr} = \mathbb{V}\text{ar}[F_{tr}]$, where the operators \mathbb{E} and $\mathbb{V}\text{ar}$ represent the mean and variance respectively. The features $F_{tr} = \{g_{\Theta}(x_{tr})\}$ are computed using only 2,000 training samples, a small subset of the training data. Since a sufficient amount of test domain data is not available at once, and only the current incoming test data, whose features are denoted as F_{te}^t , is accessible at time step t , we estimate the mean of test data features using an exponentially moving average (EMA) as follows:

$$\mu_{te}^t = (1 - \alpha) \cdot \mu_{te}^{t-1} + \alpha \cdot \mathbb{E}[F_{te}^t], \quad \text{s.t. } \mu_{te}^0 = \mu_{tr}. \quad (1)$$

Considering the typically small batch size in object detection compared to classification, we approximate the variance of the test features as $\Sigma_{te} \simeq \Sigma_{tr}$ to reduce instability.

Image-level feature alignment. We estimate the training and test feature distributions as normal distributions and minimize the KL divergence between them as follows:

$$L_{img} = D_{KL}(\mathcal{N}(\mu_{tr}, \Sigma_{tr}), \mathcal{N}(\mu_{te}^t, \Sigma_{tr})). \quad (2)$$

Region-level class-wise feature alignment. In object detection, we deal with multiple objects within a single image, making it challenging to apply the class-wise feature alignment proposed in [40], a TTA method for classification. To handle region-level features that correspond to an object, we use ground truth bounding boxes for the training data and utilize the class predictions of RoI pooled features, f_{te}^t , for unlabeled test data. In object detection, domain shifts often result in lower recall rates, as a significant number of proposals are predicted as background [22]. To mitigate this issue, we filter out features with background scores exceeding a specific threshold. Subsequently, we assign them to the foreground class with the highest probability, as follows:

$$F_{te}^{k,t} = \{f_{te}^t | \arg\max_c p_{fg} = k, p_{bg} < 0.5\}, \quad (3)$$

where $h_{cls}(f_{te}^t) = [p_{fg}, p_{bg}] = [p_0, \dots, p_{C-1}, p_{bg}]$.

We estimate the class-wise feature distribution of the test domain by exploiting $F_{te}^{k,t}$ and Eq. 1. Furthermore, we introduce a weighting scheme for aligning features of less frequently appearing classes, taking into account the severe class imbalance where specific instance (e.g., *person*) may appear multiple times within a single image, as follows:

$$N^{k,t} = N^{k,t-1} + \|F_{te}^{k,t}\|, \quad \text{s.t. } N^{k,0} = 0$$

$$w^{k,t} = \log\left(\frac{\max_i N^{i,t}}{N^{k,t}}\right) + 0.01 \quad (4)$$

$$L_{obj} = \sum_k w^{k,t} \cdot D_{KL}(\mathcal{N}(\mu_{tr}^k, \Sigma_{tr}^k), \mathcal{N}(\mu_{te}^{k,t}, \Sigma_{tr}^k)).$$

Here, the class-wise mean μ^k and variance Σ^k of the training and test data are obtained in the same way as the image-level features. We can effectively adapt the object detector by updating the model to align the feature distribution at both the image and object levels as $L = L_{img} + L_{obj}$.

3.4. When to update: Adaptation on demand

As shown in Fig. 1, *Ours*, which only updates the adaptor proposed in Sec. 3.2, efficiently adapts to changes in the test domain, even with a small subset of early test samples. We leverage its rapid adaptation characteristics to reduce computational costs by skipping model updates (i.e., skipping backward passes) when the model has already sufficiently adapted to the current test domain and resuming model updates when confronted with a new test domain. Therefore, we introduce two criteria to determine when to update the model or not as follows:

(Criterion 1) When the distribution gap exceeds the in-domain distribution gap. Recall that L_{img} (Eq. 2) measures the distribution gap between the test and train distributions. We assume a model is well-adapted to the current test domain when L_{img} is closer to the in-domain distribution gap. We measure the in-domain distribution gap by

