Efficiency-preserving Scene-adaptive Object Detection

Zekun Zhang[1]
zekzhang@cs.stonybrook.edu
Vu Quang Truong[2]
v.truongvq1@vinai.io
Minh Hoai[3]
mh.nguyen@adelaide.edu.au

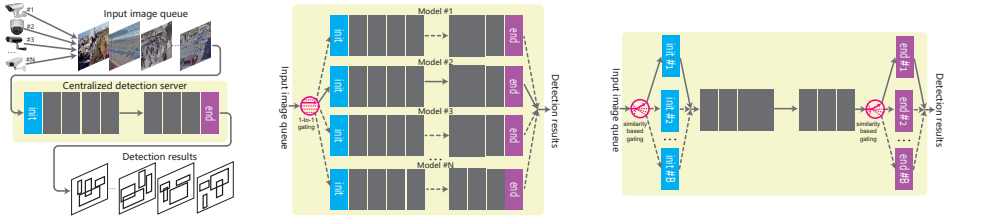
[1] Stony Brook University, Stony Brook, NY 11790, USA
[2] VinAI Research, Hanoi, Vietnam
[3] The University of Adelaide, Adelaide, SA 5069, Australia

Abstract

We present a framework that enables an object detector to self-enhance its accuracy while preserving its efficiency. This framework is particularly useful in settings where a single object detector is deployed to detect objects in video streams from numerous cameras. Our approach improves the object detector's precision by adapting it to specific scenes in a novel way that does not hinder the inference speed or overall system throughput. Specifically, it involves augmenting the object detector with a mixture-of-experts structure that only moderately increases the parameter count, avoiding the expense of replicating the entire model. The resulting enhanced detector operates as a self-contained unit, facilitating an efficient client-server architecture with a shared detection engine for multiple video streams. Our framework supports self-supervised learning, eliminating the reliance on manually annotated data, and it is compatible with various established object detector architectures. Experiments on the Scenes100 dataset demonstrate the wide applicability and effectiveness of our method in enhancing detection precision while maintaining operational efficiency. Our code is available at https://github.com/cvlab-stonybrook/scenes100/tree/main/moe.

1 Introduction

Real-time object detection in video feeds is critical for various computer vision applications, including anomaly detection in security systems, obstruction spotting on railway lines, and vehicle detection for traffic monitoring. While detection accuracy is critical for these applications to function correctly, one also needs to be mindful of the required computational resources so that the costs do not outweigh the benefits. Optimizing for detection accuracy within computational resource constraints is challenging. Smaller and quantized networks require less computation and can reduce cost. However, they also have limited representation capacity and generalization ability. This issue becomes even more pronounced when analyzing video streams from multiple scenes with a single detector, as these scenes can vary greatly in perspective, lighting, and appearance. One approach to address the limited capacity challenge is using a set of scene-specific detectors over a single scene-generic detector.



(a) Model-serving architecture for scene-adaptive object detection

(b) Dedicate a scene-specific detector to adapt to each scene

(c) Proposed mixture-of-experts detector adapts to all scenes simultaneously

Figure 1: The centralized model serving architecture and two scene adaptation approaches. (a) The centralized model serving architecture involves hosting a detector on a centralized server that serves multiple video streams. This setup is preferred for cost efficiency in scenarios with many video streams. (b) The conventional adaptation approach, generating a separate model copy for each scene, significantly increases memory usage and decreases processing throughput. (c) The proposed sparse mixture-of-experts (MoE) approach with a budget of B. At each branching point, a similarity-based gating module routes the image to an expert. This method preserves the computational cost per image of the base model. The increase in the number of parameters is only sublinear relative to the number of scenes. The maximum number of video streams that can be supported depends solely on the computational speed of the inference hardware.

Since each detector is dedicated to a particular scene, the detection accuracy can be potentially improved. However, the naive method [39] of training a separate detector for each scene still requires a multiplied amount of labeled training data. It also prevents the use of efficient model serving architectures with a shared inference engine [18, 19, 20, 53, 57, 67]. Each detector would be tied to a separate inference engine and consume a portion of the memory, as shown in Figure 1a. Consequently, the number of video streams that can be processed simultaneously by the same inference hardware is limited by the total available memory, thereby reducing the system's throughput and increasing the total cost.

In this paper, we address the challenge of adapting a base object detector to diverse scenes, with the aim of improving detection precision while preserving inference speed and system throughput. Our method begins with the network architecture of a base detector, which we enhance with a sparse mixture-of-experts structure. In contrast to the significant overhead of replicating the entire model, only a small number of parameters are added. This enhanced detector operates as a single entity, enabling an efficient client-server architecture with a shared inference engine for multiple video streams. It is capable of adapting to diverse scenes via self-supervised learning, eliminating the need for manual annotation. This framework is compatible with various object detector architectures, including Faster-RCNN [73], YOLOv8s [42], and DINO-5scale [88]. Our experiments on a scene-adaptive object detection dataset [39] demonstrate that our proposed method substantially surpasses existing models, delivering improved detection accuracy while maintaining runtime efficiency.

Note that our focus is to enhance the detector deployed on a centralized processing server. Centralized AI processing has many benefits compared to distributed setups in scenarios with numerous video streams, even with the increasing prevalence of AI cameras [26, 62, 68]. AI-enabled cameras require expensive processing chips, and the model is hard to update. In contrast, a centralized processing server can support many cheaper, regular cameras, and updating a centralized detection model is relatively easier.

