Security and Safety, Vol. 3, 2024002 (2024)
https://doi.org/10.1051/sands/2024002
sands.edpsciences.org

Security and Safety in Arti cial Intelligence

Intelligent Transportation

Robust object detection for autonomous driving based on semi-supervised learning

Wenwen Chen[1], Jun Yan[1;*], Weiquan Huang[1], Wancheng Ge[1], Huaping Liu[2], and Huilin Yin[1;*]

[1] College of Electronic and Information Engineering, Tongji University, Shanghai 201804, China
[2] School of Electrical Engineering and Computer Science, Oregon State University, Corvallis 97331-3211, USA

Received: 1 September 2023 / Revised: 17 January 2024 / Accepted: 4 February 2024 / Published online: 18 March 2024

Abstract Deep learning based on labeled data has brought massive success in computer vision, speech recognition, and natural language processing. Nevertheless, labeled data is just a drop in the ocean compared with unlabeled data. How can people utilize the unlabeled data e ectively? Research has focused on unsupervised and semi-supervised learning to solve such a problem. Some theoretical and empirical studies have proved that unlabeled data can help boost the generalization ability and robustness under adversarial attacks. However, current theoretical research on the relationship between robustness and unlabeled data limits its scope to toy datasets. Meanwhile, the visual models in autonomous driving need a signi cant improvement in robustness to guarantee security and safety. This paper proposes a semi-supervised learning framework for object detection in autonomous vehicles, improving the robustness with unlabeled data. Firstly, we build a baseline with the transfer learning of an unsupervised contrastive learning method|Momentum Contrast (MoCo). Secondly, we propose a semi-supervised co-training method to label the unlabeled data for retraining, which improves generalization on the autonomous driving dataset. Thirdly, we apply the unsupervised Bounding Box data augmentation (BBAug) method based on a search algorithm, which uses reinforcement learning to improve the robustness of object detection for autonomous driving. We present an empirical study on the KITTI dataset with diverse adversarial attack methods. Our proposed method realizes the state-of-the-art generalization and robustness under white-box attacks (DPatch and Contextual Patch) and black-box attacks (Gaussian noise, Rain, Fog, and so on). Our proposed method and empirical study show that using more unlabeled data bene ts the robustness of perception systems in autonomous driving.

Keywords Adversarial attack, robustness, autonomous driving, object detection, semi-supervised learning

Citation Chen W, Yan J, Huang W, et al. Robust object detection for autonomous driving based on semi-supervised learning. Security and Safety 2024;3: 2024002. https://doi.org/10.1051/sands/2024002

1 Introduction

Deep learning [1, 2] based on supervised learning has been widely used in computer vision tasks such as image classi cation [3{5], object detection [6{12], semantic segmentation [13{15]. In these tasks, deep convolutional neural networks [1, 16, 17] extract image features after the training process that utilizes backpropagation with su cient labeled data. However, adversarial samples [18] generated by adding some

* Corresponding authors (email: yanjun@tong.edu.cn (Jun Yan); yinhuilin@tongji.edu.cn (Huilin Yin))

noises that are imperceptible to the human eyes [19] can deceive such CNN models based on supervised deep learning. Such noises do not disturb human recognition, but they can easily fool DNN into making a wrong decision.

Supervised learning requires a large number of annotated data and is expensive. By comparison, the collection of unlabeled data is much simpler and more convenient. Besides, in most applications, more unlabeled data exist than labeled data. It would be a waste if deep neural networks could not utilize unlabeled data. Research has shown that the generalization and adversarial robustness are enhanced with the introduction of extra unlabeled data [20{23]. At the same time, unsupervised pre-trained models such as SimCLRv1 [24], SimCLRv2 [25], MoCov1 [26], MoCov2 [27] have also achieved great success which boosts the downstream visual tasks. In unsupervised and semi-supervised learning, a large number of unlabeled data can be exploited for the visual system to learn more representative features and make fewer mistakes.

In autonomous driving, safety and security is the number one priority. The limitations of AI models and human misuse may put the safety of the intended functionality (SOTIF) [28] at risk. The performance on the clean testing dataset is promising. However, improving the robustness of current object detection models under adversarial attacks, including white-box attacks and black-box image corruptions, with the simulation of adversarial weather is urgent. Although some theoretical research work has exploited the ability of semi-supervised learning to improve adversarial robustness on toy datasets such as CIFAR [29], MNIST [30], few studies apply semi-supervised learning to increase the security and safety in autonomous driving. In the autonomous driving eld, unlabeled data far exceeds labeled data. Semi-supervised learning makes the model more robust to various road and environmental conditions. Furthermore, semi-supervised learning can help models learn a broader range of feature representations with new annotated data. Autopilot systems may encounter a variety of noise in the real world, including visual occlusion and weather changes. Semi-supervised learning is usually more tolerant of such noise. Therefore, this work explores a new framework for the generalization ability and robustness of the 2D object detector in autonomous driving.

Our proposed framework leverages unlabeled data from nuScenes [31], mixed with labeled data from KITTI [32], to realize semi-supervised learning. Our work and contributions are listed as follows:

(1) *Semi-supervised method to improve generalization*: We utilize the pre-trained MoCo model, which uses contrastive learning to learn representative features. Then, we develop a semi-supervised co-training method, which uses unlabeled data to boost the generalization of object detection models.
(2) *BBAug to improve model robustness*: In the process of co-training, we apply an unsupervised data augmentation strategy of BBAug to improve the robustness under both white-box and black-box adversarial attacks.
(3) *Benchmarks of robustness on KITTI*: We conduct adversarial attacks on distinct detection models, namely Dpatch, Contextual Patch, and Image Corruptions. Empirical studies evaluate the corresponding robustness of the proposed method on the KITTI dataset.

This paper follows such an organization: Section 2 reviews the relevant works. Section 3 illustrates our proposed semi-supervised co-training methods. Training details and experiment results are presented in Section 4, followed by concluding remarks in Section 5.

2 Related works

2.1 Object detection

As one of the most basic and challenging problems in computer vision, object detection has attracted much attention in recent years [33]. Object detection aims to recognize and localize the objects in images. Over the years, with the explosion of deep learning, more and more object detection applications have been implemented in areas of autonomous driving, video security monitoring systems, and robotic vision.

Current state-of-the-art object detection models are built with deep convolution neural networks to extract features [10, 12, 16, 17]. Object detection can be divided into two categories: two-stage detection and one-stage detection [34]. The two-stage framework implements a coarse and ne process, while the one-stage framework completes detection in only one step. The genre of the RCNN algorithms [35] belongs

to two-stage detection. Its main idea is to generate a series of sparse proposal boxes through a heuristic method, namely Selective Search or a region proposal network, and then classify and regress them. Thus, the two-stage method has high accuracy but a relatively longer inference time. The main idea of the one-stage method, such as YOLO [8] and SSD [7], is to conduct intensive and uniform sampling at di erent image positions. The sampling utilizes di erent scales and aspect ratios, and then the features can be extracted by a CNN for direct classi cation and regression. The process requires only one step, reducing the time complexity compared with the two-stage detection models. However, a signi cant weakness of uniform intensive sampling is that it makes training harder, mainly because of the extreme imbalance between positive and negative samples, namely background [36], which a ects the accuracy of the object detection model.

2.2 Adversarial attack

Although deep neural networks have shown e ectiveness in computer vision tasks, Szegedy *et al.* [18] found that applying small unrecognizable perturbations to the image can confuse the classi er and cause it to make a wrong judgment, for instance, recognizing a panda as a gorilla. They believe that the non-linearity of deep neural networks leads to adversarial vulnerability. However, Goodfellow *et al.* [19] conjecture that linearity rather than non-linearity of the neural networks results in adversarial perturbations. They argue that even tiny perturbations can accumulate through a high-dimensional weight matrix and cause the value of the activation function to change signi cantly. Tanay *et al.* [37] argue that adversarial examples exist since there is a certain angle between the class boundary and the submanifold despite the close distance.

In autonomous driving, an arti cial attack on the detector could be fatal regarding vehicle safety and passengers' lives. Thus, it is meaningful to strengthen the visual models to avoid prediction errors even in an environment with perturbations.

Adversarial Attacks are grouped into two genres, black-box attacks and white-box attacks [38].

2.2.1 White-box attack

The white-box adversary would have prior knowledge of the model, such as architectures, training strategies, and parameters. Classic white-box attacks include FGSM [19], JSMA [39], C&W [40], DeepFool [41], ATNs [42] *etc*. However, these methods are limited in image classi cation and could be more e ective for object detection. In our experiments, two types of white-box attacks, Dpatch [43] and Contextual Adversarial Patch [44] are utilized. These two approaches are feasible to generate adversarial examples to fool the object detectors.

The Dpatch attack method applies in two scenarios: targeted attack and untargeted attack. In an untargeted attack, the objects in adversarial images can be predicted into arbitrary categories or simply be ignored by the detector. In a targeted attack, the detector would be deceived to recognize all the objects as the same speci c class.

Saha *et al.* [44] propose the contextual adversarial patch that would exploit the contextual information for the deceit of the detectors. The main idea of this approach is to make the detector \blind" to a speci c chosen class.

2.2.2 Black-box attack

Compared with the white-box attack, the adversary does not know the algorithms and parameters used by the deep learning model in the black-box attack. Our experiments use the image corruption method [45] that injects noise into the image. Hendrycks *et al.* [45] build a benchmark on ImageNet to evaluate the robustness of classi ers. Michaelis *et al.* [46] utilize Image Corruptions to generate benchmark datasets in the object detection task, including autonomous driving datasets. This package o ers a series of corruptions compromising 15 diverse corruption types and covers noise, blur, weather, and digital categories. Each of these 15 types of corruption has ve di erent severities, which can manifest in di erent intensities.

In particular, six types of image corruptions are utilized, namely Gaussian noise, impulse noise, shot noise, snow, fog, and frost, to test the robustness of the models under black-box attacks. The limitations of

the defense under such black-box attacks (abnormal weather and camera distortion) would cause the risk issue of the safety of the intended functionality (SOTIF) in autonomous driving. Our proposed method can improve the defense ability against these kinds of attacks to increase the safety of autonomous driving.

2.2.3 The focus of our work

Firstly, we transfer the adversarial examples generated by the Dpatch and Contextual Patch method to diverse object detecors [43, 44]. The experiment result shows the e ect of these adversarial patches on the models with known and unknown parameters.

Secondly, Image Corruptions are used to establish benchmarks on the KITTI dataset and evaluate various detection approaches' robustness to perturbations.

2.3 Unsupervised learning and semi-supervised learning

This subsection reviews the contrastive learning and semi-supervised learning methods. Contrastive learning is used in the pre-training phase as unsupervised learning. After ne-tuning in a supervised manner, co-training as semi-supervised learning is implemented to improve the robustness.