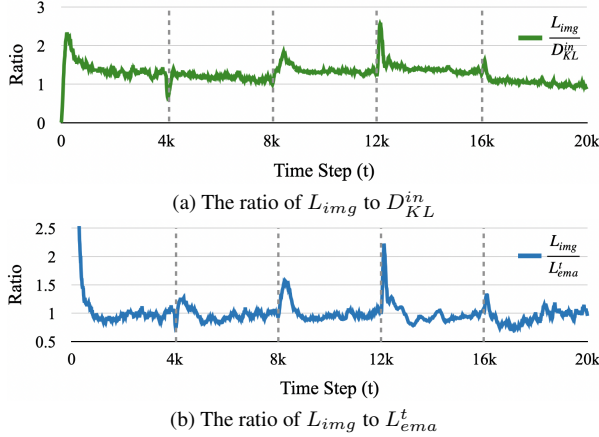


Figure 3. The test domain undergoes a shift every 4,000 time steps, and each metric reaches its peak at the same intervals.

sampling two disjoint subsets, x_i and x_j , of training features F_{tr} from Sec. 3.3 as follows:

$$D_{KL}^{in} = D_{KL}(\mathcal{N}(\mu_{tr}^i, \Sigma_{tr}^i), \mathcal{N}(\mu_{tr}^j, \Sigma_{tr}^j)), \quad (5)$$

where $\mu_{tr}^i, \Sigma_{tr}^i$ are obtained from $x_i \sim P_{train}(x)$ and $\mu_{tr}^j, \Sigma_{tr}^j$ from $x_j \sim P_{train}(x)$. In other words, if L_{img} is noticeably higher than the in-domain distribution gap D_{KL}^{in} , we consider a model encountering a test domain whose distribution differs from $P_{train}(x)$ and needs to be updated. Based on this, we introduce a new index $\frac{L_{img}}{D_{KL}^{in}}$. Fig. 3a plots the trend of this index during the model adaptation to a continually changing test domain. It shows that the index has a large value in the early stages of a domain change, decreases rapidly, and then maintains a value close to 1. This index exhibits a similar trend regardless of the backbone type and dataset, as included in the appendix. Therefore, we establish the criterion that model updates are necessary when this index exceeds a certain threshold, τ_1 , as $\frac{L_{img}}{D_{KL}^{in}} > \tau_1$.

(Criterion 2) When the distribution gap suddenly increases. Additionally, we can determine when the test distribution changes and model updates are necessary by observing the trend of the distribution gap (*i.e.*, L_{img}). The convergence of L_{img} indicates that a model is well-adapted to the current test domain. To put it differently, L_{img} will exhibit a sudden increase when the model encounters a new test domain. We introduce an additional index, denoted as $\frac{L_{img}}{L_{ema}^t}$, representing the ratio of the current L_{img} to its exponentially moving average L_{ema}^t at time t . We calculate it using the following formula: $L_{ema}^t = 0.99 \cdot L_{ema}^{t-1} + 0.01 \cdot L_{img}$. Fig. 3b illustrates the trend of the ratio of L_{img} over the timesteps. It tends to reach a value of 1 as the loss stabilizes at a specific level. Nevertheless, when the model encounters shifts in the test distribution, the ratio experiences a sharp increase, indicating the necessity of a model update when it exceeds a specific threshold, τ_2 , as $\frac{L_{img}}{L_{ema}^t} > \tau_2$.

If at least one of the two criteria is satisfied, we conclude that the model requires adaptation and proceed to update it.

4. Experiments

Sec. 4.1 presents the two object detection benchmark datasets with test distributions that change continuously, either in a drastic or gradual manner, and our implementation detail is in 4.2. Sec. 4.4 compares our method with other TTA baselines described in Secs. 4.3. We present detailed ablation studies of our method analyzing the effectiveness and efficiency of our method in terms of what, how, and when to update the models for CTAOD in Sec. 4.5.

4.1. Datasets

We experiment with the following three scenarios.