2 Related Work

Model efficiency is a key consideration of our work. Various approaches can improve efficiency, including designing efficient architectures [35, 37, 39, 79], neural architecture search-

ing [22], quantizing the parameters [27], and pruning network parameters [8]. Our proposed method complements these approaches and can be applied to various network architectures, including those that are optimized, quantized, or pruned. Recently LoRA [1, 36, 92, 93] has been utilized to fine-tune trained models for new tasks with reduced memory usage. Our proposed method has the advantage of not increasing memory costs as the number of adapted scenes grows. Additionally, LoRA can be integrated as a plugin if memory consumption remains a concern after implementing our method.

Scene-adaptive object detection is the main objective of our work, which is considered in [89] as the task to improve the detection precision of a trained base detector on video streams from a set of diverse scene cameras, addressing the domain shift problems typically encountered by any pre-trained object detector, e.g., [34, 63, 64, 65, 66]. It is closely related to domain-adaptive object detection [15, 17, 32, 38, 40, 43, 45, 46, 47, 48, 52, 61, 74, 75, 78, 85, 86, 87, 90]. Most existing methods do not take efficiency into consideration, thus often result in increased computational demands. Our approach is designed with computational resource limitations and scalability in mind while still outperforming methods that solely concentrate on improving detection accuracy.

Mixture-of-experts (MoE) [2, 11, 16, 21, 24, 77, 84] is the network architecture utilized by the proposed method to achieve both efficiency and scene-adaptability. An MoE model contains a set of experts, and a gating module activates a subset of the experts according to each input sample. We show that only a small portion of a detector network is scene-specific and needs to be converted to MoE, greatly reducing the parameter count. Our gating module is sparse and only activates one expert for each input image, so the computational cost for both training and inference is the same as the non-MoE base model. We also apply a two-stage training schedule to mitigate the overfitting issue caused by MoE and gating.

Knowledge distillation is the primary technique used by the proposed framework to actuate scene-adaptation training. It was originally proposed [33] to transfer the learned information from a teacher network to another student network. Self-distillation refers to the case when the teacher and student have the same architecture, and it achieves remarkable results in self-supervised vision representation learning [8, 12, 13, 14, 31, 58]. Many of such frameworks apply the idea of contrastive learning, which involves different data augmentations on the input of the teacher and the student. In our proposed self-supervised adaptation method, the teacher and the student also have identical architectures and weights at the beginning of training. We apply upscaling as the data augmentation for the teacher. Comparable methods are used in self-supervised domain-adaptive object detectors [10, 17, 45, 48, 91]. Our proposed method does not require complicated data augmentation methods, nor careful tuning of the hyper-parameters. Yet, it still achieves a significantly higher detection precision boost. Self-distillation methods can suffer from model collapse, which requires special treatments such as negative pair sampling, regularization, or diversity enforcement. We avoid this issue by not updating the teacher during training.

3 Efficiency-Preserving Scene Adaptation

This section describes our proposed framework which enhances an object detector with the capability of self-adapting to many different scenes while also preserving its suitability as the core component of a shared inference engine for all scenes. It utilizes a sparse mixture-of-experts (MoE) strategy, which assigns each video stream to its corresponding route within the model to improve adaptability. Only one expert is activated for each input image. It ensures

that the model has enough capacity for scene-specific adaptation to improve its precision, while maintaining both the processing latency and the throughput of the inference engine.

Problem definition. Given a base detector $\mathcal{M}$, which is the core component of an inference engine that processes multiple camera streams from diverse scenes, instead of creating a separate model for each scene which linearly increases the overall parameter count, we aim to obtain an enhanced model $\mathcal{M}^*$ to replace $\mathcal{M}$, while ensuring the following characteristics. **(1) Improved precision**: $\mathcal{M}^*$ should have the capacity to adapt to different scenes, thereby providing improved detection precision over $\mathcal{M}$. **(2) Consistent latency and throughput**: On the same inference hardware, the time $\mathcal{M}^*$ takes to process each image should be equivalent to that of $\mathcal{M}$. $\mathcal{M}^*$ should also match $\mathcal{M}$ in its ability to process the same number of video streams at an identical frame rate. **(3) Memory efficiency**: The parameter count for $\mathcal{M}^*$ must not be excessively large, even with a large number of scenes, ensuring that it can operate on the existing inference hardware without the need for additional memory. And **(4) Self-supervised learning**: $\mathcal{M}^*$ should adapt to the scenes and through self-supervised learning, eliminating the necessity for manually-labeled data or human oversight.

3.1 Architecture of Enhanced Model

To design a detector with the aforementioned characteristics, we divide the detector's modules into scene-generic and scene-specific ones. Scene-generic modules are shared by all scenes, while each scene-specific module only adapts to a subset of scenes. This enables the model to adapt to diverse scenes while scaling sublinearly with the number of scenes. It allows for serving as many concurrent video streams as possible, limited only by the computational speed capacity of the inference hardware and not by additional memory constraints.

Most of the existing object detection architectures [6, 7, 9, 28, 29, 41, 42, 44, 50, 51, 70, 71, 72, 73, 83] are composed of two primary components: (1) a feature extractor network that generates feature maps from images, and (2) a detection head for localization and category classification of object instances. For a collection of diverse scenes, it is apparent that the most significant differences between them are object sizes, camera perspectives, and lighting conditions. Consequently, the most appropriate layers for scene-specific adaptation are those closest to the input or output. In particular, the initial layers of the feature extractor play a pivotal role in adjusting to geometric and lighting variations, and the terminal layers of the head are crucial for finalizing the detector's judgments. The intermediary layers, which deal more with abstract visual representation and semantics, tend to differ less across different scenes. This division is shown in Figure 1b. These scene-specific modules usually comprise only a few percent of the network's total parameters. In §4.3 we show that using scene-specific modules benefits scene-adaptation.

