Robust and Accurate Object Detection via Adversarial Learning

Xiangning Chen[1,2*] Cihang Xie[3] Mingxing Tan[1] Li Zhang[1] Cho-Jui Hsieh[2] Boqing Gong[1]

[1]Google [2]UCLA [3]UCSC

Abstract

Data augmentation has become a de facto component for training high-performance deep image classifiers, but its potential is under-explored for object detection. Noting that most state-of-the-art object detectors benefit from fine-tuning a pre-trained classifier, we first study how the classifiers' gains from various data augmentations transfer to object detection. The results are discouraging; the gains diminish after fine-tuning in terms of either accuracy or robustness. This work instead augments the fine-tuning stage for object detectors by exploring adversarial examples, which can be viewed as a model-dependent data augmentation. Our method dynamically selects the stronger adversarial images sourced from a detector's classification and localization branches and evolves with the detector to ensure the augmentation policy stays current and relevant. This model-dependent augmentation generalizes to different object detectors better than AutoAugment, a model-agnostic augmentation policy searched based on one particular detector. Our approach boosts the performance of state-of-the-art EfficientDets by +1.1 mAP on the COCO object detection benchmark. It also improves the detectors' robustness against natural distortions by +3.8 mAP and against domain shift by +1.3 mAP.

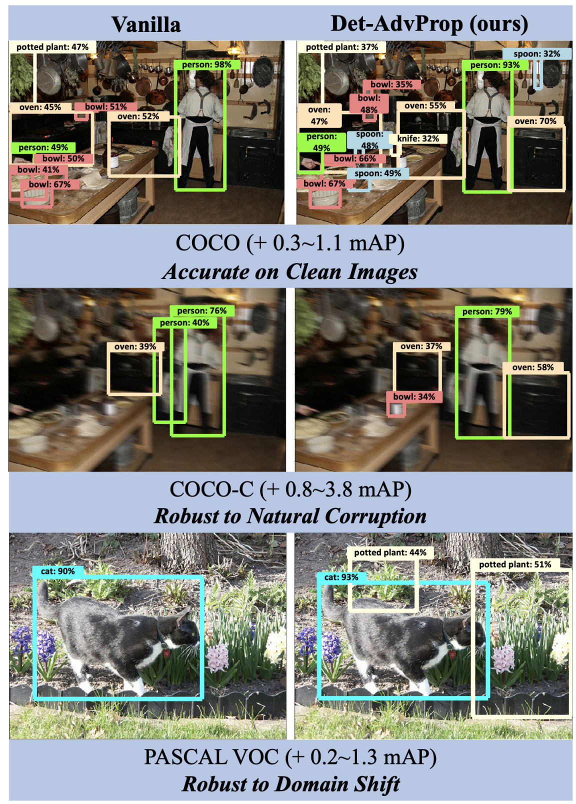


Figure 1. **Top:** Det-AdvProp improves object detectors' accuracy on clean images. Our model correctly detects some objects (e.g., "spoon" and "knife") missed by the vanilla detector trained without Det-AdvProp. **Middle:** Det-AdvProp improves the detectors' robustness against natural corruption. The vanilla detector misses "bowl" and "oven" and produces a false positive for "person" after the image is corrupted by motion blur. **Bottom:** Det-AdvProp improves robustness against cross-dataset domain shift. We can successfully detect the "potted plants" behind the "cat" from an image out of the dataset for training. (best viewed in color).

1. Introduction

Deep neural networks (DNNs) are powerful tools for visual representation learning. As the training data grows in size and diversity, DNNs keep up the pace and achieve unprecedented performance on a wide range of benchmarked tasks [13, 31, 30, 24, 12, 29]. The learned representations also demonstrate good transferability to downstream tasks for which there is often a small amount of curated data. This pre-training and then fine-tuning paradigm is one of the crucial enablers for state-of-the-art object detectors [32, 11, 30, 20, 22]. In this paper, we aim to enhance this learning paradigm for training not only accurate but also robust object detectors.

We first revisit the role of pre-training in object detection, given He *et al.*'s study [11] about vanilla ImageNet [5] pre-training and yet the new advances in data-augmented ImageNet pre-training [36, 34]. We examine both the accuracy and robustness of the detectors. In the study with the top-performing EfficientDet detectors [33], we find that the performance gains for ImageNet classification, brought

*Work done during an internship at Google.

by advanced data augmentation methods in pre-training, diminish after fine-tuning regardless of the detectors' accuracy or robustness. This observation motivates us instead to investigate fine-tuning, the second stage in the paradigm for training object detectors.

Our high-level idea is to reposition the recently developed data augmentations from the pre-training stage to fine-tuning. We first study AutoAugment [43] in the object detector fine-tuning because its policy is purposely searched for augmenting object detectors' training. However, experiments reveal that AutoAugment fails to provide consistent improvements to the detectors we studied, probably because it was searched based on only one object detector and one dataset, limiting its generalization ability.

In light of the lessons above, we switch to the model-dependent data augmentation of AdvProp [34] for fine-tuning object detectors. AdvProp uses adversarial examples to improve image classification models. It employs separate batch normalization layers for the clean training images and the adversarial examples to accommodate their distinct statistics. Unlike AutoAugment or many other augmentation methods, AdvProp can dynamically evolve with the primary model during training to ensure the augmentation is up to date.

We improve AdvProp to fit it into the object detection fine-tuning (denoted by Det-AdvProp). Previous works show that detectors benefit from shape cues [8], and adversarial examples help CNNs learn shape-related representations [40]. There are two sources to generate adversarial examples using a detector: its classification head and its localization head. We conduct a local comparison at each training iteration to identify the source that is more "adversarial" than the other, which is then selected to augment the training data. We show that this local comparison is crucial. Straightforwardly aggregating the two sources gives rise to weak adversarial examples because some of the adversarial gradients mutually conflict [38]. Another alternative, keeping both and separating them to different batch normalization layers, incurs a too strong regularization to the detector, leading to low accuracy, albeit high robustness.