COCO \rightarrow COCO-C simulates continuous and drastic realistic test domain changes over a long sequence. MS-COCO [25] collects 80 classes of common objects in their natural context with 118k training images and 5k validation images. COCO-C is created by employing 15 types of realistic corruptions [29], such as image distortion and various weather conditions, to simulate test domain changes. In the experiments, the model is only trained on the COCO train set and sequentially evaluated on each corruption in the COCO-C validation set during test-time for reproducing continually changing test domains. Finally, the model is evaluated on the original COCO validation set to assess how well it preserves knowledge of the original domain (denoted as Org.). **SHIFT-(Discrete / Continuous)** [41] is a synthetic driving image dataset with 6 classes under different conditions using five weather attributes (*clear, cloudy, overcast, fog, rain*) and three time-of-day attributes (*daytime, dawn, night*). In **SHIFT-Discrete**, there are image sets for each attribute, and the model is sequentially evaluated on these attributes, *cloudy \rightarrow overcast \rightarrow foggy \rightarrow rainy \rightarrow dawn \rightarrow night \rightarrow clear* which contains 2.4k, 1.6k, 2.7k, 3.2k, 1.2k, 1.4k, and 2.8k validation images, respectively. This simulates scenarios where the domain undergoes drastic changes. In **SHIFT-Continuous**, the model is evaluated on four sequences, each consisting of 4k frames, continuously transitioning from *clear* to *foggy* (or *rainy*) and back to *clear*.

4.2. Implementation Detail

We conduct experiments on Faster-RCNN [35] with FPN [26], using ImageNet1k pre-trained ResNet50 [11] and ImageNet21k pre-trained Swin-Tiny [28] backbones. Specifically, Swin-Tiny is employed for its superior domain transfer [20] and detection efficacy over other advanced backbones [9]. For COCO \rightarrow COCO-C with ResNet50, we use a publicly available model trained with strong augmentation [48]. Other models are trained using detectron2 framework following [35] and [28]. For test-time adaptation, we always set the learning to 0.001 for the SGD opti-

Table 1. Comparison of mAP, the number of backward and forward passes, and FPS between baselines and our model on COCO → COCO-C. Our model consistently outperforms baselines on the two different backbones. Furthermore, Ours-Skip with ResNet notably reduces backward passes by as much as 90.5%, leading to a significantly improved frames per second (FPS) rate by up to 109.9%.

Backbone	Method	Noise			Blur				Weather				Digital				# step				
		Gau	Sht	Imp	Def	Gls	Mtn	Zm	Snw	Frs	Fog	Brt	Cnt	Els	Px	Jpg	Org.	Avg.	For.	Back.	FPS
Swin-T [28]	Direct-Test	9.7	11.4	10.0	13.4	7.5	12.1	5.2	20.7	24.8	36.1	36.0	12.9	19.1	4.9	15.8	43.0	17.7	80K	0	21.5
	ActMAD	10.7	12.0	9.4	12.3	5.7	9.5	4.5	15.3	17.5	27.6	28.2	1.1	16.7	2.6	8.7	36.3	13.9	80K	80K	8.3
	Mean-Teacher	10.0	12.1	11.2	12.8	8.1	12.1	4.9	19.6	23.7	34.9	34.0	8.0	18.9	6.1	17.6	41.0	17.2	160K	80K	6.9
	Ours	13.6	16.6	16.1	14.0	13.6	14.2	8.3	23.7	27.2	37.4	36.4	27.2	27.2	22.2	22.3	42.3	22.6	80K	80K	9.5
	Ours-Skip	13.3	15.3	15.1	14.0	12.8	13.9	6.5	22.0	25.4	35.5	34.9	26.5	25.9	23.4	20.2	41.2	21.6	80K	9.7K	17.7
ResNet50 [11]	Direct-Test	9.1	11.0	9.8	12.6	4.5	8.8	4.6	19.1	23.1	38.4	38.0	21.4	15.6	5.3	11.9	44.2	17.3	80K	0	25.8
	NORM	9.9	11.9	11.0	12.6	5.2	9.1	5.1	19.4	23.5	38.2	37.6	22.4	17.2	5.7	10.3	43.4	17.5	80K	0	25.8
	DUA	9.8	11.7	10.8	12.8	5.2	8.9	5.1	19.3	23.7	38.4	37.8	22.3	17.2	5.4	10.1	44.1	17.1	80K	0	25.8
	ActMAD	9.1	9.6	7.0	11.0	3.2	6.1	3.3	12.8	14.0	27.7	27.8	3.9	12.9	2.3	7.2	34.3	10.5	80K	80K	9.6
	Mean-Teacher	9.6	12.5	12.0	4.0	2.9	4.8	3.1	16.2	23.5	35.1	34.0	21.8	16.6	8.2	12.7	40.3	14.5	160K	80K	8.1
	Ours	12.7	17.8	17.5	12.4	11.5	11.3	6.6	22.8	26.9	38.6	38.5	28.0	25.1	21.2	22.2	41.8	22.2	80K	80K	10.1
Ours-Skip	14.4	17.1	16.0	13.9	11.7	12.2	6.3	22.1	25.5	37.7	37.1	25.5	24.1	23.1	21.1	42.8	21.9	80K	7.6K	21.2	

mizer, and α of Eq. 1 to 0.01, while τ_1 and τ_2 are set to 1.1 and 1.05, respectively. We use the same hyper-parameters across all backbones and datasets, with a batch size of 4.