We enhance the base detector by selectively duplicating these scene-specific modules, as illustrated in Figure 1c. Let $|\mathcal{M}|$ be the parameter count of the original model, α the parameter proportion of scene-specific modules, and each of the scene-specific modules is duplicated for B times. B can be smaller than the number of scenes N, as a single module can still adapt to multiple similar video streams adequately. The parameter count of the enhanced MoE model is: $|\mathcal{M}^*| = (1+\alpha(B-1))|\mathcal{M}|$, which is significantly smaller than $B|\mathcal{M}|$.

The processing path for each input image in the MoE model $\mathcal{M}^*$ depends on the unique *scene ID* indicating the camera from which the image originates. At each branching junction within $\mathcal{M}^*$, a gating module determines the image's route based on its scene ID. This sparse mixture-of-experts approach ensures that, despite the branching, the computational cost for processing one image remains unchanged from the base detector $\mathcal{M}$. In this paper, we apply

Algorithm 1: Similarity-based gating (B-Means)

Input: N video streams and a feature extractor $\mathcal{F}$
Parameters: Branching budget B and the number of samples per scene M
Output: Branch assignments for each scene $a_1, \ldots, a_N$, where $1 \leq a_i \leq B$

1 Sample M frames from each of N scenes: $\{X_{ij} | 1 \leq i \leq N, 1 \leq j \leq M\}$.
2 Extract feature vectors: $f_{ij} \leftarrow \mathcal{F}(X_{ij})$.
3 Run k-means on $\{f_{ij}\}$ with $k = B$, get the cluster assignments $\{k_{ij}\}$.
4 Assign branch IDs to scenes using voting: $a_i \leftarrow \text{argmax}_{1 \leq k \leq B} \sum_{j=1}^{M} \delta(k_{ij} = k)$.

consistent gating rules across all gating modules in $\mathcal{M}^*$. This means the number of parallel modules at each branching point equals B, resulting in a total of B distinct paths in $\mathcal{M}^*$. It is feasible to vary the number of branches at different branching points and combine a branch from one point with an arbitrary one from another, yielding a combinatorial number of pathways. We reserve such considerations for future work.

3.2 Similarity-Based Gating

The MoE detector $\mathcal{M}^*$ is designed to handle B distinct processing paths. B is a hyperparameter adjusted for system memory constraints. When memory is very limited, we can set B=1, effectively adapting a single model to all scenes. Conversely, with more memory, B could match the total number of scenes N, though it is commonly unnecessary. When $1 < B < N$, effective scene-to-branch assignment is crucial. As random allocation can be suboptimal, we propose a data-driven clustering strategy to assign similar scenes to the same branch. M images are sampled from each of the N scenes to compile a representative set on which the grouping is based on. A feature extractor $\mathcal{F}$ then calculates an image-level feature vector for every image. k-means clustering is performed on the feature vector set. Then, for every scene, we evaluate the cluster IDs assigned to its sample images and select the one for the entire scene using majority voting. This algorithm for scene-to-branch assignment, named *B-Means*, is outlined in Algorithm 1. It is desirable to use a feature extractor $\mathcal{F}$ that already demonstrates effective object detection performance. In our experiments, we take the backbone of the warmed up model described in §3.3 as $\mathcal{F}$. In §4.3 we verify that B-Means improves performance over random allocation.

3.3 Training for Adaptation

Adapting the MoE detector to different scenes requires images from those scenes with corresponding training signals to update the parameters of the base detector. The images are already produced by the cameras, and various strategies can be used to generate self-supervised pseudo annotation. For instance, tracking [3, 4, 23, 54, 55, 56, 76] aids in identifying missed detections, as enforcing temporal persistence reduces false positives. Model ensembles [25, 59] enhance accuracy but demand more computational power. Alternatively, input data augmentation [60, 69] transforms the original image into variants, each analyzed by the

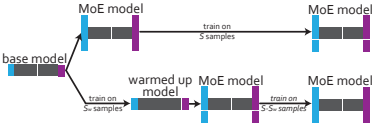


Figure 2: Two-stage (bottom) compared with one-stage (top) training schedule. With the same total of S training samples, the model is first trained on S_w samples without branching to obtain the warmed up single model. Then, it is enhanced with MoE branching and further training on $S - S_w$ samples. Two-stage training mitigates the overfitting issue brought by MoE and gating.

Model	Backbone	COCO	Scenes100
Faster-RCNN [23]	R-101	52.77	41.96
Faster-RCNN [23]	R-18	46.96	35.68
YOLOv8s [41]	D-53	48.61	44.10
DINO-5scale [53]	R-50	54.65	40.54

Table 1: Mean Average Precision (AP^m) of base models on COCO2017-val [19] and Scenes100 [39]. Models are trained on COCO2017 training set with remapped object classes. R-101, R-50, R-18, and D-53 stand for ResNet-101, ResNet-50, ResNet-18, and CSPDarkNet-53 backbones, respectively.

base detection model. The aggregated results from all variants can be used as the pseudo annotation. Employing tracking, model ensembles, and input augmentation individually or in combination could likely yield more reliable labels than the base detector's outputs.