2.3.1 Contrastive learning

The key to contrastive learning is contrastive loss, which closes the gap between positive samples and pulls out the distance between the positive and negative samples. The method Momentum Contrast (MoCo) [26] utilizes the structure of a momentum encoder to realize contrastive learning and achieves state-of-the-art performance in downstream tasks such as linear classi cation and object detection.

MoCov2 [27] is an improved version of MoCov1 [26], which is based on SimCLR [24], another contrastive learning method. Unlike MoCo, SimCLR applies an \end-to-end" architecture to realize the comparison. That is, the negative samples are from the current batch and two encoders, both of which require backpropagation to be updated. The innovative work of SimCLR is to add a nonlinear projection head between the representation and contrastive loss, which improves the representation quality. In addition, the application of various data augmentation methods also boosts performance. MoCov2 adds the above two modi cations in the momentum encoder mechanism and outperforms SimCLR with a smaller batch size.

There exist other approaches to implement contrastive learning such as SwAV [47], which clusters the augmented versions instead of comparing the positive and negative samples and uses a mechanism to predict the code of a view from another view's representation, and BYOL [48], which does not utilize negative samples and trains an online network to predict the representation of target network using a di erent augmented version.

2.3.2 Semi-supervised learning

According to the research of Carmon$et\ al.$ [23], unlabeled data lls the gap of sample complexity between standard and robust classi cation. They add unlabeled data to the model trained on the CIFAR-10 dataset [29] and implement self-training, which outperforms the adversarial training [49, 50]. Alayrac$et\ al.$ [21] theoretically and empirically prove that using unlabeled data signi cantly improves robustness accuracy compared with the one that uses an equal number of labeled data. Naja$et\ al.$ [22] exploit the role of unlabeled data under a perturbed condition in theory and propose a framework that shows excellent performance in a classi cation task. As a consequence, semi-supervised learning is bene cial to adversarial robustness. Co-training is a semi-supervised learning and is utilized in our work to improve the robustness ability.

Co-training [51] is an improved version of self-training [52]. Blum$et\ al.$ [51] rst proposed co-training to classify web pages, in which two classi ers were trained from two di erent \views" separately to classify unlabeled data, which improved classi cation accuracy.

Apart from co-training, a type of data augmentation BBAug, is also used, which is especially e ective in unsupervised learning and object detection to enhance the ability to defend against adversarial attacks.

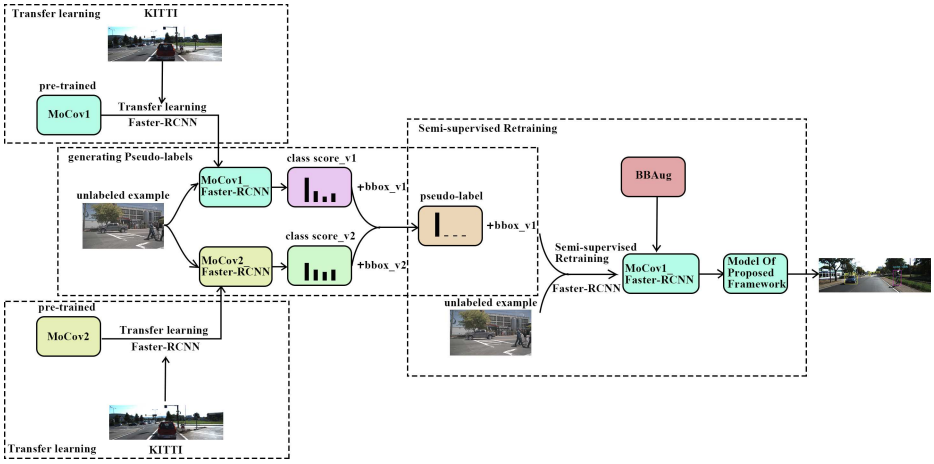

Figure 1. The structure of Semi-supervised Co-training: Firstly, Faster-RCNN based on MoCov1 and MoCov2, which are unsupervised learning methods during pre-training, are implemented on the KITTI dataset. Secondly, the two models pre-trained with MoCov1 and MoCov2 make predictions on the unlabeled data respectively. Only the prediction results of objects higher than a defined threshold would be retained. After that, the one with a lower classification score from two predictions is discarded and the retained prediction result is regarded as pseudo-label with the information of class and bounding box. Finally, the unlabeled data with pseudo-labels are added to the training dataset and used for semi-supervised learning, where BBAug is also used during the re-training process to boost the robustness

2.3.3 The focus of our work

Previous research on unsupervised and semi-supervised learning may not involve practical applications of autonomous driving safety or improving robustness. Our work transfers the model based on MoCo to the object detection task on the autonomous driving dataset KITTI. During semi-supervised learning, the unsupervised data augmentation method BBAug can help boost the robustness to abnormal weather conditions, alleviating the the safety of the intended functionality (SOTIF) risk.

3 Co-training for robust object detection

Figure 1 shows the structure of our training regime. This section introduces the details of our methodology. Our work consists of three parts: Transfer learning based on MoCo, Semi-supervised Co-training, and BBAug data augmentation.

3.1 Transfer learning based on MoCo

Momentum Contrast (MoCo) [26] is built on the framework of the dictionary lookup. The encoded augmented vector of an image is a query, and another augmented vector is a key. In each iteration, the queries match themselves to the keys using contrastive loss. Since previously encoded keys can be reused, this mechanism does not require a large GPU memory. Meanwhile, the size of negative samples is enlarged and can lead to a better representation of images. The contrastive loss function used in MoCo is InfoNCE [53]:

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}. \quad (1)$$

This loss is based on Noise Contrastive Estimation (NCE). k_+ is the positive key that queries q matches. k_i are negative keys. The variable τ is a temperature hyper-parameter. The keys and query are encoded vectors from augmented images in the mini-batch. The main objective of this loss is to make the two augmented versions from the same sample, namely k_+ and q, as similar as possible. The other augmented versions from di erent samples in a mini-batch, namely k_i, are forced to be dissimilar to q.

The models MoCov1 and MoCov2 that are pre-trained on ImageNet [54] are utilized, which both use Resnet-50 as the encoder. The di erences between them lie in the architecture and training strategy. The

mechanism of MoCov2 adds an MLP head between the encoder and contrastive loss. During the training process, extra blur augmentation is applied to the dataset, and a cosine learning rate schedule is utilized, in which the learning rate drops gradually to make the neural network get closer to the global minimum of the loss value.

The pre-trained representations and di erences are used to transfer to the object detection task. By using the pre-trained model trained on a large dataset for feature extraction, the training process can converge faster and avoid the \random initialization" training mode that a ects the performance of the object detection model in autonomous driving. It can obtain better generalization through knowledge transfer in a relatively e cient way. Furthermore, for the multi-layer CNN structure, the image features learned at shallower layers are more general, and those learned at deeper layers are more relevant to speci c tasks. Therefore, it would greatly help the downstream autonomous driving task to learn the \general features" using contrastive learning on large-scale image data.

The ne-tuned method is implemented on the architecture of Faster-RCNN.

In addition, in order to align with other object detection algorithms, the data augmentation is adjusted by adding random cropping and color distortion.

For the loss function, the original settings are kept, namely softmax cross-entropy loss for class loss and smooth L1 loss for box regression loss.

3.2 Semi-supervised co-training

Co-training is a method based on divergence. It assumes that all data can be classi ed from di erent perspectives or views and that di erent classi ers can be trained from di erent perspectives. These classi ers are used to classify unlabeled samples, and the unlabeled samples, considered con dent, are selected and added to the training dataset. Since these classi ers are trained from di erent views, they can complement each other and improve classi cation accuracy, just as things can be better understood from di erent points of view.

In our work, two models, MoCov1 and MoCov2, are utilized to predict the same dataset. Even for the same dataset, di erent network architectures can learn di erent prediction distributions [55, 56].

Models pre-trained with MoCov1 and MoCov2 are ne-tuned on KITTI using the Faster-RCNN method. For simplicity, the model pre-trained by MoCov1 is de ned as M_1 and pre-trained by MoCov2 as M_2. In this section, our implementation principles are also introduced, and the realization process will be explained in detail.

Suppose there is a labeled dataset $\mathcal{L}$ and an unlabeled dataset $\mathcal{U}$. The samples and annotations belonging to labeled dataset $\mathcal{L}$ are de ned as $\{x_i^l, y_i, b_i\}$, Where y_i denotes the corresponding class labels and b_i means ground-truth bounding boxes in ith image. The samples in the unlabeled dataset $\mathcal{U}$ are de ned as x_j^u.

In rst step, the M_1 and M_2 are ne-tuned on $\mathcal{L}$ respectively to obtain model F_1 and F_2. And then, we use F_1 and F_2 to predict the data in $\mathcal{U}$.

$$\begin{aligned}(\hat{y}_{1j}, \hat{b}_{1j}, s_{1j}) &= F_1(x_j^l; \Theta_1),\\ (\hat{y}_{2j}, \hat{b}_{2j}, s_{2j}) &= F_2(x_j^l; \Theta_2),\end{aligned} \tag{2}$$

where the $\hat{y}_j$ denotes the predicted class labels, $\hat{b}_j$ denotes the predicted bounding boxes and s_j is the class con dence for each detected objects in jth image. Θ_1 and Θ_2 signify that model F_1 and F_2 have di erent parameters.

Assuming that in jth image, F_1 has detected $\mathcal{M}_j$ objects and F_2 has detected $\mathcal{N}_j$ objects. After predicting, the objects whose classi cation score is lower than threshold $\mathcal{T}$ are ltered out.

$$\begin{aligned}(\hat{y}_{1j}^{y}, \hat{b}_{1j}^{y}, s_{1j}^{y}) &= \{(\hat{y}_{1j}^{m}, \hat{b}_{1j}^{m}, s_{1j}^{m}) | s_{1j}^{m} > \mathcal{T}, m \in \mathcal{M}_j\},\\ (\hat{y}_{2j}^{y}, \hat{b}_{2j}^{y}, s_{2j}^{y}) &= \{(\hat{y}_{2j}^{n}, \hat{b}_{2j}^{n}, s_{2j}^{n}) | s_{2j}^{n} > \mathcal{T}, n \in \mathcal{N}_j\},\end{aligned} \tag{3}$$

where the sign $\dagger$ indicates that the remaining prediction results after ltering.

In the next step, the two prediction results are integrated via only retaining the objects that F_1 and F_2 can detect, whose inference bounding boxes are intersected. In object detection, there usually appear a

few objects in one image. When the prediction results from two models for the same image are compared, the key issue is ensuring that the two compared objects are the same object. In other words, they should belong to the same class and be located in almost the same position. In order to address this issue, a criterion helps judge whether these objects predicted by two distinct models are the same or not. The two predictions could be assigned to the same object when the class labels are identical and the distance between the top-left coordinates is lower than a threshold $\mathcal{A}$. This criterion is loose but effective and practical. Since the unreliable results are filtered out, the predictions from F_1 and F_2 are accurate.