4.3. Baselines

Direct-Test evaluates the model trained in the training domain without adaptation to the test domain. **ActMAD** [31] is a TTA method aligning the distribution of output features across all BN layers. To apply ActMAD to the Swin Transformer-based model, we align the output features of the LN layers. We implement **Mean-Teacher** using a teacher-student network framework to reproduce as close as possible to TeST [38], as its implementation is not publicly available. We follow the FixMatch [39] augmentation method and report results after tuning all hyper-parameters in our scenario. **NORM** [37] and **DUA** [30], TTA methods initially designed for classification, are directly applicable to detection tasks by either mixing a certain amount of current batch statistics or updating batch statistics via EMA. However, these are only compatible with architectures containing BN layers. Additional details are provided in Appendix.

4.4. Main Results

We compare the performance of each method using mAP and efficiency metrics, including the number of forward and backward passes, as well as FPS during test-time adaptation. Results of COCO and SHIFT are in Tab. 1 and 2, respectively.

COCO → COCO-C. Tab. 1 demonstrates the effective adaptation performance of *Ours* in the challenging COCO benchmark with 80 classes due to object-level class-wise feature alignment. *ActMAD* also aligns feature distribution for TTA, but is not effective since it only aligns whole feature maps without considering specific classes in the image. *NORM* and *DUA*, applicable only to ResNet [11], show minimal performance improvement by adaptation as they are not specifically tailored for object detection and only modify batch statistics across the entire feature map. Ad-

ditionally, *ActMAD* and *Mean-Teacher*, updating full parameters, gradually lose task knowledge in the continually changing test distributions, resulting in much lower performance on Org., the domain identical to the training data, than that of *Direct-Test*. In contrast, *Ours* effectively prevents catastrophic forgetting by freezing the original parameters of the models and updating only the adaptor, obtaining performance on par with *Direct-Test* on the Org. domain and consistently high performance across corrupted domains, with an average mAP improvement of 4.9%p compared to that of *Direct-Test*. Furthermore, leveraging the rapid adaptation ability of the adaptor, *Ours-Skip*, which skips unnecessary adaptation, allows using only a maximum of about 12% of the total samples for adaptation without significant performance loss. This leads to a substantial improvement in inference speed, more than doubling compared to other TTA methods, reaching over 17.7 FPS.

SHIFT-Discrete. *Ours* is also effective in SHIFT, which simulates continuous changes in weather and time in driving scenarios according to the left section of Tab. 2. Especially, *Ours* shows significant improvements in mAP by 7-9%p, particularly for the foggy and dawn attributes where *Direct-Test* obtains lower performance due to severe domain shift. In contrast, with ActMAD, catastrophic forgetting takes place when adapting to the cloudy and overcast weather. This is due to the updating of the full parameters, despite that *Direct-Test* already shows proficient performance in these conditions. As a result, the performance in the later domains is worse than that of the *Direct-Test*. *DUA*, which updates batch statistics using EMA, shows a gradual decrease in performance as the domain continuously changes, resulting in much lower performance in the original clear domain (*i.e.*, clear). On the other hand, *NORM*, which utilizes the statistics of the current batch samples, exhibits no catastrophic forgetting and relatively good adaptation, as SHIFT is a relatively easier task compared to COCO due to having only 6 classes. Compared to *NORM*, *Ours* shows better adaptation performance, and is

Table 2. Comparison of mAP, the number of backward and forward passes, and FPS between baselines and our model on SHIFT-Discrete and SHIFT-Continuous. Baselines perform effectively in a particular setting but lack generalizability across various settings. Our method consistently achieves results that are either better or on par with the best model in all settings, demonstrating its strong stability. Ours-Skip also effectively reduces the number of backward passes without compromising mAP performance, resulting in a higher FPS.