However, determining the optimal pseudo label generation method is not the primary focus of this work. We show that for several state-of-the-art detectors, self-distillation can significantly increase the precision of the MoE model. The base model serves as the teacher network, and the adapted MoE model is the student. Teacher and student networks have the same parameters at the beginning of adaptation training, while the input of the teacher is augmented by bilinear upscaling. Upscaling greatly improves the detection precision of the base model, likely because in the tested dataset, the object is considerably smaller than the dataset on which the base models are trained, and upscaling narrows this data distribution gap. Pseudo labels for training the student network can be generated by keeping detected objects from the teacher with confidence scores above a threshold θ. Please note that the input images of the student network keep their original size to preserve runtime efficiency. We avoid the model collapse issue of self-distillation by not updating the teacher during training, which is shown to be beneficial by experiments in the supplementary material. In the supplementary material, we also examine the effectiveness of our method with pseudo labels generated through tracking and model ensembles [39].

Branching certain modules of a model can enhance its ability to adapt to diverse scenes, but it also potentially causes overfitting. In the vanilla model, every parameter is exposed to all the training samples. In contrast, with the same number of training samples, each training sample follows only one path during training in an enhanced MoE model, resulting in each of the parallel modules being exposed to far fewer samples. We propose a two-stage training schedule for the MoE model. The first stage involves training a vanilla model without branching on samples from all scenes, producing the so-called "warmed up" model. In the second stage, this warmed up model is enhanced with MoE and further trained with gating on the remaining samples. This *two-stage* training schedule is depicted in Figure 2. In §4.3, we show that two-stage training results in higher detection performance over one-stage training while using the same number of total training samples.

4 Experiments

This section describes our extensive experiments, starting with the dataset, implementation details, and baselines. Afterwards, it compares the performance of the proposed method with the baselines in terms of both precision and efficiency.

4.1 Dataset, Evaluation Metrics, and Baselines

We use Scenes100 [39] for the experiments. It is the only publicly available dataset with a sufficient number of lengthy videos and ample bounding boxes for scene-adaptive detection research. Following [39], we evaluate a detector's precision by calculating the per-class mean Average Precision score across different IoU thresholds from 0.5 to 0.95 (AP^m), and then

averaged weighted by the prevalence of the instances of each class. Adaptation starts from the two-class base detector in [89], which is trained on COCO2017 [49] training set with remapped object classes *person* and *vehicle*. Most of the baseline methods aim to achieve high detection accuracy without considering efficiency, so they are based on bigger models such as Faster-RCNN [73] with a large backbone. In the experiments, for comparison with baseline methods, we use base Faster-RCNN models on ResNet-101 [30] backbone and use the weights provided in [89]. We test the proposed method on several other object detector architectures, including Faster-RCNN with a smaller ResNet-18 backbone, the lightweighted YOLOv8s [42], and a transformer-baed DINO-5scale [88]. For the base models of these architectures, we train them using the same protocol on COCO2017 until convergence.

Table 1 shows the AP^m scores of the base detectors on both the COCO2017 validation set and Scenes100. For Faster-RCNN models, the larger R-101 backbone enables significantly better performance on COCO2017. YOLOv8s is considerately smaller, thus achieving AP^m more comparable to the smaller R-18 backbone. DINO-5scale, being the most computationally heavy, also achieves the highest AP^m. When comparing the performance on COCO2017 and Scenes100, Faster-RCNN and DINO-5scale models see a AP^m drop of more than 10 points. But AP^m of YOLOv8s only drops about 4 points. This is likely because the objects in Scenes100 are considerably smaller in scale compared to COCO2017. YOLOv8s is more accurate at detecting smaller objects, making it less prone to the data distribution shift. This also implies that YOLOv8s might see less improvement in detection precision from upscaling-based self-distillation discussed in §3.3.

In adaptation training, we apply $\times 2$ upscaling and score thresholding as described in §3.3 to generate pseudo labels. We use the corresponding standard loss functions for each detector architecture. The efficiency of models is measured using relative inference latency, with the reference being the base Faster-RCNN with R-101 backbone. More details on upscaling, thresholding, training schedules, hardware configuration, absolute latency measures, and experiments using different batch sizes can be found in the supplementary material.

We compare our proposed framework with several domain-adaptive and scene-adaptive object detection methods as follows. Self-Train (**ST**) [74] uses detection and tracking to obtain pseudo bounding boxes, referred to as *DtTr*, for self-supervised adaptation. Cross-Teach (**CT**) [89] further utilizes an ensemble of base detectors, and we refer to these pseudo labels as *EnDtTr*. Mid-Fusion with location-aware Mixup (**MFM**) [89] exploits scene consistency from fixed scene cameras by modeling the background as an additional input modality and applies artifact-free object mixup for data augmentation. We directly use the AP^m numbers reported in the original paper for comparison. Geometric Shift (**GS**) [82] aims to correct the distortion from the camera perspective by learning a set of homography transforms. Learning to Zoom and Unzoom (**LZU**) [80, 81] is a differentiable plugin designed to zoom in on specific parts of the input image. **LODS** [45] is a source-free domain-adaptive object detection method in which a teacher network generates pseudo labels to train the student network. Details of the implementation of the baselines can be found in the supplementary material. Other baselines [48, 78, 86, 90] are shown to perform poorly on Scenes100 by [89], so we do not include results from them.