Suppose the number of remaining objects in $(\hat{y}_{1j}^{y}, \hat{b}_{1j}^{y}, s_{1j}^{y})$ is $\mathcal{P}_j$ and the number of remaining objects in $(\hat{y}_{2j}^{y}, \hat{b}_{2j}^{y}, s_{2j}^{y})$ is $\mathcal{Q}_j$, the equation description is shown as follow:

$$(\hat{y}_j, \hat{b}_j) = \begin{cases} \{(\hat{y}_{1j}^{yp}, \hat{b}_{1j}^{yp}) | p \in \mathcal{P}_j\} & \text{if } \hat{y}_{1j}^{yp} = \hat{y}_{2j}^{yq} \\ & \text{and } |\hat{b}_{1j}^{yp} - \hat{b}_{2j}^{yq}| < \mathcal{A} \\ & \text{and } s_{1j}^{yp} > s_{2j}^{yq}, \\ \{(\hat{y}_{2j}^{yq}, \hat{b}_{2j}^{yq}) | q \in \mathcal{Q}_j\} & \text{if } \hat{y}_{1j}^{yp} = \hat{y}_{2j}^{yq} \\ & \text{and } |\hat{b}_{1j}^{yp} - \hat{b}_{2j}^{yq}| < \mathcal{A} \\ & \text{and } s_{1j}^{yp} \leqslant s_{2j}^{yq}, \end{cases} \tag{4}$$

where the $(\hat{y}_j, \hat{b}_j)$ signifies the ultimate pseudo-class labels and bounding boxes after combining the results predicted by models F_1 and F_2. The final step is to add the selected unlabeled samples and their pseudo-labels to the labeled dataset to recompose a new labeled dataset $\mathcal{L}$.

$$\mathcal{L} = \{(x_i^l, y_i, b_i), (x_j^u, \hat{y}_j, \hat{b}_j)\}. \tag{5}$$

At last, a new model F is trained using F_1 or F_2. The overall algorithm procedure is depicted in Algorithm 1.

Algorithm 1 Co-training based on MoCov1 and MoCov2

Require: labeled dataset $\mathcal{L} = \{(x_i^l, y_i, b_i)\}_{i=1}^{I}$; unlabeled dataset $\mathcal{U} = \{x_j^u\}_{j=1}^{J}$; models F_1 and F_2 based on MoCov1 and MoCov2; filtering out threshold $\mathcal{T}$

1: Use models F_1 and F_2 to make predictions for unlabeled dataset $\mathcal{U}$.
2: Discard the predictions of objects whose classification scores are lower than $\mathcal{T}$.
3: Compare two predictions for the same object and retain the predictions whose classification score is larger.
4: Add pseudo-labels and the corresponding unlabeled data to the training dataset: $\mathcal{L} \leftarrow \mathcal{L}^* \cup \{(\hat{y}_j^*, \hat{b}_j^*)\}$.
5: return Train model F^* with F_1 or F_2

Output: new model F^*

3.3 BBAug data augmentation

Bounding Box Augmentation (BBAug) [57], which is a data augmentation strategy applied for unsupervised learning, is utilized to boost the generalization and robustness of the detector. BBAug is derived from AutoAugment [58], in which the optimal data augmentation scheme is automatically selected. This method searches for the best scheme by solving a discrete search problem using a search algorithm and search space. The basic idea is to use the reinforcement learning method to find the best image transformation strategy from the dataset. While AutoAugment is applied for the image classification task, the BBAug is specially designed for the object detection task, in which the bounding box transformation is consistent with the geometric operations.

In the procedure of BBAug, an augmentation policy is defined as a set of sub-policies. When the model is trained, one of the sub-policies is selected randomly to augment the image. Within each sub-policy, there are augmentations to be applied to the image in turn. Each transformation also has two hyperparameters: probability and magnitude. Probability indicates the likelihood that the augmentation will be applied, and magnitude indicates the extent of the augmentation.

BBAug is utilized as assistance during semi-supervised co-training with creating more unlabeled data to improve the robustness of the model against the black-box attack such as Gaussian noise, impulse, snow, frost, etc. The fundamental categories of data augmentation, such as Flipping, Color space, Cropping, etc., are applied in our method as well. Nevertheless, the basic augmentation approaches bring about merely a slight improvement in robustness, especially when the detector deals with images injected with noises. In contrast to the basic supervised data augmentations, BBAug has achieved a great improvement in defending the black-box attack. It contains color operations and geometric operations. The augmentations are more diverse and can help the model learn more interpretable features. Besides, more unlabeled data can be generated through BBAug, which also bene ts the robustness against black-box attacks.

4 Experiments

In this section, the datasets, training strategy, and metrics of the experiments will rst be introduced. Then, our experimental results are displayed, including the performance of classic object detection algorithms in a supervised manner and our method based on semi-supervised learning. Their robustness against white-box attacks, which invade the model with the prior knowledge or in the transfer settings, and black-box attacks, will then be compared. After that, the in uence of unlabeled data will be inspected, and di erent lter criteria are also chosen to determine which can achieve the best performance. Finally, ablation experiments are conducted to show the e ect of data augmentation, namely with the combination of Sharpness, Color, Contrast, Cutout, Shear,etc.

4.1 Experiment details

4.1.1 Labeled dataset

KITTI [32] dataset is jointly founded by Karlsruhe Institute of Technology and Toyota American Institute of Technology. It is the most authoritative benchmark dataset in the autonomous driving scenario in the world at present. There are 7481 training images for 2D object detection, which provides object detection benchmarks of various detection algorithms and networks and uses precision-recall curves for evaluation. The original labels in KITTI are subdivided into eight categories: Car, Van, Truck, Pedestrian, Person-sitting, Cyclist, Tram, and Misc. In order to keep aligned with the evaluation benchmark provided by KITTI, the categories Car, Van, Truck, and Tram are grouped and relabeled as \car", Pedestrian and Person-sitting are relabeled as \pedestrian" and the class Cyclist remained unchanged. In this way, eight classes are merged into three classes, and our following experiments are also based on the three classes.

4.1.2 Unlabeled dataset

nuScenes [31] is a round-the-clock autonomous driving dataset, whose data comes from driving scenes in Boston and Singapore and includes 1.4 million camera images. All objects captured by six cameras in the nuScenes dataset have a semantic category and 3D bounding boxes and attributes for each frame in which they appear. It allows the objector to accurately infer an object's position and direction in space compared with a 2D bounding box. Around 18 368 images without the annotation information are selected as the unlabeled data.

Our work does not depend on speci c data distribution and is e ective regardless of the dataset.

4.1.3 Adversarial attack

For Dpatch, the objective function of an untargeted attack is:

$$\hat{P}_{\mathrm{u}} = \arg\max_{P} \mathbb{E}_{x;s}\left[L(A(x,s,P);\hat{y},\hat{B})\right]. \tag{6}$$

$A(x,s,p)$ means adding patch P on the scene x with shift s. The loss function should be maximized to obtain the adversarial patch for untargeted attacks. The label $\hat{y}$ and the ground-truth bounding box $\hat{B}$ keep the real value unchanged. The objective function of a targeted attack is:

$$\hat{P}_{\mathrm{t}} = \arg\min_{P} \mathbb{E}_{x;s}\left[L(A(x,s,P);y_{\mathrm{t}},B_{\mathrm{t}})\right]. \tag{7}$$

The label y_t and bounding box B_t are target annotations. The goal is to train a patch that can raise the attention of the detector to the location of the patch and recognize the patch as the expected target class.

In the experiment, it is shown that an untargeted attack is not as e ective as a targeted attack, in which it is assumed that the target class is \car". Thus, the targeted attack method is employed in our robustness tests.

The patch size is set to 100× 100, and the normalized pixel values of the patch are initiated to random numbers uniformly distributed from 0 to 1. The patch is put on the top left to avoid occluding the objects. Then, the values of model parameters are xed, and the patch pixel values are regarded as new parameters. The patch pixel values are updated via stochastic gradient descent.

For Contextual Patch, two sorts of blindness attacks are provided: per-image attack and universal attack. The di erence between a per-image attack and a universal attack is that, in the per-image method, each image has its patch, and every patch is merely e ective in its corresponding image. A universal patch is trained to apply to all the images in the universal method. The per-image patch can achieve a better attack e ect than the universal patch. Nevertheless, this approach could be more practical in the physical world. Thus, we select the universal attack in our experiment.

The objective is to minimize the probability P(category|object) of the target class. The patch has a size of 100× 100 and is attached to the top left of the image. The patch pixel values are initialized with all zero values and updated via projected gradient descent (PGD).

For black-box attacks, six corruptions are applied, namely Snow, Fog, Frost, Gaussian, Impulse, and Shot. The three corruptions, Snow, Fog, and Frost, imitate di erent weathers in autonomous driving conditions. Gaussian, Impulse, and Shot are caused by the nature of light or the camera itself. The parameter Severity = 1 is set for all corruptions, which indicates that the intensity of corruption is at the lowest level.

4.1.4 Architecture

The detection approaches keep their original settings unchanged, that is, YOLOv2 with the backbone of Darknet-19 [10], YOLOv4 with the backbone of CSPDarknet-53 [12], SSD with the backbone of VGG-16 [17], Faster-RCNN and MoCo with the backbone of Resnet-50 [16]. Moco is the baseline model of this study.

4.1.5 Training strategy

In the self-supervised pre-training phase, MoCov2 [27] is trained longer with 800 epochs and MoCov1 [26] is trained only with 200 epochs. For YOLOv2, YOLOv4, and SSD, a model is trained on KITTI to build a benchmark. For Faster-RCNN, instead of using a multi-step learning rate strategy, the cosine annealing learning rate strategy with the warm restart setting is utilized. In particular, the parameter is set as T_mult = 2, which means that the next period is twice as long as the one before. In this way, in the later training period, the learning rate will not increase but decline until the end of the training, and validation accuracy will always achieve the optimal point at the lowest learning rate.

The batch size is 2, and the model is trained for 10 340 iterations, where the learning rate arrives at the bottom and validation loss achieves the minimum value. Our model is trained on two 1080Ti GPUs, and each semi-supervised training process costs 23 h.

As for BBAug, the augmentation policy includes 20 sub-policies and each sub-policy consists of two kinds of augmentations. The transformations comprise color operations such as Equalize, Sharpness, Color, Contrast, Brightness, Solarize, and geometric operations such as Translate, Cutout, Shear, Rotate, and Flip.

4.1.6 Metrics

The most commonly used evaluation index of object detection, mAP (mean Average Precision), is applied to measure the performance of distinct models. The main idea is to draw a precision-recall curve and then calculate the area under the PR curve as the AP value. In particular, the mAP de ned at VOC07 [59] is used, in which the IoU threshold is 0.5. Higher mAP denotes a better performance of the detector.

4.2 Main result of baseline

The performance of distinct detectors on KITTI is depicted in Table 1. In contrast to one-stage algorithms such as YOLOv2, YOLOv4, and SSD, the two-stage algorithm outperforms these approaches in accuracy by a large margin, and the mAP surpasses more than 20% compared with the one-stage detector.