Backbone	Method	SHIFT-Discrete											SHIFT-Continuous				
		mAP							# step				mAP		# Avg. step		
		cloudy	overc.	fog	rain	dawn	night	clear	Avg.	For.	Back.	FPS	clear \leftrightarrow fog	clear \leftrightarrow rain	For.	Back.	FPS
Swin-T [28]	Direct-Test	50.0	38.9	23.1	45.1	26.9	39.5	45.9	38.5	15.3K	0	27.5	18.1	21.1	4K	0	28.3
	ActMAD	49.8	38.4	21.4	43.1	19.0	32.0	44.8	35.5	15.3K	15.3K	9.3	15.6	16.3	4K	4K	9.8
	Mean-Teacher	50.0	39.2	25.7	45.4	26.0	37.5	42.2	38.0	30.6K	15.3K	7.8	20.4	24.3	8K	4K	6.5
	Ours	50.3	39.2	32.2	46.7	30.4	39.9	44.3	40.4	15.3K	15.3K	11.2	23.9	22.6	4K	4K	11.6
	Ours-Skip	50.3	39.7	29.1	47.1	30.2	41.5	45.9	40.6	15.3K	6.1K	20.0	25.1	23.8	4K	0.83K	19.2
ResNet50 [11]	Direct-Test	49.4	37.9	19.7	43.1	20.1	35.3	45.6	35.9	15.3K	0	30.1	12.1	15.4	4K	0	30.0
	NORM	49.7	38.6	22.9	44.7	25.1	37.4	45.5	37.7	15.3K	0	30.1	16.9	19.4	4K	0	30.0
	DUA	45.2	31.5	27.7	31.9	15.2	18.6	21.1	27.3	15.3K	0	30.1	22.5	22.4	4K	0	30.0
	ActMAD	49.2	37.7	18.0	40.6	16.0	32.9	44.3	34.1	15.3K	15.3K	11.3	12.7	16.3	4K	4K	11.2
	Mean-Teacher	49.6	38.4	26.8	43.4	26.6	33.1	41.6	37.1	30.6K	15.3K	9.9	16.0	20.8	8K	4K	9.8
	Ours	49.7	38.7	27.4	46.3	27.4	37.6	43.8	38.7	15.3K	15.3K	12.9	20.9	21.9	4K	4K	13.9
	Ours-Skip	49.7	38.8	26.9	46.2	27.6	38.8	45.0	39.0	15.3K	8.9K	21.5	20.0	22.5	4K	0.75K	21.3

Table 3. Comparison of adaptation performance (mAP), the number of trainable parameters (# Params), and memory usage (Cache) according to which part of the backbone is updated. SD / SC denotes SHIFT-Discrete/Continuous, respectively.

Backbone	Trainable Params	mAP		# Params		Cache	
		SD	SC	Num	Ratio	Avg.	Max
Swin-T	Full-params	38.4	20.6	27.7M	100%	0.86	11.0
	LayerNorm	38.5	20.0	0.03M	0.1%	0.65	7.49
	adaptor (Ours)	40.4	23.2	0.15M	0.5%	0.65	6.96
ResNet50	Full-params	37.6	20.4	23.7M	100%	1.65	9.29
	BatchNorm	37.9	20.2	0.05M	0.2%	1.47	9.11
	adaptor (Ours)	38.7	21.7	0.21M	0.9%	1.48	5.41

also applicable to BN-layer-free Swin Transformers.

SHIFT-Continuous. In scenarios where the test domain gradually changes across the entire sequence, *Ours* also demonstrates effectiveness, improving mAP by up to 7%p, as shown in the right section of Tab. 2. While *DUA* performs well in the clear to foggy transition, it is prone to catastrophic forgetting in situations where the sequence becomes longer, and the test domain changes more diversely, as seen in the left section. Our strategy for determining when model adaptation is necessary is particularly effective in SHIFT. It improves FPS by about 9, reaching about 20 FPS, while enhancing mAP. This is likely due to avoiding overfitting that can occur when adapting to all repetitive frames in SHIFT, which consists of continuous frames, leading to improvements in both inference speed and adaptation performance.

4.5. Additional Analyses

We aim to demonstrate the effectiveness and detailed analysis of our proposed model in terms of 1) which parts of, 2) how, and 3) when the model should be updated.

Which part to update? Tab. 3 shows how updating different parts of the backbone model affects the performance and the memory usage during continual test-time adapta-

Table 4. Ablation on each component of our loss. SHIFT-D / C denotes SHIFT-Discrete / Continuous, respectively. The left and right value in each cell corresponds to the mAP for the Swin-T and ResNet50 backbone, respectively.