We report the following main findings in this paper. Although the pre-training stage remains more effective for object detection than random initialization under a reasonable computing budget (we run the fine-tuning for up to 300 epochs on the COCO object detection dataset [21]), the performance gain at the pre-training-stage diminishes after fine-tuning, regardless of any strong data augmentations for the pre-trained backbone. Instead, we demonstrate that it is more promising to incorporate advanced data augmentations into fine-tuning. Our Det-AdvProp boosts state-of-the-art EfficientDets' accuracy by 0.3–1.1 mAP, robustness to natural corruption by 0.8–3.8 mAP, and robustness to domain shift by 0.2–1.3 mAP (illustrated in Figure 1). Finally, we see that our model-dependent Det-AdvProp substantially outperforms the model-agnostic AutoAugment in object detection under various settings.

2. Related Work

Data Augmentation. By applying label-preserving transformation to images, data augmentation has become a standard paradigm for training an image classifier [3, 4, 34, 37, 16, 19, 41, 18]. Most of them are model agnostic policy. In comparison, AdvProp [34] augments the training data by adversarial attack, its augmentation policy is unique to model and data. For the downstream object detection task, there are much fewer works specifically crafted for the fine-tuning process [42, 43]. One representative work - AutoAugement [43] searches based on RetinaNet [20] and a subset of COCO dataset [21], containing operations like rotation and bbox-only-translation.

Attacks and Adversarial Training for Object Detectors. Many effective attacks crafted for object detectors have been proposed recently. Most of them generate adversarial examples solely based on one individual loss (classification or localization loss) of the detection task [35, 2, 7, 25]. For instance, DAG [35] optimizes over a loss function that misleads the detectors to produce incorrect classification results. Some other works [17, 23] simultaneously attacks both the bounding box regression and classification to disable their predictions. To defend those attacks, Zhang *et al*. [38] extend adversarial training [26] to the scenario of object detection by leveraging the attacks sourced from both classification and localization domains. Our proposed Det-AdvProp can largely outperform their adversarial training in terms of both clean accuracy and robustness (Appendix B).

3. Motivation

Data augmentation is an effective way to improve image classification models [3, 37, 4, 36]. For example, AdvProp [34] uses adversarial images to boost the accuracy of EfficientNets [32] by up to 0.6% on ImageNet [5] accuracy, and Noisy Student [36] trained with noise and data as augmentation surpasses the vanilla models by over 1% on ImageNet. When it comes to object detection, which is arguably more complex than image classification, there are two natural choices to augment a detector: (i) Borrow an already augmented model from the upstream classification task and then fine-tune it for object detection. (ii) Directly augment the detector during training.

We first examine the first choice (i). We follow the common practice to fine-tune pre-trained ImageNet classifiers using the MS COCO object detection dataset [21]. Specifically, we initialize three sets of EfficientDets [33] (each set contains a small-scale detector D2 and a large-scale detector D5) by three ImageNet models, which are respectively

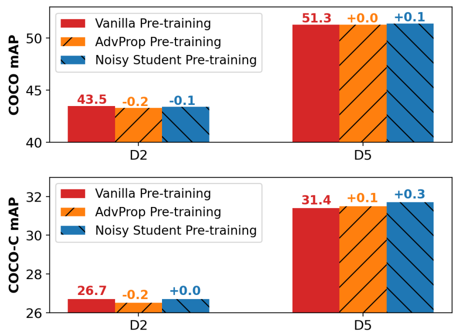


Figure 2. Results of EfficientDet-D2 and D5 on COCO and COCO-C with vanilla, AdvProp [34], and Noisy Student [36] pre-trained backbones. Fine-tuning substantially attenuates the effect of these data augmentation methods applied to pre-training.

pre-trained with no strong augmentation, AdvProp [34], and Noisy Student [36]. We fine-tune the networks up to 300 epochs to ensure they converge.

Figure 2 shows the comparison results on the validation sets of MS COCO [21] and COCO-C [27], respectively. The COCO-C dataset provides natural corruption to the COCO images aiming to test a detector's robustness to the real-world adversary. Although the checkpoints produced by AdvProp and Noisy Student surpass their vanilla counterparts (e.g., for B5, AdvProp +0.6% accuracy on ImageNet and -6.1% mean corruption error on ImageNet-C [14]), their advantages do not transfer to the object detection. The detectors built upon them perform similarly and sometimes even worse on COCO than the detector fine-tuned from the vanilla checkpoint. A similar observation holds on COCO-C. Our study is in the same vein as [11], but we examine the detectors' robustness in addition to their accuracy.

Since the fine-tuning attenuates the performance gains by strong data augmentation in the upstream classification task, we shift to the second option, augmenting during fine-tuning for harvesting data augmentation to improve a detector's accuracy and robustness. It is nontrivial to kill accuracy and robustness with one stone. Our first trial is with AutoAugment [43], which is searched based on the RetinaNet object detector [20] over COCO. However, it does not generalize well to other detectors (EfficientDets in this work), albeit still on the same COCO dataset (see Tables 1, 3, 5 for the results).

In light of these challenges, we develop a model-dependent augmentation for object detection via adversarial learning, given AdvProp's promising results [34]. Our method enables a detector to model adversarial images sourced from both object classification and localization, enforcing the detector to learn from its own weaknesses without the need for policy search as in AutoAugment [43].

4. Approach

This section describes our main approach to improving the object detectors' robustness and accuracy. We start by reviewing AdvProp for classification [34] and then tailor it for object detection.

4.1. AdvProp for Classification Revisited

Prior works that jointly train with clean and adversarial examples meet with performance degradation on clean images despite the improvement of robustness (e.g., against adversarial attacks) [26, 15, 39]. Most recently, Xie *et al*. manage to leverage the adversarial examples to improve image classifiers' accuracy, besides robustness, by a new training paradigm named Adversarial Propagation (AdvProp) [34]. Intuitively, they argue that the clean and adversarial images are drawn from distinct distributions, making it suboptimal to share the same statistical estimation in batch normalization (batchnorm) layers. To disentangle the two underlying distributions, AdvProp introduces auxiliary running mean and running variance for the adversarial images, leaving the main batchnorm layers to serve solely for the clean images. In particular, in every epoch, they first generate adversarial images based on the auxiliary batchnorm. Then they forward the clean and adversarial mini-batches into the network, each with their exclusive batchnorm, followed by standard backpropagation to optimize the total loss.