In our experiments, Dpatch is trained to be \disguised" like a car to mislead the detector to mistake a patch for a car. This targeted attack can conduct a more successful attack e ect compared with the untargeted attack. When the Dpatch adversary attacks these models, SSD is the most vulnerable, and the mAP drops by 15.83%. YOLOv2 and Faster-RCNN with supervised pre-training are also a ected by a decrease of mAP by 8.15% and 5.84%. Faster-RCNN with self-supervised MoCo pre-training possesses the strongest robustness against Dpatch, and the mAP barely drops after the attack.

For the contextual adversarial patch attack, the classi cation score of a particular class is minimized to make the detector \blind" to this speci c class. The experimental data proves that SSD has the worst defense capability against the contextual patch, with a decline of mAP by 10.35%. The mAP of YOLOv2 and YOLOv4 have decreased to some extent by 4.33% and 6.17%. Still, Faster-RCNN combined with MoCo pre-training achieves the greatest performance against the contextual patch and the performance is stable under the attack of the contextual patch

In addition, experiments are also designed to test whether the patches trained in a white-box manner can attack the other models. The results are displayed in Table 2. The bold type words indicate a white-box attack. In other words, the generated adversarial patch attacks the model with the same structure. The results show that most models are robust to the transferred contextual patch. The only transfer-based black-box attack is the white box patches of YOLOv2 training transferring to YOLOv4 because the model structure is similar.

Although Faster-RCNN with MoCo is robust to white-box attacks, it does not show the same ability as before in the black-box attack. By comparison, the one-stage methods are more robust, with a drop of mAP by 15.61%. Nevertheless, the mAP of Faster-RCNN decreases by 30.45%, which is a signi cant degradation of robustness. Even with MoCo pre-training, the improvement in robustness is minimal. To summarize, transfer learning based on MoCo establishes a baseline for acceptable generalization. Although the employment of MoCo pre-training brings about strong robustness against the white-box attack, Faster-RCNN's defense ability against black-box attacks remains weak. Due to the mediocre robustness of the current baseline, a new approach is implemented to boost the robustness of the current model. Menwhile, the generalization ability should not be sacri ced too much as the cost of robustness improvement.

4.3 In uence of unlabeled data

Unlabeled data from nuScenes boosts the detection accuracy and robustness, especially for the black-box attack. Table 3 shows the result. In particular, we have utilized the model pre-trained with MoCov1 and transferred to Faster-RCNN for the subsequent semi-supervised retraining process, since the performance of Faster-RCNN is superior compared with the other detection models. In theory, another model pre-trained with MoCov2 can also be utilized for the corresponding experiment. However, the performance of this model even deteriorates after semi-supervised learning. We will discuss the worse performance of the model based on MoCov2 further.

Before semi-supervised retraining, pseudo-labels should be generated through the collaboration of models based on transferred MoCov1 and transferred MoCov2. In order to lter out the unlabeled images in which objects have low classi cation con dence, we conduct an experiment with threshold study of prediction con dence to decide whether the images and the objects should be retained. Three values 0.5, 0.7, and 0.9 are selected as the threshold. As the results indicate, the threshold plays a vital role in the performance of this model. The mAP on both clean data and noisy data can obtain an improvement when we set the threshold at 0.5 and 0.9. In particular, compared with standard MoCo framework, our co-training method with the threshold at 0.9 has gained the most signi cant increase by 4.38% on clean data. It increases by 1.97% on data added with black-box perturbation in contrast to the counterpart trained without unlabeled data. Nevertheless, the one with the threshold at 0.7 does not show an improvement. The performance drops by 1.61% on clean data and by 3.45% on noisy data with the black-box attack. The model always remains robust against the white-box attack.

Table 1. Robustness test of baseline models on KITTI (AP and mAP)

	YOLOv2				YOLOv4				SSD			
		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)
White-box	Clean	65.17	47.98	40.24	51.13	Clean	80.77	44.89	55.27	60.31	Clean	82.11	46.42	41.99	56.83
	Dpatch	44.33	44.48	40.12	42.98	Dpatch	80.60	44.15	54.32	59.69	Dpatch	81.02	38.85	3.13	41.00
	Contextual patch	58.04	43.20	39.15	46.80	Contextual patch	79.79	43.13	39.49	54.14	Contextual patch	82.76	46.80	9.87	46.48
Black-box	Noise_snow	55.74	18.94	27.25	33.97	Noise_snow	63.70	19.41	27.31	36.81	Noise_snow	73.54	20.13	33.40	42.36
	Noise_fog	56.86	36.27	30.56	41.23	Noise_fog	63.77	29.53	42.07	45.12	Noise_fog	58.36	14.19	20.82	31.12
	Noise_frost	59.53	29.85	31.49	40.29	Noise_frost	67.55	28.75	37.35	44.55	Noise_frost	66.14	19.36	27.15	37.55
	Noise_gaussian	59.02	29.49	32.71	40.41	Noise_gaussian	75.49	38.32	43.91	52.57	Noise_gaussian	71.95	24.00	35.30	43.75
	Noise_shot	61.50	39.87	37.54	39.87	Noise_shot	79.29	43.93	51.03	58.08	Noise_shot	78.03	37.12	40.65	51.94
	Noise_impulse	57.15	27.68	31.26	38.70	Noise_impulse	74.34	28.43	40.13	47.63	Noise_impulse	70.72	18.02	32.99	40.58
	Average of b-box	58.30	30.35	31.80	39.08	Average of b-box	70.69	31.40	40.30	47.46	Average of b-box	69.79	22.14	31.72	41.22
	Faster-RCNN					MoCov1+Faster-RCNN					MoCov2+Faster-RCNN				
		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)
White-box	Clean	90.27	80.20	79.62	83.20	Clean	90.03	85.05	78.12	84.40	Clean	89.87	82.48	77.40	83.25
	Dpatch	73.66	80.78	77.63	77.36	Dpatch	90.01	84.76	78.19	84.32	Dpatch	90.10	81.05	77.86	83.00
	Contextual patch	90.26	80.78	74.84	81.96	Contextual patch	90.06	84.05	78.51	84.21	Contextual patch	90.09	81.04	77.79	82.97
Black-box	Noise_snow	70.50	32.50	34.34	45.78	Noise_snow	71.17	40.26	40.31	50.58	Noise_snow	69.89	36.62	34.23	46.92
	Noise_fog	53.95	41.79	42.59	46.11	Noise_fog	71.11	44.32	43.74	53.06	Noise_fog	53.43	35.96	41.52	43.64
	Noise_frost	80.17	52.83	52.52	61.84	Noise_frost	79.42	57.06	51.32	62.60	Noise_frost	71.32	52.41	50.69	58.14
	Noise_gaussian	71.53	33.78	43.34	49.55	Noise_gaussian	71.24	36.58	50.68	52.83	Noise_gaussian	77.04	49.66	50.13	58.94
	Noise_shot	89.59	66.01	60.81	72.14	Noise_shot	88.98	64.75	67.77	73.84	Noise_shot	89.10	65.79	67.45	74.11
	Noise_impulse	62.77	20.66	39.74	41.06	Noise_impulse	53.80	24.54	40.21	39.52	Noise_impulse	62.60	28.31	40.78	43.90
	Average of b-box	71.42	41.26	45.56	52.75	Average of b-box	72.62	44.59	49.01	55.40	Average of b-box	70.56	44.79	47.47	54.27

Table 2. Performance under patch attack in a black-box manner (AP and mAP)

Cpatch in a black-box manner							Dpatch in a black-box manner					
	YOLOv2	YOLOv4	SSD	Faster-RCNN	MoCov1 +Faster-RCNN	MoCov2 +Faster-RCNN		YOLOv2	YOLOv4	SSD	Faster-RCNN	MoCov1 +Faster-RCNN	MoCov2 +Faster-RCNN
Clean	51.13	60.31	56.83	82.91	**84.40**	83.25	Clean	51.13	60.31	56.83	83.20	**84.40**	83.25
YOLOv2	46.80	53.07	53.28	82.91	**84.36**	82.86	YOLOv2	42.98	56.60	56.29	83.21	**84.36**	82.88
YOLOv4	50.27	54.14	56.80	82.90	**84.37**	83.01	YOLOv4	50.27	59.69	56.78	83.21	**84.38**	83.01
SSD	50.30	59.26	46.48	82.92	**84.37**	83.03	SSD	50.31	59.17	41.00	83.20	**83.36**	83.02
Faster-RCNN	48.50	55.13	56.80	81.96	**83.36**	82.94	Faster-RCNN	50.44	57.78	56.84	77.36	**83.32**	82.96
MoCov1+Faster-RCNN	50.41	57.65	56.82	82.91	**84.20**	82.99	MoCov1+Faster-RCNN	50.44	57.59	56.74	83.20	**84.32**	82.90
MoCov2+Faster-RCNN	50.26	56.07	56.42	82.93	**83.40**	82.97	MoCov2+Faster-RCNN	50.30	57.23	56.79	83.20	**83.39**	83.00

Table 3. Performance of co-training with difference filter threshold of 0.5, 0.7, and 0.9 (AP and mAP)

	Semi-supervised_0.5					Semi-supervised_0.7					Semi-supervised_0.9				
		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)
White-box	Clean	90.58	**89.79**	80.09	86.82	Clean	89.38	76.65	82.33	82.79	Clean	**90.61**	89.17	**86.57**	**88.78**
	Dpatch	**90.55**	**89.60**	80.04	86.73	Dpatch	89.28	76.20	82.14	82.54	Dpatch	90.54	89.18	**86.44**	**88.72**
	Contextual patch	90.53	**89.66**	80.07	86.75	Contextual patch	89.22	76.24	82.10	82.52	Contextual patch	**90.56**	89.16	**86.25**	**88.66**
Black-box	Noise_snow	**62.66**	33.53	41.34	45.85	Noise_snow	62.13	31.65	**41.83**	45.20	Noise_snow	62.42	**40.45**	41.60	**48.16**
	Noise_fog	**63.09**	**52.70**	51.44	**55.74**	Noise_fog	53.38	36.36	43.58	44.44	Noise_fog	62.73	44.74	**52.60**	53.36
	Noise_frost	**72.05**	**69.05**	52.25	**64.45**	Noise_frost	70.44	50.45	50.92	57.27	Noise_frost	71.91	60.97	**59.06**	63.98
	Noise_gaussian	**71.87**	41.08	**51.40**	54.78	Noise_gaussian	71.72	34.42	48.79	51.64	Noise_gaussian	71.72	**43.32**	51.33	**55.46**
	Noise_shot	89.66	**75.60**	69.49	78.25	Noise_shot	81.11	67.69	67.28	72.03	Noise_shot	**89.82**	75.36	**70.36**	**78.51**
	Noise_impulse	54.20	**30.25**	42.09	42.18	Noise_impulse	**62.62**	21.63	39.18	41.14	Noise_impulse	62.56	29.58	**42.13**	**44.76**
	Average of b-box	68.92	**50.37**	51.34	56.87	Average of b-box	66.90	40.37	48.60	51.95	Average of b-box	**70.19**	49.07	**52.85**	**57.37**