4.2 Comparison of Precision and Efficiency

We first compare the proposed adaptation method with the baseline methods in Table 2a. All models are adapted from the same Faster-RCNN R-101 base model for fair comparison. ST is trained on DtTr labels. CT, MFM, GS, and LZU are all trained on the same EnDtTr pseudo

Method	Pseudo label	Training samples ↓	Deployment model size ↓	GFLOPs ↓	Relative latency ↓	AP^m ↑
Base model (no adaptation)	Not applicable		230MB	558	1.00	41.96
ST [74]	DtTr	8.00M	230MB×100	558	1.00	43.35
CT [33]	EnDtTr	8.00M	230MB×100	558	1.00	43.63
MFM [30]	EnDtTr	8.00M	230MB×100	987	1.75	45.74
GS [33]	EnDtTr	8.00M	231MB×100	1440	2.71	44.06
LZU [33]	EnDtTr	8.00M	230MB×100	558	1.20	44.06
LODS [15]	teacher	0.10M	230MB×100	558	1.00	42.98
Proposed (B=10)	×2	1.08M	259MB	558	1.01	50.27
Proposed (B=100)	×2	1.08M	547MB	558	1.00	**50.39**

(a) Comparison of the proposed method with other adaptation methods. All models are trained using the same Faster-RCNN base model with an R-101 backbone. The proposed models with B=10 and B=100 perform similarly, outperforming other methods by a wide margin. It does not increase computational costs or consume significantly more memory, as some baselines do.

Base model	Method	Pseudo label	Training samples ↓	Deployment model size ↓	GFLOPs ↓	Relative latency ↓	APD^m ↑
Faster-RCNN with R-101 backbone	1-for-100	×2	1.08M	230MB	558	1.00	0
	1-for-100 (no adapt)	Not applicable		230MB	558	1.00	−7.70
	1-for-1×100	×2	1.08M	230MB×100	558	1.00	0.42
	Proposed (B=10)	×2	1.08M	259MB	558	1.01	0.61
	Proposed (B=100)	×2	1.08M	547MB	558	1.00	**0.73**
Faster-RCNN with R-18 backbone	1-for-100	×2	1.08M	107MB	310	0.54	0
	1-for-100 (no adapt)	Not applicable		107MB	310	0.54	−8.96
	1-for-1×100	×2	1.08M	107MB×100	310	0.54	0.37
	Proposed (B=10)	×2	1.08M	134MB	310	0.54	**0.90**
	Proposed (B=100)	×2	1.08M	397MB	310	0.54	0.69
YOLOv8s with D-53 backbone	1-for-100	×2	0.95M	43MB	73	0.22	0
	1-for-100 (no adapt)	Not applicable		43MB	73	0.22	−1.63
	1-for-1×100	×2	0.95M	43MB×100	73	0.22	**0.98**
	Proposed (B=10)	×2	0.95M	55MB	73	0.23	0.73
	Proposed (B=100)	×2	0.95M	174MB	73	0.24	0.75
DINO-5scale with R-50 backbone	1-for-100	×2	0.02M	181MB	1620	5.58	0
	1-for-100 (no adapt)	Not applicable		181MB	1620	5.58	−8.96
	1-for-1×100	×2	0.02M	181MB×100	1620	5.58	1.34
	Proposed (B=10)	×2	0.02M	198MB	1620	5.71	**2.45**
	Proposed (B=100)	×2	0.02M	373MB	1620	5.85	1.95

(b) Comparison of the proposed method with the baselines of adapting a single generic *1-for-100* model to all scenes and creating a scene-specific *1-for-1* model for each scene for adaptation. State-of-the-art detector architectures (Faster-RCNN [74], YOLOv8s [40], and DINO-5scale [33]) are tested. APD^m represents the difference in mean average precision compared to adapted 1-for-100 model with the same detector architecture. The proposed method achieves similar or better performance compared to 1-for-1×100 models, while consuming significantly less memory and maintaining the same computational cost as the base model.

Table 2: Comparison of different adaptation methods in terms of detection performance on Scenes100 [33] and computational cost in terms of giga floating-point operations (GFLOPs) and inference latency. Pseudo label column shows the labels used in adaptation, being either *DtTr* from [74], *EnDtTr* from [33], *teacher* labels from a teacher network, or ×2 meaning using upscaled image for the base model to generate pseudo labels for self-adaptation. All models take images without upscaling at inference time. Training samples column shows the total number of images seen by the models during adaptation. It is the sum of all individual models in the case of individual adaptation. Please refer to §4.1 and §4.2 for more details on evaluation metrics, latency measurement, and input scale.

labels. ST, CT, and LODS cannot effectively improve the detection precision over the base model. GS and LZU both utilize geometric transforms, so the performance is higher. MFM introduces background extraction and mixup, which significantly improve the performance. The proposed B=10 model uses two-stage training with B-Means, and the B=100 model uses two-stage training with 1-to-1 gating. Both models achieve similar detection precision after adaptation, which is significantly higher than all baselines. All the baseline methods adapt to individual scenes by creating a separate model for each scene (1-for-1 approach). This strategy results in significant memory consumption and requires a large number of training samples (except for LODS) for each model to achieve convergence. In contrast, the proposed method, being both data and computation efficient, has a significantly smaller overall model size, requiring much fewer training samples. At inference time, MFM uses two backbone passes. GS and LZU introduce additional geometrical transforms or backbone passes. The increased computational cost is reflected in increased GFLOPs and latency. The proposed