Under such a training scheme, the separate batchnorms take care of the distribution shift between clean and adversarial examples, jointly giving rise to higher performance than learning only with the clean images. Another benefit is robustness. The models trained by AdvProp is more robust against image distortions, probably because they are immune to the adversarially distorted examples by training. Inspired by their success in image classification, we propose Det-AdvProp to build accurate and robust detectors as object detection plays a crucial role in many real-world applications such as autonomous driving.

4.2. Det-AdvProp

One-stage object detectors take an image $\mathbf{x}$ as input and predict a set of objects $\{(\hat{\mathbf{y}}, \hat{\mathbf{b}})\}$, each includes a vector $\hat{\mathbf{y}}$ representing the probabilities over all possible classes and 4-dimensional bounding box coordinates $\hat{\mathbf{b}}$. During training, the objective function is usually formulated as follows:

$$\min_{\theta} \mathbb{E}_{\mathbf{x}\sim\mathcal{D};y,\mathbf{b}\sim\mathcal{B}(\mathbf{x})} L_{det}(\mathbf{x}, y, \mathbf{b}; \theta), \tag{1}$$

$$L_{det}(\mathbf{x}, y, \mathbf{b}; \theta) = L_{cls}(\mathbf{x}, y; \theta) + w \cdot L_{loc}(\mathbf{x}, \mathbf{b}; \theta), \tag{2}$$

where $\mathcal{D}$ is the set of training images, $\mathcal{B}(\mathbf{x})$ collects all the class y and bounding box $\mathbf{b}$ labels of the objects in image $\mathbf{x}$, θ is the model parameter, and w is the weight to balance the classification loss L_{cls} (e.g., focal loss [20]) and the localization loss L_{loc} (e.g., Huber loss).

Equation (2) essentially defines a multi-task learning problem, so possible attacks on object detectors can emerge from two heterogeneous domains for classification and localization, respectively. In some sense, it is easier to attack object detectors than image classifiers. Many existing methods have successfully fooled popular object detectors by attacking an individual loss in equation (2) [35, 2, 7].

It is a double-edged sword to have the adversarial examples of heterogeneous domains for object detectors. On the one hand, they may be viewed as versatile augmentations to the training data, potentially improving the detectors' accuracy on clean test data. On the other hand, the detectors need to be enhanced against both domains of adversarial examples to achieve robustness.

AdvProp over L_{det}. Due to the multi-task nature of the detectors, a straightforward application of AdvProp to object detection, i.e., by attacking the total loss L_{det} and dedicating two batchnorms, does not work well (see Table 8). Its performance is about the same or worse than attacking the single-task classification loss L_{cls}. We can further understand this observation following [38], which finds that the two gradients, $\nabla_{\mathbf{x}} L_{cls}$ and $\nabla_{\mathbf{x}} L_{loc}$ which are generated when we attack the overall detection loss L_{det}, have distinct value ranges and inconsistent directions. They reduce the augmentation effect of the resultant adversarial examples and even mutually cancel out.

AdvProp over L_{cls} and L_{loc}. Another plausible approach is to model the clean images and the attacks sourced from classification and localization as three distinct domains. To disentangle them, we use three batchnorms. One reserved for clean images, another accounting for the adversarial examples generated by attacking the classification loss, and the last for the localization-sourced adversarial examples. However, this method hurts the detectors' performance on clean images though it yields the highest robustness on corrupted images (see Table 8). We conjecture the two auxiliary batchnorms are overly strong regularization during training, leading to the detector's under-fitting to the clean training set.

Det-AdvProp. Learning from the two lessons above, we develop the following training scheme named Det-AdvProp. Similar to the AdvProp over L_{det}, we still use one, not two, auxiliary batchnorm to take account of the adversarial examples. Similar to the AdvProp over L_{cls} and L_{loc}, we generate two adversarial examples for each input by attacking the two losses separately. However, we keep only one of them to avoid the potential conflict between the adversarial gradients of the classification loss and the localization loss by a max-max rule first proposed in [38].

Algorithm 1: Det-AdvProp

Input: Object detection dataset $\mathcal{D}$
Output: Learned network parameter θ
for each training epoch **do**
  Sample a random batch $\{\mathbf{x}^i, \{y^i, \mathbf{b}^i\}\} \sim \mathcal{D}$
  Generate $\hat{\mathbf{x}}^i_{cls}$ based on $L_{cls}(\mathbf{x}^i, y^i)$ using auxiliary batchnorm
  Generate $\hat{\mathbf{x}}^i_{loc}$ based on $L_{loc}(\mathbf{x}^i, \mathbf{b}^i)$ using auxiliary batchnorm
  Select $\hat{\mathbf{x}}^i$ based on Eq (5)
  Compute $L_{det}(\mathbf{x}^i, \{y^i, \mathbf{b}^i\})$ with main batchnorm
  Compute $L_{det}(\hat{\mathbf{x}}^i, \{y^i, \mathbf{b}^i\})$ with auxiliary batchnorm
  Perform a step of gradient descent w.r.t. θ $\min\ L_{det}(\mathbf{x}^i, y^i, \mathbf{b}^i) + L_{det}(\hat{\mathbf{x}}^i, y^i, \mathbf{b}^i)$
end for

More concretely, we use the FGSM algorithm [10] with non-targeted attack to explain Det-AdvProp without loss of generality. Given an input image $\mathbf{x}$ and a bounding box $\mathbf{b}$ over an object of class y, FGSM produces the adversarial examples by a one-step projected gradient descent:

$$\hat{\mathbf{x}}_{cls} = \mathcal{P}(\mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} L_{cls}(\mathbf{x}, y; \theta))), \tag{3}$$

$$\hat{\mathbf{x}}_{loc} = \mathcal{P}(\mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} L_{loc}(\mathbf{x}, \mathbf{b}; \theta))), \tag{4}$$

$$\hat{\mathbf{x}} = \underset{\mathbf{x} \in \{\hat{\mathbf{x}}_{cls}, \hat{\mathbf{x}}_{loc}\}}{\arg\max}\ L_{det}(\mathbf{x}, y, \mathbf{b}; \theta), \tag{5}$$