Figure 2. An example with the setting of corruption = fog, severity = 1. Our method surpasses Faster-RCNN by detecting more objects including cyclists and pedestrians. Also, BBAug greatly boosts the robustness compared with the counterpart without BBAug: (a) Ground-truth annotation, (b) result of Faster-RCNN with supervised pretrained model, (c) result of Faster-RCNN with self-supervised pretrained MoCo, (d) result of Co-training without BBAug, (e) result of Co-training with BBAug

After co-training, the Faster-RCNN model based on MoCo shows improvement in generalization ability and robustness in contrast to the supervised ne-tuned counterpart based on MoCo. Compared to the one-stage detection method, this approach is still vulnerable to black-box attacks. Thus, to further improve the robustness, BBAug as data augmentation is employed to strengthen this model. The results are shown in Table 4. In contrast to the above co-training, the performance of this model on clean data declines to some extent and the threshold at 0.9 drops at the most by 10.06%. The model with the threshold of prediction con dence 0.5 achieves the best performance. Although the mAP decreases compared to the counterpart without BBAug, the robustness against black-box attack is improved by 5.44% on average. Besides, the detection accuracy of the model on clean data drops slightly compared to the MoCo baseline,

Table 4. Performance of co-training with BBAug (AP and mAP)

	Semi-supervised_0.5_BBAug					Semi-supervised_0.7_BBAug					Semi-supervised_0.9_BBAug				
		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)		Car	Cyclist	Pedestrian	mAP (%)
White-box	Clean	**89.79**	**83.17**	**77.50**	**83.49**	clean	87.99	67.85	72.26	76.04	Clean	88.13	72.64	75.40	78.72
	Dpatch	**89.75**	**83.13**	**77.50**	**83.46**	Dpatch	87.94	67.62	71.10	75.55	Dpatch	87.92	72.63	75.40	78.65
	Contextual patch	**89.53**	**83.17**	**77.44**	**83.38**	Contextual patch	87.88	67.45	72.03	75.79	Contextual patch	88.09	72.42	75.22	78.58
Black-box	Noise_snow	**68.53**	**38.20**	40.37	49.03	Noise_snow	67.57	37.49	**42.07**	**49.04**	Noise_snow	59.27	37.04	33.40	43.24
	Noise_fog	**88.54**	**68.11**	**67.23**	**74.62**	Noise_fog	79.53	56.07	55.77	63.79	Noise_fog	79.40	61.21	64.60	68.40
	Noise_frost	**87.02**	**71.20**	**65.78**	**74.66**	Noise_frost	77.79	53.34	59.98	63.70	Noise_frost	77.39	61.60	60.44	66.48
	Noise_gaussian	**70.35**	**47.05**	**48.87**	**55.42**	Noise_gaussian	68.91	36.36	44.07	49.78	Noise_gaussian	60.92	35.97	44.50	47.13
	Noise_shot	**88.13**	**70.01**	**65.54**	**74.56**	Noise_shot	79.34	53.80	54.26	62.46	Noise_shot	78.98	56.27	57.74	64.33
	Noise_impulse	**61.20**	**35.34**	**40.06**	**45.54**	Noise_impulse	60.35	24.72	35.91	40.32	Noise_impulse	52.03	20.39	30.69	34.37
	Average of b-box	**77.30**	**54.99**	**54.64**	**62.31**	Average of b-box	72.25	43.63	48.68	54.85	Average of b-box	68.00	45.41	48.56	53.99

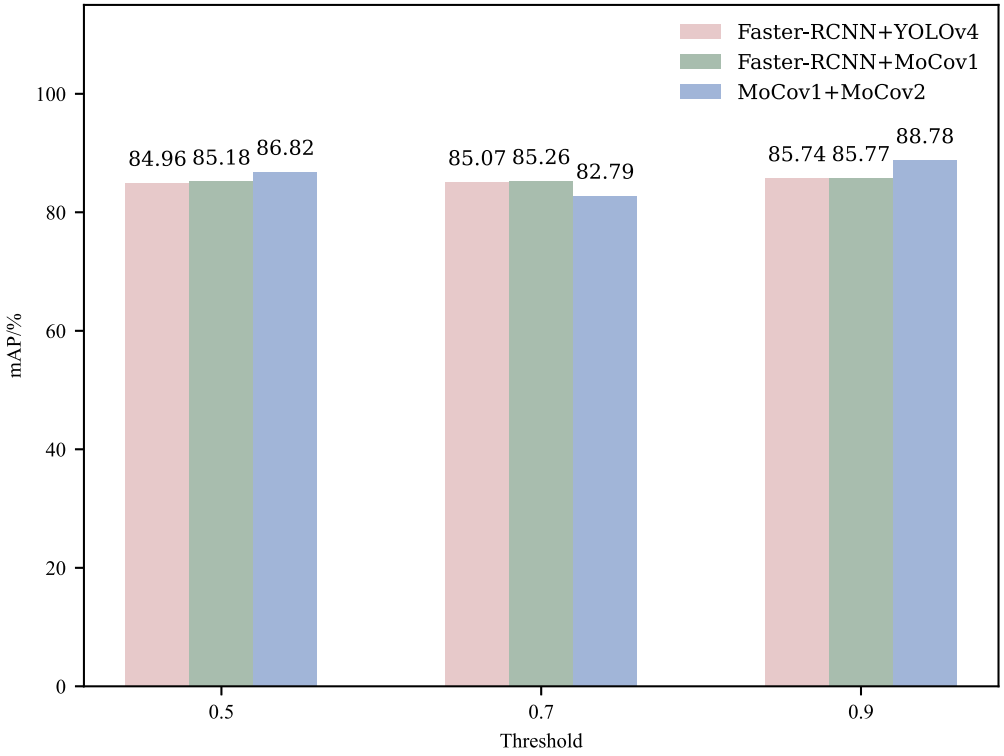

Figure 3. The blue bars show the performance of our method on clean dataset. The other two colors of the bars are based on supervised pre-training instead of MoCo. The pink bars indicate using supervised Faster-RCNN and YOLOv4 to generate pseudo-labels and the green bars indicate using supervised Faster-RCNN and Faster-RCNN based on MoCov1. After applying MoCo, the general detection performance on clean dataset is improved

and the robustness against black-box noise is strengthened by 6.91%. Thus, our method makes up for the de ciency and maintains high detection accuracy.

The e ect of our approach is also shown in Figure 2, where the qualitative analysis shows the robustness of our method under the black-box attacks. Our proposed method detects more objects with high con dence in a foggy condition. In particular, our detector can detect tiny objects such as cyclists and pedestrians compared with Faster-RCNN, demonstrating superior robustness.

In order to verify the superiority of MoCo in the semi-supervised co-training, experiments are also implemented. The evaluation results on the clean dataset are shown in Figure 3X-coordinate signi es the ltering out threshold of prediction con dence for generating pseudo-labels. Y-coordinate signi es mAP in percentage. The pink bars and green bars denote the co-training based on traditional supervised pre-training, and the blue bars denote the co-training based on MoCo self-supervised pre-training. For simplicity of expression, the Faster-RCNN in this gure denotes the model built on supervised pre-training. MoCov1 and MoCov2 denote the models pre-trained with MoCo and transferred through Faster-RCNN. The pink bars denote using models trained with supervised Faster-RCNN and YOLOv4 to generate pseudo-labels for co-training. The green bars denote supervised Faster-RCNN and Faster-RCNN based on MoCov1 for co-training. The result of our method, namely using MoCov1 and MoCov2 for co-training, is displayed in blue bars. In the threshold of 0.5 and 0.9, our method surpasses the other two groups that use supervised pre-trained models. Thus, applying MoCo can e ectively boost the performance of the detector.

4.4 Comparison with recent methods

We also implement the experiments of the robustness test of two recent anchor-free one-stage detection methods, namely Fully Convolutional One-Stage Object Detection (FCOS) [60] and CenterNet [61]. Table 5 demonstrates the results. In comparison to our semi-supervised method at the ltering threshold

Table 5. Robustness test of recent methods (AP and mAP)

	CenterNet					FCOS					Semi-supervised_0.5_BBAug				
		Car	Cyclist	Pedestrian	mAP(%)		Car	Cyclist	Pedestrian	mAP(%)		Car	Cyclist	Pedestrian	mAP(%)
White-box	Clean	**92.39**	86.50	68.70	82.53	clean	90.31	**87.69**	**78.18**	**85.39**	clean	89.79	83.17	77.50	83.49
	Dpatch	87.89	70.82	61.54	73.42	Dpatch	79.06	64.79	65.71	69.86	Dpatch	**89.75**	**83.13**	**77.50**	**83.46**
	Contextual patch	87.20	69.60	60.20	72.33	Contextual patch	78.02	65.46	64.13	69.20	Contextual patch	**89.53**	**83.17**	**77.44**	**83.38**
Black-box	Noise_snow	**83.55**	**65.41**	47.52	**65.49**	Noise_snow	70.51	47.07	**50.80**	56.13	Noise_snow	68.53	38.20	40.37	49.03
	Noise_fog	70.94	52.25	39.73	54.31	Noise_fog	71.42	63.00	59.12	64.51	Noise_fog	**88.54**	**68.11**	**67.23**	**74.62**
	Noise_frost	77.86	59.07	43.91	60.28	Noise_frost	80.25	65.25	60.82	68.77	Noise_frost	**87.02**	**71.20**	**65.78**	**74.66**
	Noise_gaussian	**85.93**	**69.74**	52.46	**69.38**	Noise_gaussian	80.03	62.59	**58.92**	67.18	Noise_gaussian	70.35	47.05	48.87	55.42
	Noise_shot	**91.37**	**78.25**	65.04	**78.22**	Noise_shot	87.81	68.31	**67.71**	74.61	Noise_shot	88.13	70.01	65.54	74.56
	Noise_impulse	**77.40**	**58.40**	45.06	**60.29**	Noise_impulse	71.56	44.78	**55.75**	57.36	Noise_impulse	61.20	35.34	40.06	45.54
	Average of b-box	**81.18**	**63.85**	48.95	64.66	Average of b-box	76.93	58.50	**58.85**	**64.76**	Average of b-box	77.30	54.99	54.64	62.31