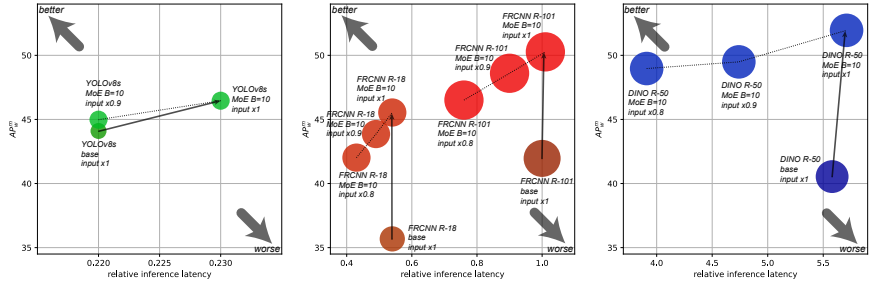


Figure 3: AP^m and latency of B=10 MoE models for different detector architectures at different input scales. The size of each model is indicated by the area of the corresponding circle. Sub-figures show the same limits for the vertical axis (AP^m) but different limits for the horizontal axis (relative inference latency). For all architectures, our framework can improve the base models in detection precision and efficiency at the same time, while keeping similar memory consumption.

method can maintain the computational cost of the base model.

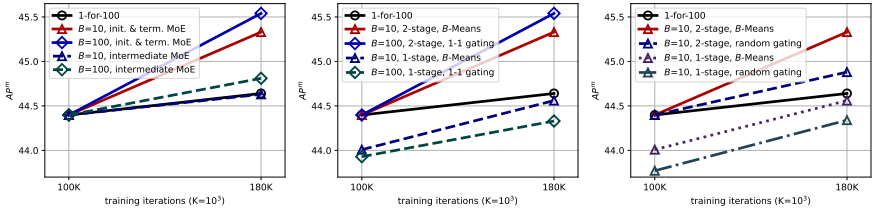
The adaptation performance and computational cost of the proposed method on different detector architectures are compared in Table 2b. We compare with the case when we use a single generic model to adapt to all 100 scenes (*1-for-100*). We calculate the difference between detection AP^m values of an adapted model with this model, which we refer to as APD^m. We also compare the adaptation performance of using 100 scene-specific *1-for-1* models to adapt to each scene, which can be executed when computational resources are unlimited. For a fair comparison, we also apply the two-stage training schedule for 1-for-1 models. Since different architectures have very different base model performances (Table 1), the comparison is only meaningful among the models adapted from the same base model.

We again verify that the proposed MoE models do not incur additional neural computation. They have the same GFLOPs and nearly identical inference latency as the corresponding base model, regardless of B. The gating module introduces minimal overhead. Since only a small portion of the network parameters are enhanced, the MoE models are only moderately larger than the base models, resulting in a modest increase in memory consumption at inference time. Though scene-specific 1-for-1 models do not introduce additional computational cost either, for the inference engine to serve all the scenes simultaneously, it needs to host all 100 copies of the network in memory, which is impractical.

When comparing the effectiveness of adaptation, our proposed framework can produce an MoE model with precision similar to or even higher than the scene-specific 1-for-1 models. For Faster-RCNN and DINO-5scale, the proposed model outperforms 1-for-1 models. This suggests that larger models are more prone to overfitting, so sharing certain network parameters among different scenes can be beneficial. In Figure 3, we visually demonstrate that the proposed method can obtain models that have increased detection precision over the base model while maintaining the computation cost. We can further reduce the latency by using downscaled input images for the adapted MoE models. Consequently, the models can still obtain improved or similar detection precision over the base models with reduced latency.

4.3 Ablation Study

The proposed method converts initial and terminal layers to MoE as described in §3.1, which we claim to be beneficial for scene-specific adaptation and is shown to outperform the single 1-for-100 model. Here, we instead use intermediate layers to enhance with MoE. For a fair comparison, we keep the number of parameters of the converted intermediate layer similar to



(a) MoE layers (b) Schedules (c) Gating strategies

Figure 4: Ablation study on different MoE layers, training schedules, and gating strategies. All models are based on Faster-RCNN with R-18 backbone, trained with the same batch size and learning rate. AP^m at training iterations 100,000 and 180,000 are shown. All two-stage trained models start from the warmed up 1-for-100 model at iteration 100,000. In (a) B=10 models use B-Means gating and B=100 models use 1-to-1 gating. In (b) and (c), all B=10 and B=100 models use proposed MoE layers enhancement. The proposed MoE layers, two-stage training schedule, and the B-Means gating strategy are all beneficial over other alternatives.

the proposed method, all models start from the same warmed up model, and all use the same gating strategy. The adaptation performance is compared in Figure 4a. It is clear that unlike the proposed method, using intermediate layers for MoE does not provide much performance gain over the non-MoE 1-for-100 model.

We also conduct experiments using the one-stage training schedule, where the MoE model is constructed, and the gating rules are determined from the base model without warm-up. These results are shown in Figure 4b. It is evident that if the model is branched directly from the base model without a warm-up phase, B=10 and B=100 models actually underperform compared to the generic 1-for-100 adaptation model, suggesting overfitting. Two-stage training results in improved performance with the same number of training samples.

We further compare different gating strategies of B=10 MoE models. One utilizes the proposed B-Means gating, and the other adopts a random gating strategy where an MoE branch is randomly assigned to each scene. The results are presented in Figure 4c. Only the B-Means approach achieves an AP^m comparable to that of the B=100 models (as reported in Table 2b), demonstrating the effectiveness of grouping similar scenes into the same branch. Even for one-stage trained models, B-Means is beneficial over random gating.