where ϵ is the attack strength and $\mathcal{P}$ denotes the projection onto the norm ball $\{\hat{\mathbf{x}} \mid \|\hat{\mathbf{x}} - \mathbf{x}\|_\infty \leq \epsilon\}$. We first generate two adversarial examples by maximizing the single task loss and then choose the one that maximizes the total loss of the detection task. Equation (5) is the max-max rule to keep the adversarial example (out of two) that maximizes the total detection loss L_{det}. The inner "max" in the max-max rule refers to the adversarial examples that maximize L_{cls} and L_{loc}, respectively, and the outer "max" indicates we take the one that maximizes the total detection loss L_{det}. The overall training objective is given below:

$$\min_\theta \mathbb{E}_{\mathbf{x} \sim \mathcal{D}; y, \mathbf{b} \sim \mathcal{B}(\mathbf{x})} L_{det}(\mathbf{x}, y, \mathbf{b}; \theta) + L_{det}(\hat{\mathbf{x}}, y, \mathbf{b}; \theta). \tag{6}$$

Intuitively, this max-max scheme lets the stronger one survive between the two adversarial examples $\hat{\mathbf{x}}_{cls}$ and $\hat{\mathbf{x}}_{loc}$. By separately attacking the classification and localization branches, we avoid the gradient misalignment problem. Since we only produce one adversarial example per clean image, coupled with one auxiliary batchnorm, the training does not suffer from the excessive regularization problem mentioned earlier.

Algorithm 1 describes Det-AdvProp in detail. Note that we can safely discard the auxiliary batchnorm after training

Model	mAP	AP50	AP75	APl	APm	APs
EfficientDet-D0	34.3	52.4	36.6	53.8	40.0	13.1
+ AutoAugment	34.4 (+0.1)	52.8 (+0.4)	36.7 (+0.1)	53.1 (-0.7)	40.2 (+0.2)	**13.9 (+0.8)**
+ Det-AdvProp (ours)	**34.7 (+0.4)**	**52.9 (+0.5)**	**37.2 (+0.6)**	**54.1 (+0.3)**	**40.6 (+0.6)**	**13.9 (+0.8)**
EfficientDet-D1	40.2	58.8	42.8	57.7	45.9	**21.2**
+ AutoAugment	40.1 (-0.1)	**59.2 (+0.4)**	43.2 (+0.4)	57.9 (+0.2)	45.7 (-0.2)	19.9 (-1.2)
+ Det-AdvProp (ours)	**40.5 (+0.3)**	**59.2 (+0.4)**	**43.3 (+0.5)**	**58.8 (+1.1)**	**46.2 (+0.3)**	20.6 (-0.6)
EfficientDet-D2	43.5	62.5	47.1	60.9	48.6	23.7
+ AutoAugment	43.5 (+0.0)	**62.8 (+0.3)**	46.6 (-0.5)	59.8 (-1.1)	48.7 (+0.1)	23.9 (+0.2)
+ Det-AdvProp (ours)	**43.8 (+0.3)**	62.6 (+0.1)	**47.3 (+0.2)**	**61.0 (+0.1)**	**49.6 (+1.0)**	**25.6 (+1.9)**
EfficientDet-D3	46.8	65.3	50.6	62.8	51.6	29.8
+ AutoAugment	47.0 (+0.2)	66.0 (+0.7)	50.8 (+0.2)	63.0 (+0.2)	51.7 (+0.1)	29.8 (+0.0)
+ Det-AdvProp (ours)	**47.6 (+0.8)**	**66.3 (+1.0)**	**51.4 (+0.8)**	**64.0 (+1.2)**	**52.2 (+0.6)**	**30.2 (+0.4)**
EfficientDet-D4	49.3	68.2	53.3	63.7	53.6	**33.0**
+ AutoAugment	49.5 (+0.2)	**68.7 (+0.5)**	53.7 (+0.4)	64.9 (+1.2)	54.0 (+0.4)	31.9 (-1.1)
+ Det-AdvProp (ours)	**49.8 (+0.5)**	68.6 (+0.4)	**54.2 (+0.9)**	**65.2 (+1.5)**	**54.2 (+0.6)**	32.4 (-0.6)
EfficientDet-D5	51.3	70.1	55.8	65.1	55.1	35.9
+ AutoAugment	51.5 (+0.2)	70.4 (+0.3)	56.0 (+0.2)	65.2 (+0.1)	56.1 (+1.0)	35.4 (-0.5)
+ Det-AdvProp (ours)	**51.8 (+0.5)**	**70.7 (+0.6)**	**56.3 (+0.5)**	**66.1 (+1.0)**	**56.2 (+1.1)**	**36.2 (+0.3)**

Table 1. Comparison of vanilla training, AutoAugment [43], and Det-AdvProp on MS COCO [21]. Our proposed Det-AdvProp consistently outperforms vanilla training for different detectors, and it performs better than AutoAugment on all object sizes.

and use the main batchnorm for inference. The object detectors trained by Det-AdvProp have the same parameters and latency as those obtained by vanilla training.

5. Experiments

5.1. Setups

We select EfficientDet [33] of various scales as the default object detectors, including the lightweight EfficientDet-D0 model with fewer than 4M parameters and the large-scale EfficientDet-D5 detector that achieves over 50 mAP on COCO [21]. We train the detectors using the COCO 2017 object detection dataset [21] for 300 epochs and evaluate them on COCO's validation set to obtain the so called clean accuracy. We also test the detectors' robustness to natural corruptions on the COCO-C dataset [27], including 15 types of corruption each with 5 severity levels. Finally, we apply the detectors to the PASCAL VOC 2012 dataset [6] to evaluate robustness under domain shift. Please refer to Appendix D for details and training complexity.