L_{img}	L_{obj}	COCO	SHIFT-D.	SHIFT-C.
-	-	17.7/ 17.3	38.5/ 35.9	19.6/ 13.8
✓	-	16.7/ 18.1	36.6/ 37.0	19.1/ 16.0
✓	no class weight	17.8/ 18.9	39.7/ 38.0	25.1/ 23.4
✓	class weight $w^{k,t}$	22.6/ 22.2	40.4/ 38.7	23.2/ 21.7

tion. We compare (1) updating full parameters, (2) affine parameters of the normalization layer, and (3) our proposed adaptor for each backbone on the SHIFT dataset. Although our adaptor has fewer parameters, about 0.9% or less of the full parameters, it demonstrates the best adaptation performance. Updating only the affine parameters of the normalization layer, while having fewer parameters, seems less effective for adaptation in object detection compared to classification [32, 45]. Additionally, our adaptor requires only about 60% of the memory compared to updating the full parameters, making it memory-efficient.

Ablation study on each component in our loss. Tab. 4 presents the effects of image-level feature alignment, L_{img} , object-level feature class-wise alignment L_{obj} , and class frequency weighting $w^{k,t}$ proposed to address class imbalance. Aligning only the image-level feature distribution with L_{img} (first row) leads to modest adaptation in the ResNet50 backbone, while performance in the Swin-T backbone is even lower than without adaptation. Notably, aligning object-level features with L_{obj} leads to a substantial improvement, with the mAP increasing by approximately 10%p compared to the no-adaptation scenario. Introducing class-specific frequency-based weighting $w^{k,t}$, despite a slight performance decrease in the SHIFT-Continuous setting, proves highly effective, particularly in scenarios with significant class imbalance, such as COCO

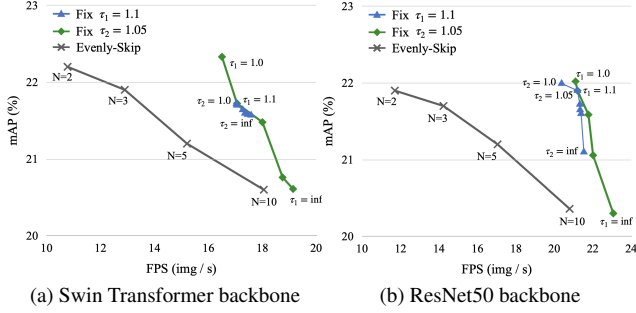


Figure 4. Comparison of mAP and FPS from *Ours-Skip* with varying values of τ_1 (◆) and τ_2 (▲) against *Evenly-Skip* (×), adapting every N -th instances, on COCO \rightarrow COCO-C using both (a) Swin-T and (b) ResNet50. The upward and rightward movement indicates a better strategy with higher mAP and faster inference speed, showing that *Ours-Skip* is consistently better than *Evenly-Skip*.

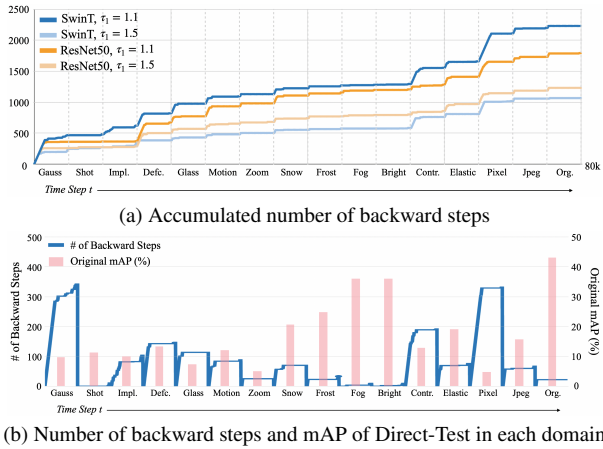


Figure 5. Analysis of the adaptation of *Ours-Skip*.