5 Summary

We have presented a novel framework that adapts a base object detector to video streams from various scene cameras without affecting inference speed or system throughput. By incorporating a mixture-of-experts structure into the base network's architecture, we have achieved an enhanced network that preserves inference latency and memory usage. This architecture can be trained with pseudo labels generated by the base detector itself, enabling self-supervised learning and eliminating the need for human supervision during the adaptation process. Additionally, using downscaled input images with the mixture-of-experts model can improve precision and reduce latency, thus optimizing both critical objectives simultaneously. Our approach has been tested across various state-of-the-art detection network architectures, outperforming the baselines in detection precision, memory consumption, inference latency, and data efficiency.

Acknowledgment. This research was partially supported by NSDF DUE-2055406.

References

[1] Sidra Aleem, Julia Dietlmeier, Eric Arazo, and Suzanne Little. Convlora and adabn based domain adaptation via self-training. In *IEEE International Symposium on Biomedical Imaging*, 2024.

[2] Tara Baldacchino, Elizabeth J. Cross, Keith Worden, and Jennifer Rowson. Variational bayesian mixture of experts models and sensitivity analysis for nonlinear dynamical systems. *Mechanical Systems and Signal Processing*, 66-67:178–200, 2016.

[3] Chenglong Bao, Yi Wu, Haibin Ling, and Hui Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[4] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *International Conference on Computer Vision*, 2019.

[5] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *ArXiv*, 2020.

[6] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. In *ArXiv*, 2020.

[7] 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*, 2020.

[8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, 2020.

[9] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *International Conference on Computer Vision*, 2021.

[10] Meilin Chen, Weijie Chen, Shicai Yang, Jie Song, Xinchao Wang, Lei Zhang, Yunfeng Yan, Donglian Qi, Yueting Zhuang, Di Xie, et al. Learning domain adaptive object detection with probabilistic teacher. In *International Conference on Machine Learning*, 2022.

[11] Tianlong Chen, Xuxi Chen, Xianzhi Du, Abdullah Rashwan, Fan Yang, Huizhong Chen, Zhangyang Wang, and Yeqing Li. Adamv-moe: Adaptive multi-task vision mixture-of-experts. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[12] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *ArXiv*, 2020.

[13] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[14] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *International Conference on Computer Vision*, 2021.

[15] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[16] Kyunghyun Cho and Yoshua Bengio. Exponentially increasing the capacity-to-computation ratio for conditional computation in deep learning. In *ArXiv*, 2014.

[17] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[18] KServe Developers. Kserve. https://kserve.github.io/website, 2023.

[19] PaddlePaddle Developers. Paddle serving. https://github.com/PaddlePaddle/Serving, 2022.

[20] TorchServe Developers. Torchserve. https://pytorch.org/serve, 2023.

[21] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. In *International Conference on Learning Representations*, 2013.

[22] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: a survey. *Journal of Machine Learning Research*, 20(1):1997–2017, jan 2019.

[23] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[24] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(1), Jan 2022.

[25] M.A. Ganaie, Minghui Hu, A.K. Malik, M. Tanveer, and P.N. Suganthan. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115: 105151, 2022.

[26] Andrea Camille Garcia, Jealine Eleanor Gorre, Joshua Angelo Karl Perez, and Mary Jane Samonte. Deep learning in smart video surveillance for crowd management: A systematic literature review. In *International Conference on Frontiers of Educational Technologies*, 2021.

[27] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *ArXiv*, 2021.

[28] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision*, 2015.

[29] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

[31] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[32] Zhenwei He and Lei Zhang. Multi-adversarial faster-rcnn for unrestricted object detection. In *International Conference on Computer Vision*, 2019.

[33] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, 2015.

[34] Minh Hoai and Andrew Zisserman. Talking heads: Detecting humans and recognizing their interactions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[35] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, 2017.

[36] 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. In *International Conference on Learning Representations*, 2022.

[37] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[38] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Model adaptation: Historical contrastive learning for unsupervised domain adaptation without source data. In *Advances in Neural Information Processing Systems*, 2021.

[39] Zehao Huang and Naiyan Wang. Data-driven sparse structure selection for deep neural networks. In *European Conference on Computer Vision*, 2018.

[40] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[41] Glenn Jocher. YOLOv5 by Ultralytics, May 2020. URL https://github.com/ultralytics/yolov5.

[42] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, January 2023. URL https://github.com/ultralytics/ultralytics.

[43] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *International Conference on Computer Vision*, 2019.

[44] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications. In *ArXiv*, 2022.

[45] Shuaifeng Li, Mao Ye, Xiatian Zhu, Lihua Zhou, and Lin Xiong. Source-free object detection by learning to overlook domain style. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[46] Wuyang Li, Xinyu Liu, and Yixuan Yuan. Sigma: Semantic-complete graph matching for domain adaptive object. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[47] 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 *AAAI Conference on Artificial Intelligence*, 2021.

[48] Yu-Jhe Li, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, Kan Chen, Bichen Wu, Zijian He, Kris Kitani, and Peter Vajda. Cross-domain adaptive teacher for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[49] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. In *ArXiv*, 2014.

[50] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *International Conference on Computer Vision*, 2017.

[51] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, 2016.

[52] Yen-Cheng Liu, Chih-Yao Ma, Zijian He, Chia-Wen Kuo, Kan Chen, Peizhao Zhang, Bichen Wu, Zsolt Kira, and Peter Vajda. Unbiased teacher for semi-supervised object detection. In *International Conference on Learning Representations*, 2021.