5.2. Det-AdvProp Improves Accuracy

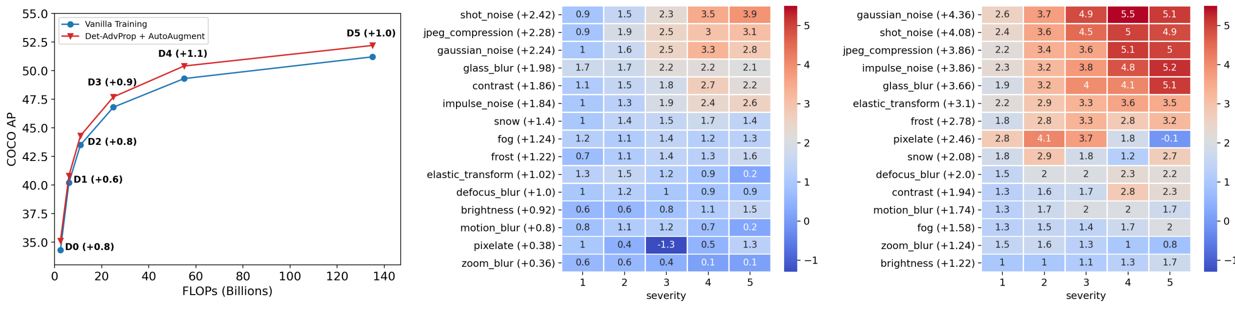
Results on COCO. We train the EfficientDet detectors with Det-AdvProp and compare them with the models out of vanilla training and the AutoAugment searched for object detection [43] in Table 1. As different scales of detectors favor different attack manners (targeted vs. non-targeted), which we ablate next, we first report the best clean mAP here. See Appendix D for the corresponding attack manners. Compared with the vanilla training baseline, our method can consistently increase the mAP score over all the detectors of various scales. The improvement mainly comes from large and medium-sized objects. The performance gain is especially notable for large-scale detectors with high capacities. Det-AdvProp improves D0–D2 by at most 0.4 mAP, but boosts D3—D5 by at least 0.5 mAP. On the contrary, AutoAugment cannot transfer well to the EfficientDets. It even incurs performance drop for some models (e.g., -0.1 mAP on D1). Det-AdvProp outperforms AutoAugment not only for the overall results but also on all the sizes of objects.

To push the limit of our method, we further combine the two strategies by employing AutoAugment to fine-tune the detectors obtained by Det-AdvProp. According to Figure 3a, the combined augmentation leads to even more accurate detectors. Similar to the previous observations, the performance gain is proportional to the model capacity. Detectors larger than D3 all increase by at least 1.0 mAP. With the combined augmentation strategy, D4 can achieve 50.4 mAP with just 21M parameters. As a comparison, AmoebaNet accompanied by NAS-FPN [9] and AutoAugment [43] achieves 50.7 mAP but has 209M parameters, which is 10x larger than D4. The analysis in Appendix C further shows that we can achieve performance gain on most classes and reduce various types of detection errors.

Model	D0	D1	D2	D3	D4	D5
Vanilla	33.4	36.6	38.2	41.7	44.6	46.5
Det-AdvProp	**33.9 (+0.5)**	**37.9 (+1.3)**	**40.0 (+1.8)**	**43.8 (+2.1)**	**46.2 (+1.6)**	**48.2 (+1.7)**

Table 2. Performance comparison between Det-AdvProp and the vanilla baseline without ImageNet pre-training. Det-AdvProp gives rise to performance gains for all detectors.

Without Pre-training. In some real-world settings, the ImageNet pre-trained backbone is not always accessible due to various reasons. Previous work also shows that training object detectors from scratch can match the performance with pre-training given sufficiently many training iterations [11]. Hence, we also report the improvement of our method over the vanilla training baseline when EfficientDets are initialized from scratch. The other training settings are exactly the same as before. As illustrated in Table 2,



(a) (b) (c)

Figure 3. **Left:** Combining Det-AdvProp with AutoAugment [43] can produce slightly better results than using Det-AdvProp alone. EfficientDet-D4 achieves over 50 mAP with 21M parameters, which is 10x less than AmoebaNet+NAS with NAS-FPN [9] and AutoAugment [43]. **Middle:** Performance gains of AutoAugment over vanilla training on COCO-C [27]. **Right:** Performance gains of Det-AdvProp over vanilla training on COCO-C. The largest improvement is observed when the images are distorted by random noise and with strong corruption strength. (best viewed in color).

the proposed Det-AdvProp's effectiveness is magnified in this scenario. The augmented models can be over 2.0 mAP better than those trained via the vanilla baseline.

Single-class object detection. In certain applications, there is only one object class of interest. They desire a detector to localize that object out of the background, such as face detection, pedestrian detection, etc. To simulate this scenario, we choose three classes spanning different object sizes and numbers of training instances from the COCO dataset to test Det-AdvProp under the single-class object detection setting. We follow exactly the same experiment settings as before and always use the attack strength of $\epsilon = 1$. Table 3 shows the results of EfficientDet-D3, where AutoAugment again degrades the mAP for every class. Its policy is searched based on 80 classes and fails to adapt well to the single-class object detection. In contrast, Det-AdvProp enhances the detector by automatically learning from its own weakness and thus achieves consistent improvement. The performance gain tends to be larger when there are fewer training images.

Class	Object Size	# Images	Vanilla	Auto-Augment	Det-AdvProp (ours)
Donut	Small	1,585	25.4	23.9 (-1.5)	**28.7 (+3.3)**
Person	Medium	66,808	58.2	58.0 (-0.2)	**58.5 (+0.3)**
Truck	Large	6,377	28.1	25.5 (-2.6)	**28.7 (+0.6)**

Table 3. mAP of vanilla, AutoAugment [43], and Det-AdvProp applied to EfficientDet-D3 under the single-class object detection setting. The AutoAugement strategy is searched based on 80 classes and fails to adapt to this setting. Det-AdvProp maintains its effectiveness in part due to its model-dependent nature.