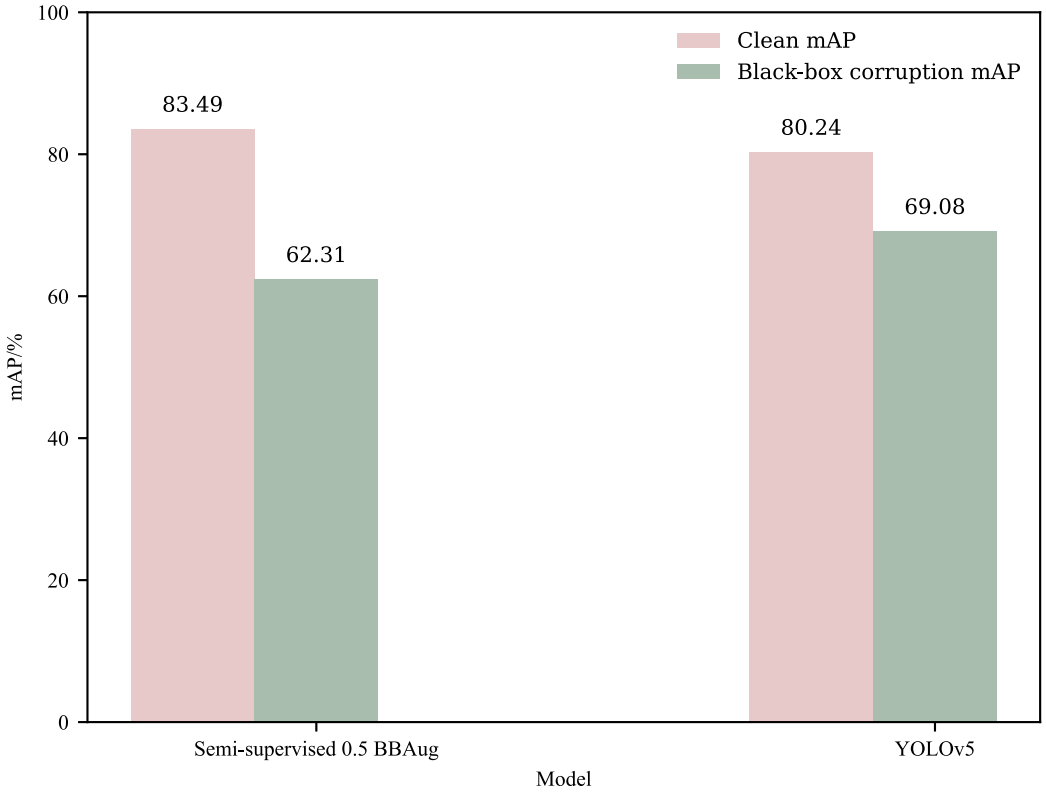

Figure 4. The comparison between YOLOv5 and theproposed framework with the co-training on the threshold confidence of 0.5 and bbaug method

of prediction con dence 0.5 in Table 4, CenterNet and FCOS outperform our method by 2.35% and 2.45% respectively for black-box attacks. Nevertheless, CenterNet and FCOS show a worse defense ability against white-box attacks. Compared with the performance on the clean data, the mAP of CenterNet and FCOS have dropped by 9.66% and 15.86% respectively on average concerning white-box attacks. The structure of Faster-RCNN is not as novel as CenterNet, but our method still shows robustness to some extent.

Our proposed framework satis es the rst principle; however, it can be enhanced further by incorporating additional engineering methods. For instance, we can leverage YOLOv5 [62], which utilizes a deeper backbone and incorporates multi-scale feature prediction and data augmentation methods like CutMix and Mosaic. As shown in Figure 4, our proposed method performs better without perturbations. However, the YOLOv5, with more fancy training tricks, demonstrates better robustness against black-box attacks. Therefore, integrating several training strategies inspired by YOLOv5 could be considered as future work.

4.5 Ablation study

Table 1 shows that the model with MoCo makes some progress in both natural and perturbed conditions. In order to study the in uence of the ltering out threshold of unlabeled data and the data augmentation, we take the testing results of supervised Faster-RCNN and MoCov1-based Faster-RCNN as baselines for comparison. Moreover, we implement an ablation experiment to decide the best scheme. Figure 5 displays the comparison results, where the X-axis represents the threshold of sifting out unlabeled data, and the Y-axis represents mAP in percentage form. The black and orange dashed lines denote the baseline model built on MoCo and the supervised learning baseline on the clean dataset. The red and violet ones denote the two baselines on the black-box noisy dataset. The blue solid lines mean using co-training without BBAug, and the green lines denote using co-training added with BBAug. The symbol squares are the experimental results on the clean dataset, and the symbol triangles are the results on the noisy dataset.

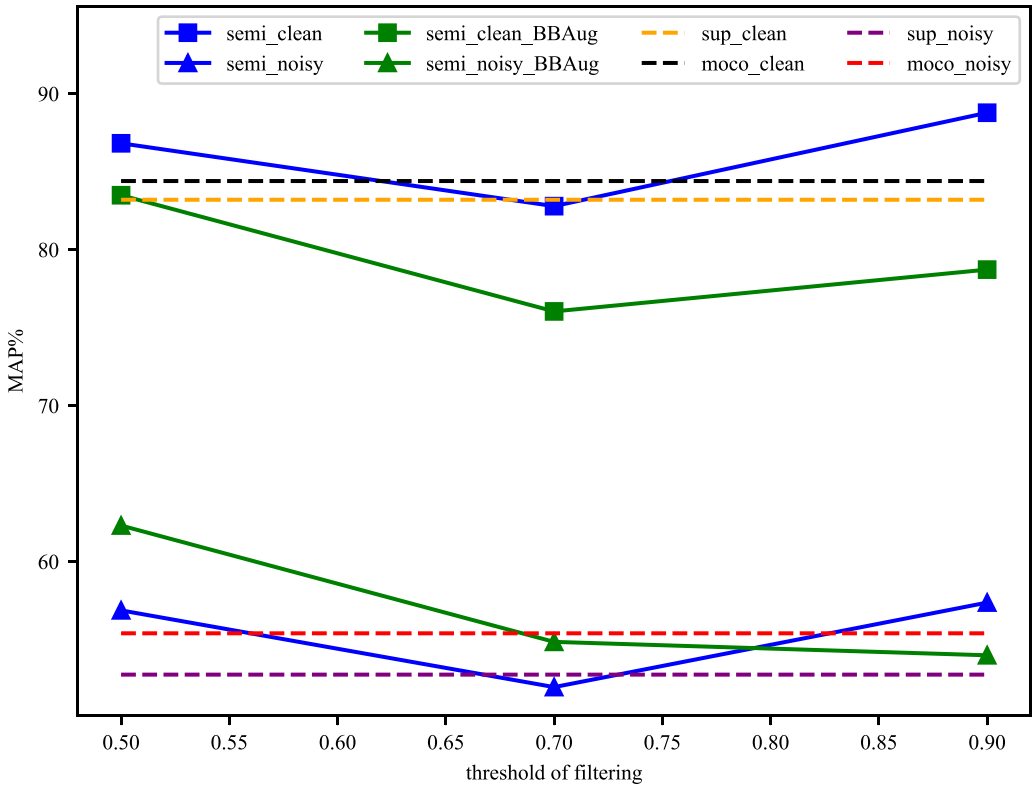

Figure 5. The four dashed lines serve as baselines. Blue lines denote co-training without BBAug and green lines are co-training with BBAug. The symbol squares are results on clean data and the triangles are results on noisy data. After adding BBAug, the mAP on the clean dataset goes down in general while the mAP on the noisy dataset goes up. Taking all this into account, the point with the threshold of prediction confidence at 0.5 and added with BBAug can meet the expectations of both clean and noisy datasets

After adding unlabeled data with pseudo-labels into the training dataset, the model performance is boosted on clean data when choosing threshold of prediction con dence 0.5 and 0.9 but deteriorates when choosing threshold 0.7. After adding BBAug into co-training, mAP drops to some extent. As the ltering threshold rises, the mAP drops more.

For adversarial attacks, the robustness against white-box attacks is improved slightly due to the use of MoCo. After that, it keeps a stable value. Thus, the ablation study pays more attention to the black-box attack. After semi-supervised co-training, the mAP on noisy data rises slightly when the sifting out threshold of unlabeled data lies at 0.5 and 0.9. Nevertheless, when the di erence of mAP between clean data and noisy data is analyzed, it does not go down. Thus, co-training alone can not improve the robustness against the black-box attack. Using the BBAug method can strengthen the robustness of the model. Although robustness is lifted at the expense of detection accuracy on the clean dataset, this sacri ce is negligible compared with the increase in robustness.

Our achievement is also shown in Figure 6, where an evident robustness improvement is displayed when using the combination of semi-supervised co-training and BBAug augmentation. Here, the \noisy" bar denotes the dataset corrupted by Image Corruptions, namely black-box attacks.

4.6 Study of di erent ratios of new labeled data

We also test the in uence of di erent ratios of newly labeled data by co-training without adding the BBAug method into the test. One group has 8120 new labeled images (full), while another has 1624 new labeled images. Figure 7 illustrates some surprising results. The group with the full-data setting performs better without perturbations, while only 20% of newly labeled data achieves better robustness under black-box corruptions. This nding highlights the limitations of co-training [51]. Firstly, the co-training

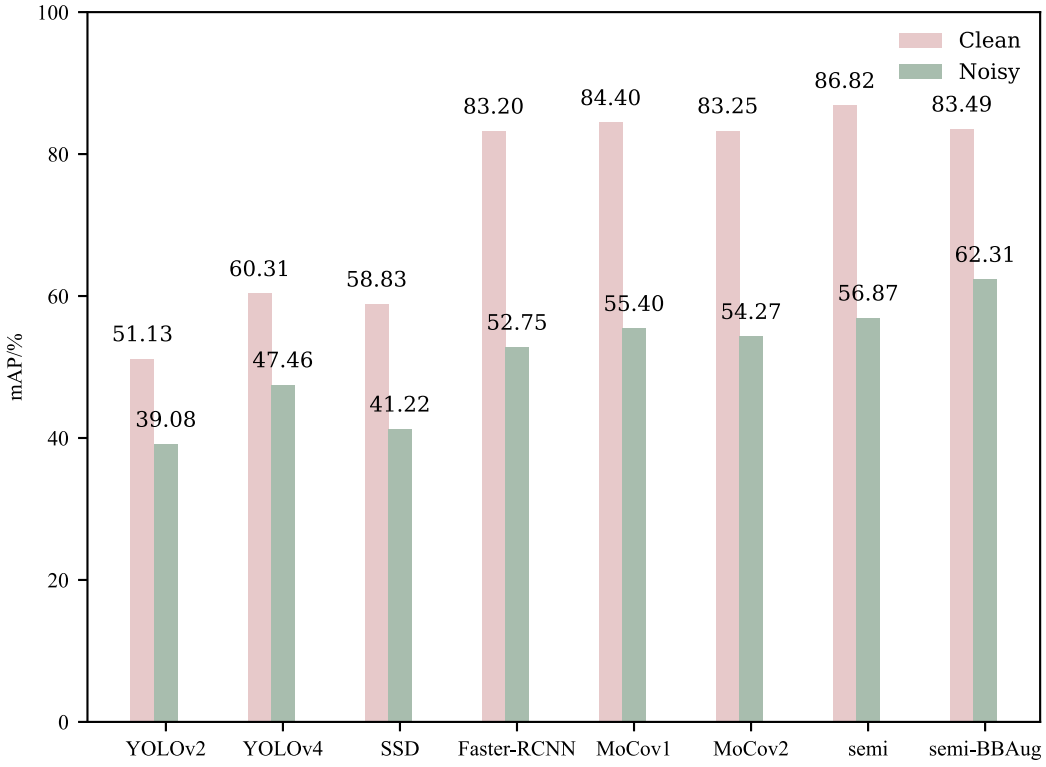

Figure 6. Our method with the confidence threshold 0.5 for the semi-supervised learning surpasses supervised learning methods both in generalization and robustness. Compared with the MoCo baseline, the performance on the clean dataset drops slightly, while it obtains a great improvement on the noisy dataset (Black-box attack)