with 80 classes, where it enhances the mAP by around 5%p. **Trade-off between adaptation performance and efficiency according to different skipping strategies.** Fig. 4 presents mAP and FPS depending on the values of τ_1 and τ_2 in the Sec. 3.4 on COCO \rightarrow COCO-C, which are used for two criteria to determine when the adaptation is needed. We also show the simple baseline *Evenly-Skip*, which adapts every N -th step and skips the rest. In Fig. 4, the blue lines (▲) show the results when τ_1 is changing from 1.0 to infinity, where only criterion 2 is used, while τ_2 is fixed at 1.05. As τ_1 decreases, more adaptation is required, leading to slower FPS but higher mAP. The green lines (◆) show the results of changing τ_2 , where ‘ $\tau_2 = \text{inf}$ ’ denotes using only criterion 1, without criterion 2. For all main experiments, we set τ_1 and τ_2 as 1.1 and 1.05, respectively, considering the balance between mAP and FPS. Additionally, our skipping strategy consistently outperforms *Evenly-Skip*, achieving higher values in both mAP and FPS. This indicates that our criterion for deciding when to bypass model updates provides an effective balance between accuracy and speed.

When do models actually update? We analyze when the model actually skips adaptation and only performs inference or actively utilizes test samples for model adaptation based on the two criteria we propose. This analysis is conducted in COCO to COCO-C with 15 corruption domains and 1 original domain. Fig. 5a plots the number of backward passes, *i.e.*, the number of batches of test samples used for adaptation, with different values of τ_1 for the two backbones. The horizontal and vertical axes represent sequentially incoming test domains and the cumulative backward numbers, respectively. A steep slope in a region indicates frequent adaptation, while a gentle slope indicates skipping adaptation, performing only inference. Notably, even without explicit information about when the test domain changes, the model actively performs adaptation, especially right after the test domain changes. This trend is consistent regardless of changes in τ value or backbone type. The number of backward passes largely depends on the value of τ_1 rather than the backbone type, suggesting that a consistent τ_1 value can be used irrespective of the backbone. Fig. 5b illustrates adaptation patterns by dividing backward steps for each domain in the case of Swin-T backbone with $\tau_1 = 1.1$. More clearly, it shows that adaptation occurs actively around the points where each domain changes, and afterward, adaptation happens intermittently or almost not at all. The light pink bars represent the performance of *Direct-Test*, showing that domains with initially high model performance tend to have less adaptation, while domains with lower performance initially need more adaptation. In other words, the amount of skipping adaptation is proportional to the amount of the domain shift. Interestingly, the second domain, ‘Shot Noise’, shows almost no adaptation despite the lower performance of the *Direct-Test*. We conjecture that the preceding domain, ‘Gaussian Noise’, shares a similar nature of noise, leading the model to decide that additional adaptation steps may not be necessary. As a result, our skipping strategy enables the model to efficiently adapt, considering both the original domain the model is trained on and the previous domain the model has been adapted to.

5. Conclusion

We introduce an efficient Continual Test-time Adaptation (CTA) method for object detection in the continually changing domain. Our approach involves 1) lightweight adaptors, 2) class-wise object-level feature alignment, and 3) skipping unnecessary adaptation, offering highly efficient and effective solution to diverse domain shifts. It significantly improve mAP performance in various CTA scenarios without serious slowdown in the inference speed.

Acknowledgement This work was supported by NRF grant (2021R1A2C3006659) and IITP grants (2021-0-01343, 2022-0-00320, 2019-0-00079), all funded by the Korea government (MSIT).