5.3. Det-AdvProp Improves Robustness

Results on distorted COCO-C. We further evaluate the robustness of the detectors trained by Det-AdvProp under various corruptions on COCO-C [27]. Detecting objects from COCO-C is much more difficult than that from clean images. The popular Faster-RCNN [31] model with ResNet50 [13] backbone achieves 36.3 mAP on clean COCO images, while its mAP reduces to only 18.2 on COCO-C. As shown in Table 4, Det-AdvProp can achieve more significant improvement on distorted images than on the clean images over the baseline. For instance, Det-AdvProp can improve EfficientDet-D4 by 2.7 mAP on COCO-C, which is over 5x of the improvement on clean COCO. Compared with AutoAugment, we can consistently double its robustness improvement on various scales of EfficientDets. To further break down the improvement into different corruptions and severities, we visualize the performance gain achieved by AutoAugment [43] and the proposed Det-AdvProp in Figures 3b and 3c. Models trained by Det-AdvProp outperform the vanilla models on all 15 corruptions, and we observe the largest improvement when the images are distorted by random noise (e.g., +4.36 mAP under Gaussian noise and +4.08 mAP under shot noise). Another interesting finding is that the performance gains tend to become larger when the corruption strengths are stronger. Although AutoAugment can also help the models generalize to COCO-C, it is less effective than ours against every type of corruptions. The results of combined Det-AdvProp and AutoAugment is shown in Appendix A, where we achieve the largest improvement of +3.8 mAP.

Model	Vanilla	AutoAugment	Det-AdvProp (ours)
EfficientDet-D0	21.4	21.8 (+0.4)	**22.2 (+0.8)**
EfficientDet-D1	24.4	25.1 (+0.7)	**25.6 (+1.2)**
EfficientDet-D2	26.7	27.1 (+0.4)	**27.6 (+0.9)**
EfficientDet-D3	28.8	29.6 (+0.8)	**30.8 (+2.0)**
EfficientDet-D4	30.1	31.5 (+1.4)	**32.8 (+2.7)**
EfficientDet-D5	31.4	32.6 (+1.2)	**33.7 (+2.3)**

Table 4. Comparison of augmentation strategies and vanilla training on COCO-C [27]. Det-AdvProp can double the improvement achieved by AutoAugment [43].

Results on cross-dataset generalization. Another manifest of the model robustness is whether it can retain strong performance against domain shift. PASCAL VOC 2012 [6] only contains 20 classes, which are much smaller than the 80 labeled classes in COCO. The underlying distributions of the two datasets are also different in the image content or the bounding box sizes and locations. We use the trained detectors to run inference directly on the VOC dataset to test their transferibility. We maintain the COCO evaluation metrics in this experiment. According to Table 5, the Det-AdvProp trained detectors always outperform those by vanilla and AutoAugment training under every model scale and every evaluation metric. The models obtained by AutoAugment even substantially underperform the vanilla models. For instance, AutoAugment lowers D2's mAP by 0.6 compared to the vanilla training baseline. We also show the results of Det-AdvProp + AutoAugment in Appendix A, where the largest improvement is +1.3 mAP on EfficientDet-D5.

Model	mAP	AP50	AP75
EfficientDet-D0	55.6	77.6	61.4
+ AutoAugment	55.7 (+0.1)	77.7 (+0.1)	61.8 (+0.4)
+ Det-AdvProp (ours)	**55.9 (+0.3)**	**77.9 (+0.3)**	**62.0 (+0.6)**
EfficientDet-D1	60.8	82.0	66.7
+ AutoAugment	61.0 (+0.2)	82.2 (+0.2)	67.2 (+0.5)
+ Det-AdvProp (ours)	**61.2 (+0.4)**	**82.3 (+0.3)**	**67.4 (+0.7)**
EfficientDet-D2	63.3	83.6	69.3
+ AutoAugment	62.7 (-0.6)	83.3 (-0.3)	69.2 (-0.1)
+ Det-AdvProp (ours)	**63.5 (+0.2)**	**83.8 (+0.2)**	**69.7 (+0.4)**
EfficientDet-D3	65.7	85.3	71.8
+ AutoAugment	65.2 (-0.5)	85.1 (-0.2)	71.3 (-0.5)
+ Det-AdvProp (ours)	**66.2 (+0.5)**	**85.9 (+0.6)**	**72.5 (+0.7)**
EfficientDet-D4	67.0	86.0	73.0
+ AutoAugment	67.0 (+0.0)	86.3 (+0.3)	73.5 (+0.5)
+ Det-AdvProp (ours)	**67.5 (+0.5)**	**86.6 (+0.6)**	**74.0 (+1.0)**
EfficientDet-D5	67.4	86.9	73.8
+ AutoAugment	67.6 (+0.2)	87.2 (+0.3)	74.2 (+0.4)
+ Det-AdvProp (ours)	**68.2 (+0.8)**	**87.6 (+0.7)**	**74.7 (+0.9)**

Table 5. Results on PASCAL VOC 2012. The proposed Det-AdvProp gives the highest score on every model and metric. It largely outperforms AutoAugment [43] when facing domain shift.

5.4. Ablation Study

Det-AdvProp with targeted and non-targeted attacks. Targeted attack aims to fool a model to recognize an image incorrectly as a specified target label, while non-targeted attack is conducted by maximizing the training loss on the true label. We first carry out targeted attack with Det-AdvProp. As previous works report performance improvement if the object and background are treated differently [43], we consider the following two ways to generate random targets: (i) Randomly generate target labels for all the predefined anchors in the detector; (ii) Only perturb the ground-truth label for the anchors that cover objects, omitting those background anchors. We find that (ii) performs almost the same as vanilla training even when the attack strength is very large ($\epsilon = 5$). It is probably because the adversarial images which fool models to misidentify the background as objects contain valuable features, and the number of object-covering anchors is considerably smaller than the background anchors. Hence, for the targeted attack mentioned below, we refer to generating adversarial labels for all anchors if not specified otherwise. For the non-targeted attack, we simply maximize the training loss for both classification and localization branches and use the same attack strength ($\epsilon = 1$) as the targeted attack for a fair comparison.

The results are shown in Table 6. On clean COCO images, the adversarial examples obtained by targeted attack consistently improve the models' mAP, but non-targeted attack can hurt the performance of lightweight detectors (D0-D2), implying possibly too strong regularization. In contrast, when it comes to larger models (D3-D5), Det-AdvProp works better with the relatively stronger non-targeted attack. On the COCO-C dataset, both attack methods can improve the detectors' robustness, but the improvement achieved by non-targeted attack is much larger, and the gap is wider for the models of higher capacities. The relative robustness achieved by non-targeted attack is also the highest for all the detectors. Intuitively, Det-AdvProp with non-targeted attack learns from the worse-case adversarial examples than the targeted attack within the ϵ norm ball and therefore provides stronger regularization and robustness to the detectors.