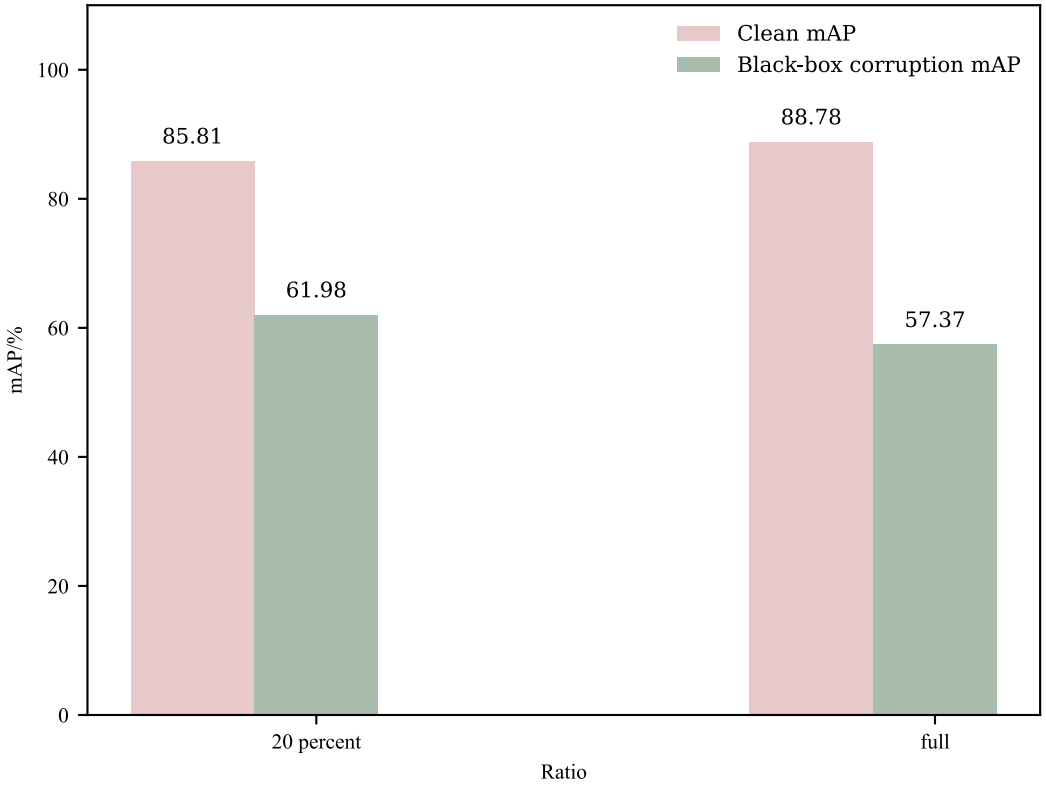

Figure 7. Performance of different ratios of new labeled data

Table 6. Inference time of different models (second)

Models	Inference time (s/img)
Yolov2	0.027
Yolov4	0.015
SSD	0.006
CenterNet	0.106
FCOS	0.092
Faster-RCNN	0.448
Faster-RCNN+MoCov1	0.448
Faster-RCNN+MoCov2	0.448
Our method	0.448

approach relies on the in-distribution assumption, assuming that the label distribution of the training data is consistent, meaning that labeled and unlabeled data have similar distributions. However, in practical situations, this assumption may not hold. Secondly, there is a problem of error propagation, where the risk exists that when one model uses its predictions as labels for another model, incorrect labels may propagate and lead to further errors. It can result in performance degradation, especially when noises or incorrect labels exist in the unlabeled data. Furthermore, there is a dependence on feature representation in co-training. The e ectiveness of co-training relies on extracting useful feature representations so that both models can learn from labeled and unlabeled data. If the quality of the feature representation is low or inappropriate, the e ectiveness of co-training may be limited. For instance, our labeling quality is coarse-grained without su cient human checks. Therefore, we are unsure about the su cient percentage of unlabeled data in real applications. Exploring additional methods such as the mean teacher method [63] and unbiased teacher method [64] could be meaningful to enhance performance. Additionally, leveraging foundation models like segment-anything [65] can help improve the quality of labeling.

4.7 Inference time

Detection time is signi cant in the application scenario of autonomous driving. Table 6 lists the inference time of di erent models. Since our method builds on a two-stage method of Faster-RCNN and focuses on improving the training regime, the detection time is almost the same as that in the original Faster-RCNN and is not superior to other methods.

4.8 Discussion

The above experiment results have demonstrated that transfer learning based on the self-supervised method MoCo can improve the detection model's robustness against white-box attacks. Our method, namely semi-supervised co-training based on MoCo combined with BBAug augmentation, can promote the generalization ability and signi cantly boost the model's robustness against black-box attacks. However, some details in the experiment results are also worth analyzing, which would inspire further research.

Why does the performance of MoCov1 surpass MoCov2 after transfer learning and co-training? In theory, downstream transfer learning with MoCov2 is expected to outperform MoCov1 based on empirical data from previous research work [26, 27]. However, our experiment yields the opposite result. We speculate that the longer pre-training of MoCov2 may have led to over tting on the KITTI dataset. Additionally, MoCov2 incorporates a projector head and augmentation inspired by SimCLR during pre-training. However, this strategy does not contribute signi cantly to object detection for autonomous driving, possibly due to the sparse scenarios encountered in this domain. As a result, MoCov1 exhibits better performance than MoCov2 after transfer learning and co-training. Certainly, tuning training parameters such as regularization items, learning rate, or batch size of MoCov2 may improve the performance of MoCov2 and mitigate the over- tting problem. Due to the limitations of our current available computing resources, we leave it as a future research problem.

Why does the mAP on a clean dataset go down after adding BBAug in the process of co-training, while the mAP on a noisy dataset goes up? There exists a trade-o between adversarial robustness and generalization ability [66]. Robust models learn more salient features than standard models, which are more intuitive to humans. BBAug may have a similar mechanism as adversarial training, which learns more interpretable features and discards some unrelated features. In a real-world application, white-box and black-box adversarial attacks and adverse weather conditions represent the corner cases of autonomous driving. Striking a signi cant trade-o between natural scenario performance and adversarial robustness is crucial.

In particular, the model with the threshold of prediction con dence at 0.9 learns from the well-generalizing but susceptible features of the data [67], such small changes to the models can dramatically alter the model's predictions, thus the mAP drops the largest.

Why does the model with the ltering threshold of prediction con dence at 0.9 achieve the best performance on the clean dataset (without BBAug)? From the information theory perspective, a robust model with high con dence outputs means a low entropy and uncertainty. A pseudo-label with a low uncertainty helps extract more precise features.

Analysis of the e ectiveness and limitation of BBAug: BBAug extends the original dataset by applying transformations such as rotation, scaling, translation, and ip, and adjusts and updates the position and size of the bounding box accordingly to maintain the correspondence between the bounding box and the image content. This approach can help improve model performance and robustness for object recognition and instance segmentation tasks. The method of BBAug, as a data augmentation technique, aims to increase the diversity of training data and improve the generalization ability of deep learning models. It can be combined with other data enhancement techniques, such as random cropping, rotation, and color transformations, to enhance the diversity of the training dataset. This enhancement of data diversity can improve the generalization of object detection models and their adversarial robustness. A weak counteract e ect exists when the framework combines semi-supervised learning and BBAug. The KITTI dataset [32] samples the cityscapes in Western European cities, while the nuScenes dataset [31] samples the cityscapes in the United States and Singapore. The slight domain discrepancy between the two datasets is dominant. Such a domain discrepancy further leads to a slight \forgetting" e ect, which should be adjusted with the suitable sample amounts and annotation threshold. Moreover, the autonomous driving datasets are characterized by a long-tailed distribution, with fewer pedestrian and bicycle samples, and uctuations in performance due to this data imbalance may be ampli ed by data augmentation.

Black-box corruptions and SOTIF: The black-box adversarial attacks generated with noise corruptions can be regarded as test cases of the safety of the intended functionality (SOTIF). Thus, the security risk issues can be converted to the test cases of safety boundaries. The delity of imitation of the real physical world is another challenging issue. For example, the snowy weather may cover the lanes, and the rainy weather may cause water re ection on the road surface. In the future, combining the generative model [68] and the black-box corruptions to simulate the adverse weather in autonomous driving is meaningful.

5 Conclusion

In recent years, more research has focused on semi-supervised and self-supervised learning and their applications in computer vision. Unlike supervised learning, self-supervised or semi-supervised learning is closer to how the human brain perceives things. Nevertheless, few previous studies covered the research of semi-supervised learning for the robustness enhancement applied in autonomous driving.

We apply semi-supervised learning in the real autonomous driving scenario. Firstly, a baseline with transfer learning builds on the contrastive learning framework MoCo. In the downstream task, we implement Faster-RCNN using the pre-trained MoCo-based model. This step boosts the detection accuracy and robustness against the white-box attack compared with the supervised learning counterpart. Next comes the co-training process: the two models based on MoCov1 and MoCov2 are used to generate pseudo-labels for unlabeled data from another autonomous driving dataset, nuScenes. Then, the re-training process utilizes these data and their pseudo-labels. We choose three thresholds of prediction con dence to lter out the data and nd that the one with 0.9 can achieve the best performance. This step helps improve the generalization ability. Finally, the BBAug method combines with the co-training method, which makes the target detector increase the network weights of robust features during learning.

The empirical study on our approach validates the e ectiveness of our framework and gives a reference to real applications in autonomous driving. Our method improves the robustness of object detection in autonomous driving. It can reduce annotation time costs and improve driving safety.

This work is a \start-up" attempt to apply the semi-supervised learning method to resolve the robustness problem in autonomous driving. It is a data-centric and robust learning framework. Some autonomous driving companies have begun to collect \shadow data" without interfering with human drivers' decision-making, allowing machines and testers to label and improve the safety of autonomous driving perception systems through iterative development. Nevertheless, some technical incremental work is still worth sustaining. In the future, we will further extend our robust semi-supervised learning framework. Other methods like the mean teacher method [63] and unbiased teacher method [64] can replace the co-training method.