References

- [1] Alexander Bartler, Florian Bender, Felix Wiewel, and Bin Yang. Ttaps: Test-time adaption by aligning prototypes using self-supervision. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022. 2
- [2] Alexander Bartler, Andre Bühler, Felix Wiewel, Mario Döbler, and Bin Yang. Mt3: Meta test-time training for self-supervised test-time adaption. In *International Conference on Artificial Intelligence and Statistics*, pages 3080–3090. PMLR, 2022. 2
- [3] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8344–8353, 2022. 1
- [4] Dhanajit Brahma and Piyush Rai. A probabilistic framework for lifelong test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3582–3591, 2023. 3
- [5] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022. 3
- [6] Yijin Chen, Xun Xu, Yongyi Su, and Kui Jia. Sftar: Improving object detection robustness at test-time by self-training with feature alignment regularization. *arXiv preprint arXiv:2303.17937*, 2023. 1, 2, 3
- [7] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4091–4101, 2021. 3
- [8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 3
- [9] Micah Goldblum, Hossein Souri, Renkun Ni, Manli Shu, Viraj Prabhu, Gowthami Somepalli, Prithvijit Chattopadhyay, Mark Ibrahim, Adrien Bardes, Judy Hoffman, et al. Battle of the backbones: A large-scale comparison of pretrained models across computer vision tasks. *Advances in Neural Information Processing Systems*, 36, 2024. 5
- [10] Shirsendu Sukanta Halder, Jean-François Lalonde, and Raoul de Charette. Physics-based rendering for improving robustness to rain. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10203–10212, 2019. 3
- [11] 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. 5, 6, 7, 2
- [12] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 3
- [13] Junyuan Hong, Lingjuan Lyu, Jiayu Zhou, and Michael Spranger. Mecta: Memory-economic continual test-time model adaptation. In *The Eleventh International Conference on Learning Representations*, 2022. 3
- [14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 3
- [15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015. 2
- [16] Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for modelagnostic domain generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3
- [17] Minguk Jang, Sae-Young Chung, and Hye Won Chung. Test-time adaptation via self-training with nearest neighbor information. In *International Conference on Learning Representations (ICLR)*, 2023. 3
- [18] Sanghun Jung, Jungsoo Lee, Nanhee Kim, Amirreza Shaban, Byron Boots, and Jaegul Choo. Cafa: Class-aware feature alignment for test-time adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 2, 3, 4
- [19] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 480–490, 2019. 3
- [20] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. A broad study of pre-training for domain generalization and adaptation. In *European Conference on Computer Vision*, pages 621–638. Springer, 2022. 5
- [21] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Chang-ick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6092–6101, 2019. 3
- [22] 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. 4
- [23] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation, 2017. 3
- [24] Hyesu Lim, Byeonggeun Kim, Jaegul Choo, and Sungha Choi. Ttn: A domain-shift aware batch normalization in test-time adaptation, 2023. 2, 3
- [25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 5
- [26] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 5

- [27] Yuejiang Liu, Parth Kothari, Bastien Van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34: 21808–21820, 2021. [1](#), [2](#)
- [28] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. [2](#), [5](#), [6](#), [7](#)
- [29] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019. [5](#)
- [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. [2](#), [6](#), [1](#)
- [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](#), [2](#), [3](#), [4](#), [6](#)
- [32] Shuaicheng Niu, Jiayang Wu, Yifan Zhang, Yafo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pages 16888–16905. PMLR, 2022. [1](#), [2](#), [3](#), [7](#)
- [33] Shuaicheng Niu, Jiayang Wu, Yifan Zhang, Zhiquan Wen, Yafo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*, 2023. [3](#)
- [34] Mario obler, Robert A Marsden, and Bin Yang. Robust mean teacher for continual and gradual test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7704–7714, 2023. [3](#)
- [35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. 2016. [5](#)
- [36] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. [3](#)
- [37] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33:11539–11551, 2020. [2](#), [6](#), [1](#)
- [38] 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](#), [2](#), [3](#), [6](#)
- [39] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020. [2](#), [6](#)
- [40] Yongyi Su, Xun Xu, and Kui Jia. Revisiting realistic test-time training: Sequential inference and adaptation by anchored clustering. *Advances in Neural Information Processing Systems*, 35:17543–17555, 2022. [3](#), [4](#)
- [41] Tao Sun, Mattia Segu, Janis Postels, Yuxuan Wang, Luc Van Gool, Bernt Schiele, Federico Tombari, and Fisher Yu. Shift: a synthetic driving dataset for continuous multi-task domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21371–21382, 2022. [5](#)
- [42] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020. [1](#), [2](#)
- [43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. [2](#)
- [44] Vibashan VS, Poojan Oza, and Vishal M Patel. Towards online domain adaptive object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 478–488, 2023. [3](#)
- [45] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020. [1](#), [2](#), [3](#), [7](#)
- [46] 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. [3](#)
- [47] Zehao Xiao, Xiantong Zhen, Shengcai Liao, and Cees GM Snoek. Energy-based test sample adaptation for domain generalization. *arXiv preprint arXiv:2302.11215*, 2023. [3](#)
- [48] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. [5](#)
- [49] Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in Neural Information Processing Systems*, 35: 38629–38642, 2022. [1](#), [2](#), [3](#)
- [50] Bowen Zhao, Chen Chen, and Shu-Tao Xia. Delta: Degradation-free fully test-time adaptation. In *International Conference on Learning Representations (ICLR)*, 2023. [2](#), [3](#)