Model	COCO mAP	COCO-C mAP	Relative rPC (%)
EfficientDet-D0	34.3	21.4	62.4
+ Det-AdvProp (TG)	**34.7 (+0.4)**	**22.2 (+0.8)**	64.0 (+1.6)
+ Det-AdvProp (NTG)	34.0 (-0.3)	22.1 (+0.7)	**65.0 (+2.6)**
EfficientDet-D1	40.2	24.4	60.7
+ Det-AdvProp (TG)	**40.5 (+0.3)**	25.6 (+1.2)	63.2 (+2.5)
+ Det-AdvProp (NTG)	40.1 (-0.1)	**26.1 (+1.7)**	**65.1 (+4.4)**
EfficientDet-D2	43.5	26.7	61.4
+ Det-AdvProp (TG)	**43.8 (+0.3)**	27.6 (+0.9)	63.0 (+1.6)
+ Det-AdvProp (NTG)	43.4 (-0.1)	**28.0 (+1.3)**	**64.5 (+3.1)**
EfficientDet-D3	46.8	28.8	61.5
+ Det-AdvProp (TG)	47.2 (+0.4)	30.1 (+1.3)	63.8 (+2.3)
+ Det-AdvProp (NTG)	**47.6 (+0.8)**	**30.8 (+2.0)**	**64.7 (+3.2)**
EfficientDet-D4	49.3	30.1	61.1
+ Det-AdvProp (TG)	49.6 (+0.3)	31.8 (+1.7)	64.1 (+3.0)
+ Det-AdvProp (NTG)	**49.8 (+0.5)**	**32.8 (+2.7)**	**65.9 (+4.8)**
EfficientDet-D5	51.3	31.4	61.2
+ Det-AdvProp (TG)	51.5 (+0.2)	32.4 (+1.0)	62.9 (+1.7)
+ Det-AdvProp (NTG)	**51.8 (+0.5)**	**33.7 (+2.3)**	**65.1 (+3.9)**

Table 6. Impact of targeted (TG) and non-targeted (NTG) attacks. All attacks are performed with strength $\epsilon = 1$. Det-AdvProp with non-targeted attack works better on large-scale detectors and can produce more robust models against distortions. rPC denotes the relative performance under corruption.

Det-AdvProp with different attack strengths. Here we ablate the effects of attack strengths represented by the radius ϵ of the perturbation norm ball. In Table 7, we vary ϵ from 1 to 3 and report the corresponding mAP scores on the COCO validation set. The attacker is set as targeted since the non-targeted attack decreases the mAP of lightweight models (D0-D2) even when ϵ is small as mentioned above. Aligning with the findings in image classification [34], large perturbation size degrades the performance of small models. EfficientDets D0-D2 work the best with $\epsilon = 1$, and stronger attack can cause performance degradation. On the contrary, EfficientDets D3-D5 work the best with relatively large perturbation $\epsilon = 2$. The clean performance of

EfficientDet-D5 is boosted by 0.2 mAP by increasing the ϵ from 1 to 2. It is reasonable to conclude that stronger attack strengths are desired for Det-AdvProp to better boost the detectors with higher capacities.

ϵ	D0	D1	D2	D3	D4	D5
0	34.3	40.2	43.5	46.8	49.3	51.3
1	**34.7**	**40.5**	**43.8**	**47.2**	49.6	51.5
2	34.2	40.0	43.5	**47.2**	**49.7**	**51.7**
3	34.1	40.0	43.4	47.1	49.5	51.6

Table 7. Impact of attack strengths. All attacks are conducted in the targeted manner. $\epsilon = 0$ means vanilla training. Larger attack strengths work better with larger model capacities.

Variants of Det-AdvProp. Different from image classifiers, object detectors need to identify objects by their class labels and box coordinates. They take the form of an inherent multi-task learning, preventing a direct application of AdvProp to the detectors. A straightforward idea would be generating adversarial examples by maximizing the total training losses L_{det}. Another approach is to generate two adversarial examples per clean image based on L_{cls} and L_{loc} separately, assuming the adversarial images sourced from classification and localization have distinct distributions. We call the last method **3BN** since it constructs three batchnorms during training. Besides, we also ablate the methods that attack the detectors based on an individual loss (either L_{cls} or L_{loc}). We choose non-targeted attack for this ablation study. According to the comparison in Table 8, we have the following observations:

- When attacking an individual task loss, choosing classification or localization does not make a big difference on clean COCO images. However, attacking L_{cls} performs much better than attacking L_{loc} on COCO-C, implying that the performance degradation caused by corruptions may mainly come from the classification branch. When attacking the total loss L_{det}, the resulting detectors' performance is in between, verifying that the adversarial gradients sourced from classification and localization may mutually conflict [38].
- **3BN** explicitly augments the detectors with both classification and localization branches, leading to the highest relative performance on corrupted images. However, the detectors fail to achieve high mAP on clean images probably because the adversarial features act as overly strong regularization. Indeed, the two auxiliary batchnorms in 3BN may dominate the optimization procedure, making the detectors under-fitting the clean training images.
- The proposed Det-AdvProp is the best method among all variants by consistently achieving the highest mAP on both COCO and COCO-C for the detectors of various scales. It uses only one auxiliary batchnorm during training to prevent excessive regularization and separately attacks the two branches to avoid the misalignment between adversarial examples.