[53] Chris Maunder and David Cunningham. Codeproject. https://www.codeproject.com, 2023.

[54] Christoph Mayer, Martin Danelljan, Danda Pani Paudel, and Luc Van Gool. Learning target candidate association to keep track of what not to track. In *International Conference on Computer Vision*, 2021.

[55] Christoph Mayer, Martin Danelljan, Goutam Bhat, Matthieu Paul, Danda Pani Paudel, Fisher Yu, and Luc Van Gool. Transforming model prediction for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[56] Xue Mei and Haibin Ling. Robust visual tracking using l1 minimization. In *International Conference on Computer Vision*, 2009.

[57] Microsoft. Azure machine learning. https://azure.microsoft.com/en-us/products/machine-learning, 2023.

[58] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[59] Ammar Mohammed and Rania Kora. A comprehensive review on ensemble deep learning: Opportunities and challenges. *Journal of King Saud University - Computer and Information Sciences*, 35(2):757–774, 2023.

[60] Alhassan Mumuni and Fuseini Mumuni. Data augmentation: A comprehensive survey of modern approaches. *Array*, 16:100258, 2022.

[61] Muhammad Akhtar Munir, Muhammad Haris Khan, M. Saquib Sarfraz, and Mohsen Ali. Ssal: Synergizing between self-training and adversarial learning for domain adaptive object detection. In *Advances in Neural Information Processing Systems*, 2021.

[62] Yanjinlkham Myagmar-Ochir and Wooseong Kim. A survey of video surveillance systems in smart city. *Electronics*, 12(17), 2023.

[63] Supreeth Narasimhaswamy, Zhengwei Wei, Yang Wang, Justin Zhang, and Minh Hoai. Contextual attention for hand detection in the wild. In *International Conference on Computer Vision*, 2019.

[64] Supreeth Narasimhaswamy, Trung Nguyen, and Minh Hoai. Detecting hands and recognizing physical contact in the wild. In *Advances in Neural Information Processing Systems*, 2020.

[65] Supreeth Narasimhaswamy, Thanh Nguyen, Mingzhen Huang, and Minh Hoai. Whose hands are these? hand detection and hand-body association in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[66] Thanh Nguyen, Chau Pham, Khoi Nguyen, and Minh Hoai. Few-shot object counting and detection. In *European Conference on Computer Vision*, 2022.

[67] NVIDIA Corporation. Triton Inference Server: An Optimized Cloud and Edge Inferencing Solution., 2022. URL https://github.com/triton-inference-server/server.

[68] Adeshina Sirajdin Olagoke, Haidi Ibrahim, and Soo Siang Teoh. Literature survey on multi-camera system and its application. *IEEE Access*, 8:172892–172922, 2020.

[69] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *ArXiv*, 2017.

[70] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *ArXiv*, 2016.

[71] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. In *ArXiv*, 2018.

[72] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *ArXiv*, 2015.

[73] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015.

[74] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[75] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Strong-weak distribution alignment for adaptive object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[76] Laura Sevilla-Lara and Erik G. Learned-Miller. Distribution fields for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[77] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[78] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection. In *ArXiv*, 2020.

[79] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 2019.

[80] Chittesh Thavamani, Mengtian Li, Nicolas Cebron, and Deva Ramanan. Fovea: Foveated image magnification for autonomous navigation. In *International Conference on Computer Vision*, 2021.

[81] Chittesh Thavamani, Mengtian Li, Francesco Ferroni, and Deva Ramanan. Learning to zoom and unzoom. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2023.

[82] Vidit Vidit, Martin Engilberge, and Mathieu Salzmann. Learning transformations to reduce the geometric shift in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2023.

[83] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *ArXiv*, 2022.

[84] Lemeng Wu, Mengchen Liu, Yinpeng Chen, Dongdong Chen, Xiyang Dai, and Lu Yuan. Residual mixture of experts. *ArXiv*, 2022.

[85] Lin Xiong, Mao Ye, Dan Zhang, Yan Gan, Xue Li, and Yingying Zhu. Source data-free domain adaptation of object detector through domain-specific perturbation. *International Journal of Intelligent Systems*, 36:3746 – 3766, 2021.

[86] Yunqiu Xu, Yifan Sun, Zongxin Yang, Jiaxu Miao, and Yi Yang. H^2FA R-CNN: Holistic and hierarchical feature alignment for cross-domain weakly supervised object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[87] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[88] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. In *International Conference on Learning Representations*, 2023.

[89] Zekun Zhang and Minh Hoai. Object detection with self-supervised scene adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023.

[90] Liang Zhao and Limin Wang. Task-specific inconsistency alignment for domain adaptive object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[91] Zijing Zhao, Sitong Wei, Qingchao Chen, Dehui Li, Yifan Yang, Yuxin Peng, and Yang Liu. Masked retraining teacher-student framework for domain adaptive object detection. In *International Conference on Computer Vision*, 2023.

[92] Zihan Zhong, Zhiqiang Tang, Tong He, Haoyang Fang, and Chun Yuan. Convolution meets loRA: Parameter efficient finetuning for segment anything model. In *International Conference on Learning Representations*, 2024.

[93] Yitao Zhu, Zhenrong Shen, Zihao Zhao, Sheng Wang, Xin Wang, Xiangyu Zhao, Dinggang Shen, and Qian Wang. Melo: Low-rank adaptation is better than fine-tuning for medical image diagnosis. *ArXiv*, 2023.