Con ict of Interest

The authors declare that they have no conflict of interest.

Data Availability

The dataset of KITTI is available at: https://www.cvlibs.net/datasets/kitti/, and the dataset of nuScenes is available at: https://www.nuscenes.org/. Code is available at: https://github.com/CHENWenwen19/co-trainingforautonomous-driving

Authors' Contributions

Wenwen Chen and Jun Yan made equal contributions and were co-first authors. Wenwen Chen contributed to most of the experimental studies and manuscript writing. Jun Yan contributed to the framework design, some experimental studies, and manuscript writing. Prof. Huilin Yin, Prof. Huaping Liu, and Weiquan Huang helped revise the manuscript. Prof. Wancheng Ge supervised Wenwen Chen during her graduate carrer. Prof. Huilin Yin led this research project.

Acknowledgements

The authors would like to thank TÜV SÜD for its kind and generous support. We also thank Breton, an electric commercial vehicle maker, for support. We are grateful for the efforts of our colleagues at the Sino-German Center of Intelligent Systems, Tongji University.

Funding

This work was supported by the National Natural Science Foundation of China under Grant No. 61701348 and No. 62133011.

References

[1] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. Proc IEEE 1998; **86**: 2278–2324.
[2] Krizhevsky A, Sutskever I and Hinton GE. Imagenet classification with deep convolutional neural networks. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2012, 1097–1105.
[3] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 770–778.
[4] Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, 4700–4708.
[5] Hu J, Shen L and Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, 7132–7141.
[6] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks, 2015, 91–99.
[7] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector. In: European conference on computer vision (ECCV), 2016, 21–37.
[8] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 779–788.
[9] He K, Gkioxari G, Dollár P, et al. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, 2961–2969.
[10] Redmon J and Farhadi A. YOLO9000: Better, faster, stronger. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, 7263–7271.
[11] Redmon J and Farhadi A. YOLOv3: An incremental improvement, arXiv preprint arXiv:1804.02767, 2018.
[12] Bochkovskiy A, Wang CY and Mark Liao HY. YOLOv4: Optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934, 2020.
[13] Long J, Shelhamer E and Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2015, 3431–3440.
[14] Zhao H, Shi J, Qi X, et al. Pyramid scene parsing network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, 2881–2890.
[15] Chen LC, Papandreou G, Kokkinos I, et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans Pattern Anal Mach Intell 2017; **40**: 834–848.

[16] He K, Zhang X, Ren S, et al. Identity mappings in deep residual networks. In: European Conference on Computer Vision (ECCV), Springer, 2016, 630–645.
[17] Simonyan K and Zisserman A. Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations (ICLR), 2015.
[18] Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks. In: 2nd International Conference on Learning Representations (ICLR), 2014.
[19] Goodfellow IJ, Shlens J and Szegedy C. Explaining and harnessing adversarial examples. In: 3rd International Conference on Learning Representations (ICLR), 2015.
[20] Zhai R, Cai T, He D, et al. Adversarially robust generalization just requires more unlabeled data, arXiv preprint arXiv:1906.00555, 2019.
[21] Alayrac JB, Uesato J, Huang PS, et al. Are labels required for improving adversarial robustness? In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2019, 12192–12202.
[22] Najafi A, Maeda SI, Koyama M, et al. Robustness to adversarial perturbations in learning from incomplete data. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2019, 5542–5552.
[23] Carmon Y, Raghunathan A, Schmidt L, et al. Unlabeled data improves adversarial robustness. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2019, 11190–11201.
[24] Chen T, Kornblith S, Norouzi M, et al. A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning (ICML), PMLR, 2020, 1597–1607.
[25] Chen T, Kornblith S, Swersky K, et al. Big self-supervised models are strong semi-supervised learners. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2020.
[26] He K, Fan H, Wu Y, et al. Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, 9726–9735.
[27] Chen X, Fan H, Girshick RB, et al. Improved baselines with momentum contrastive learning, arXiv preprint arXiv:2003.04297, 2020.
[28] ISO. Road vehicles – safety of the intended functionality. In: International Organization for Standardization: ISO/DIS 21448, 2021.
[29] Krizhevsky A and Hinton G. A Learning Multiple Layers of Features from Tiny Images, 2009, http://www.cs.toronto.edu/~kriz/cifar.html
[30] LeCun Y and Cortes C. MNIST Handwritten Digit Database, 2010.
[31] Caesar H, Bankiti V, Lang AH, et al. nuScenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, 11621–11631.
[32] Geiger A, Lenz P, Stiller C, et al. Vision meets robotics: The kitti dataset. Int J Robot Res 2013; **32**: 1231–1237.
[33] Zou Z, Shi Z, Guo Y, et al. Object detection in 20 years: A survey, arXiv preprint arXiv:1905.05055, 2019.
[34] Zhang S, Wen L, Bian X, et al. Single-shot refinement neural network for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, 4203–4212.
[35] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2014, 580–587.
[36] Lin TY, Goyal P, Girshick R, et al. Focal loss for dense object detection. In: Proceedings of the IEEE international Conference on Computer Vision (ICCV), 2017, 2980–2988.
[37] Tanay T and Griffin LD. A boundary tilting persepective on the phenomenon of adversarial examples, arXiv preprint arXiv:1608.07690, 2016.
[38] Akhtar N and Mian A. Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access 2018; **6**: 14410–14430.
[39] Papernot N, McDaniel P, Jha S, et al. The limitations of deep learning in adversarial settings. In: IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2016, 372–387.
[40] Carlini N and Wagner D. Towards evaluating the robustness of neural networks. In: IEEE Symposium on Security and Pprivacy (S&P), IEEE, 2017, 39–57.
[41] Moosavi-Dezfooli SM, Fawzi A and Frossard P. Deepfool: A simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 2574–2582.
[42] Baluja S and Fischer I. Adversarial transformation networks: learning to generate adversarial examples, arXiv preprint arXiv:1703.09387, 2017.
[43] Liu X, Yang H, Liu Z, et al. DPATCH: An adversarial patch attack on object detectors. In: Workshop on Artificial Intelligence Safety co-located with the Thirty-Third AAAI Conference on Artificial Intelligence, Volume 2301 of CEUR Workshop Proceedings, 2019.
[44] Saha A, Subramanya A, Patil K, et al. Role of spatial context in adversarial robustness for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2020, 784–785.
[45] Hendrycks D and Dietterich TG. Benchmarking neural network robustness to common corruptions and perturbations. In: 7th International Conference on Learning Representations (ICLR), 2019.
[46] Michaelis C, Mitzkus B, Geirhos R, et al. Benchmarking robustness in object detection: Autonomous driving when winter is coming, arXiv preprint arXiv:1907.07484, 2019.
[47] Caron M, Misra I, Mairal J, et al. Unsupervised learning of visual features by contrasting cluster assignments. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2020.
[48] Grill JB, Strub F, Altché F, et al. Bootstrap your own latent – A new approach to self-supervised learning. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2020.
[49] Miyato T, Maeda SI, Koyama M, et al. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. IEEE Trans Pattern Anal Mach Intell 2018; **41**: 1979–1993.

[50] Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks. In: 6th International Conference on Learning Representations (ICLR), 2018.
[51] Blum A and Mitchell T. Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT), 1998, 92–100.
[52] Lee DH, et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning of International Conference on Machine Learning ICML, 2013, 896.
[53] van den Oord A, Li Y and Vinyals O. Representation learning with contrastive predictive coding, arXiv preprint arXiv:1807.03748, 2018.
[54] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2009, 248–255.
[55] He T, Zhang Z, Zhang H, et al. Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, 558–567.
[56] Han W, Feng R, Wang L, et al. A semi-supervised generative framework with deep learning features for high-resolution remote sensing image scene classification. ISPRS J Photogr Remote Sens 2018; **145**: 23–43.
[57] Zoph B, Cubuk ED, Ghiasi G, et al. Learning data augmentation strategies for object detection. In: European Conference on Computer Vision (ECCV), 2020, 566–583.
[58] Cubuk ED, Zoph B, Mané D, et al. Autoaugment: Learning augmentation policies from data, arXiv preprint arXiv:1805.09501, 2018.
[59] Everingham M, Van Gool L, Williams CKI, et al. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html
[60] Tian Z, Shen C, Chen H, et al. Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, 9627–9636.
[61] Duan K, Bai S, Xie L, et al. Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), 2019, 6569–6578.
[62] Terven J and Cordova-Esparza D. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond, arXiv preprint arXiv:2304.00501, 2023.
[63] Tarvainen A and Valpola H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2017, 1195–1204.
[64] Liu YC, Ma CY, He Z, et al. Unbiased teacher for semi-supervised object detection. In: 9th International Conference on Learning Representations (ICLR), 2021.
[65] Kirillov A, Mintun E, Ravi N, et al. Segment anything, arXiv preprint arXiv:2304.02643, 2023.
[66] Tsipras D, Santurkar S, Engstrom L, et al. Robustness may be at odds with accuracy. In: 7th International Conference on Learning Representations (ICLR), 2019.
[67] Ilyas A, Santurkar S, Tsipras D, et al. Adversarial examples are not bugs, they are features. In: Annual Conference on Neural Information Processing Systems (NeurIPS), 2019, 125–136.
[68] Kerbl B, Kopanas G, Leimkühler T, et al. 3D Gaussian splatting for real-time radiance field rendering. ACM Trans Graph 2023; **42**: 1–14.

Wenwen Chen studied at the University of Electronic Science and Technology of China as a bachelor and worked on a double master's project at Tongji University and the Technical University of Munich. She focused on semi-supervised learning and adversarial deep learning.

Jun Yan is a Ph.D. candidate in the Department of Information and Communication Engineering, Tongji University, China. His research interest is the theory of adversarial machine learning and the design of new deep learning models in autonomous vehicles.

Weiquan Huang is a Ph.D. candidate in the Department of Computer Science, Tongji University, China. He received his bachelor and master degree at Tongji University. His research interest lies in multimodal learning.

Wancheng Ge is a professor in the Department of Information and Communication Engineering, Tongji University, China. His research interest is on wireless communication and arti cial intelligence.

Huaping Liu received the B.S. and M.S. degrees in electrical engineering from Nanjing University of Posts and Telecommunications, Nanjing, China, in 1987 and 1990, respectively, and the Ph.D. degree in electrical engineering from New Jersey Institute of Technology, Newark, NJ, USA, in 1997. From July 1997 to August 2001, he was with Lucent Technologies, Whippany, NJ, USA. In September 2001, he joined the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA, where he has been a Full Professor since 2011. His research interests include modulation and detection, multiple-antenna techniques, and localization systems.

Huilin Yin received the Ph.D. degree in control theory and control engineering from Tongji University, China, in 2006. She is currently the chaired professor of TUEV SUED Chair with the Electronic and Information Engineering College, Tongji University. She received the M.S. double-degree from Tongji University and Technical University of Munich, Germany. Her research interests include environment perception of intelligent vehicles and safety for autonomous driving.