Model	COCO mAP	COCO-C mAP	Relative rPC (%)
EfficientDet-D3	46.8	28.8	61.5
+ Det-AdvProp (**LOC**)	47.1 (+0.3)	30.0 (+1.2)	63.7 (+2.2)
+ Det-AdvProp (**CLS**)	47.2 (+0.4)	30.5 (+1.7)	64.6 (+3.1)
+ Det-AdvProp (**DET**)	47.1 (+0.3)	30.4 (+1.6)	64.5 (+3.0)
+ Det-AdvProp (**3BN**)	46.7 (-0.1)	30.6 (+1.8)	**65.5 (+4.0)**
+ Det-AdvProp	**47.6 (+0.8)**	**30.8 (+2.0)**	64.7 (+3.2)
EfficientDet-D4	49.3	30.1	61.1
+ Det-AdvProp (**LOC**)	49.6 (+0.3)	31.7 (+1.6)	63.9 (+2.8)
+ Det-AdvProp (**CLS**)	49.6 (+0.3)	32.6 (+2.5)	65.7 (+4.6)
+ Det-AdvProp (**DET**)	49.6 (+0.3)	32.7 (+2.6)	65.9 (+4.8)
+ Det-AdvProp (**3BN**)	49.2 (-0.1)	32.5 (+2.4)	**66.1 (+5.0)**
+ Det-AdvProp	**49.8 (+0.5)**	**32.8 (+2.8)**	65.9 (+4.8)
EfficientDet-D5	51.3	31.4	61.2
+ Det-AdvProp (**LOC**)	51.6 (+0.3)	33.1 (+1.7)	64.1 (+2.9)
+ Det-AdvProp (**CLS**)	51.7 (+0.4)	33.6 (+2.2)	65.0 (+3.8)
+ Det-AdvProp (**DET**)	51.6 (+0.3)	33.4 (+2.0)	64.7 (+3.5)
+ Det-AdvProp (**3BN**)	51.3 (+0.0)	33.5 (+2.1)	**65.3 (+4.1)**
+ Det-AdvProp	**51.8 (+0.5)**	**33.7 (+2.3)**	65.1 (+3.9)

Table 8. Comparison of several variants of Det-AdvProp. **LOC**, **CLS**, and **DET** generate the adversarial images based on L_{loc}, L_{cls}, and L_{det} respectively. **3BN** denotes the variant that generates two adversarial examples per clean image and employs three batchnorms during training. Det-AdvProp achieves the largest performance gains on both clean and corrupted images.

RetinaNet results. Apart from the state-of-the-art EfficientDets, we also test Det-AdvProp on the RetinaNet object detector [20] with a ResNet50 backbone [13]. We associate Det-AdvProp with non-targeted attack and the attack strength of $\epsilon = 1$. The other settings are the same as the baseline method. Det-AdvProp improves the mAP of RetinaNet from 35.6 to 36.1 on COCO and from 17.8 to 19.7 on COCO-C. We anticipate bigger improvements if we base RetinaNets on the backbones of higher capacities.

6. Conclusion

In this paper, we systematically examine the data augmentation strategies for object detectors. We discover that the performance gains on ImageNet classification including both accuracy and robustness, cannot be preserved after the object detection fine-tuning process. Instead, the proposed Det-AdvProp is specifically crafted for the fine-tuning process. Det-AdvProp dynamically learns from the stronger attack emerged from the classification and localization domains, which enables its policy to evolve during fine-tuning. This model-and-data-dependent manner is more effective than previous model-agnostic augmentation strategies. Extensive experiments show that our methods can consistently and substantially outperform the vanilla training and AutoAugment under various settings. The obtained detector is not only more accurate, but also more robust to image distortions and domain shift.

Acknowledgement: We would like to thank Yingwei Li for valuable discussions. CX is supported by a gift grant from Open Philanthropy. CJH is supported in part by Army Research Laboratory under agreement number W911NF-20-2-0158, and by NSF under IIS-2008173, IIS-2048280.

References

[1] Daniel Bolya, Sean Foley, James Hays, and Judy Hoffman. Tide: A general toolbox for identifying object detection errors. In *ECCV*, 2020.

[2] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. *Lecture Notes in Computer Science*, page 52–68, 2019.

[3] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Computer Vision and Pattern Recognition*, 2019.

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Practical automated data augmentation with a reduced search space. In *Advances in Neural Information Processing Systems*, 2020.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, 2009.

[6] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 2010.

[7] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Physical adversarial examples for object detectors. In *USENIX Conference on Offensive Technologies*, 2018.

[8] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[9] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. NAS-FPN: learning scalable feature pyramid architecture for object detection. In *Computer Vision and Pattern Recognition*, 2019.

[10] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[11] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *International Conference on Computer Vision*, 2019.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *International Conference on Computer Vision*, 2017.

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

[14] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[15] Alexey Kurakin, Ian Goodfellow, and S Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2017.

[16] Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation learning an optimal data augmentation strategy. *IEEE Access*, 2017.

[17] Yuezun Li, Xiao Bian, Ming-Ching Chang, and Siwei Lyu. Exploring the vulnerability of single shot module in object detectors via imperceptible background patches. *arXiv preprint arXiv:1809.05966*, 2018.

[18] Yingwei Li, Qihang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, and Cihang Xie. Shape-texture debiased neural network training. *arXiv preprint arXiv:2010.05981*, 2020.

[19] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In *Advances in Neural Information Processing Systems*, 2019.

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

[21] 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 *European Conference on Computer Vision*. 2014.

[22] 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.

[23] Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. Dpatch: An adversarial patch attack on object detectors. In *AAAI Workshop on Artificial Intelligence Safety*, 2019.

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2015.

[25] Jiajun Lu, Hussein Sibai, and Evan Fabry. Adversarial examples that fool detectors. *arXiv preprint arXiv:1712.02494*, 2017.

[26] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[27] 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.

[28] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. In *Computer Vision and Pattern Recognition*, 2018.

[29] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Computer Vision and Pattern Recognition*, 2018.

[30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Computer Vision and Pattern Recognition*, 2016.

[31] 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*, 2017.

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

[33] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. In *Computer Vision and Pattern Recognition*, 2020.

[34] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le. Adversarial examples improve image recognition. In *Computer Vision and Pattern Recognition*, 2020.

[35] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*, 2017.

[36] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Computer Vision and Pattern Recognition*, 2020.

[37] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[38] Haichao Zhang and Jianyu Wang. Towards adversarially robust object detection. In *International Conference on Computer Vision*, 2019.

[39] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

[40] Tianyuan Zhang and Zhanxing Zhu. Interpreting adversarially trained convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7502–7511. PMLR, 09–15 Jun 2019.

[41] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *International Conference on Learning Representations*, 2020.

[42] Zhi Zhang, Tong He, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of freebies for training object detection neural networks. *arXiv preprint arXiv:1902.04103*, 2019.

[43] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. *arXiv preprint arXiv:1906.11172*, 2019.