Learning Slimming SAR Ship Object Detector Through Network Pruning and Knowledge Distillation

Shiqi Chen, Ronghui Zhan, Wei Wang, and Jun Zhang, *Member, IEEE*

***Abstract*—The deployment of deep convolutional neural networks (CNNs) in synthetic aperture radar (SAR) ship real-time detection is largely hindered by huge computational cost. In this article, we propose a novel learning scheme for training a lightweight ship detector called Tiny YOLO-Lite, which simultaneously 1) reduces the model storage size; 2) decreases the floating point operations (FLOPs) calculation; and 3) guarantees the high accuracy with faster speed. This is achieved by self-designed backbone structure and network pruning, which enforces channel-level sparsity in the backbone network and yields a compact model. In addition, knowledge distillation is also applied to make up for the performance decline caused by network pruning. Hereinto, we propose to let small student model mimic cumbersome teacher's output to achieve improved generalization. Rather than applying vanilla full feature imitation, we redefine the distilled knowledge as the inter-relationship between different levels of feature maps and then transfer it from the large network to a smaller one. On account that the detectors should focus more on the salient regions containing ships while background interference is overwhelming, a novel attention mechanism is designed and then attached to the distilled feature for enhanced representation. Finally, extensive experiments are conducted on SSDD, HRSID, and two large-scene SAR images to verify the effectiveness of the thinner SAR ship object detector in comparison of with other CNN-based algorithms. The detection results demonstrate that the proposed detector can achieve lighter architecture with 2.8-M model size, more efficient inference (>200 fps) with low computation cost, and more accurate prediction with knowledge transfer strategy.**

***Index Terms*—Attention mechanism, feature imitation, knowledge distillation (KD), lightweight synthetic aperture radar (SAR) ship detector, network pruning.**

I. Introduction

NOWADAYS, deep convolutional neural networks (CNNs) have gained much attention in the application of synthetic aperture radar (SAR) field, such as automatic target recognition [1], urban interpretation [2], marine surveillance [3], and so on [4]. Among them, ship detection in SAR images has been widely studied due to its indispensable role in military and civil fields. Benefiting from the increase of high-quality satellite data and CNN's powerful capability of feature representation, numerous deep learning-based methods have been proposed in SAR ship detection.

Recently, many superior type of research have refreshed results on general object detection in terms of both region proposal based methods [5]–[7] and regression-based methods [8]–[10], which have also justified the effectiveness on SAR object detection. For instance, both Cui *et al.* [11] and Chen *et al.* [12] have integrated attention mechanism into the feature pyramid network for detecting multiscale SAR ships. Cui *et al.* [13] also proposed an anchor-free detector and introduced the spatial shuffle-group enhance attention module for detection in large-scale SAR images. Besides, Wei *et al.* [14] proposed a modified high-resolution ship detection network (HR-SDNet) and adopted cascade structure for precise and robust ship detection. Nevertheless, the above methods all have huge backbone network and large model size, thus leading to decrease in inference speed and difficulty in model deployment of spaceborne platforms. Therefore, how to reduce the size of trainable model parameters without notably sacrificing detection performance becomes an urgent issue to be tackled.

Specifically, some attempts can be traced in YOLO series models [8], [15], [16]. YOLT [17] combined two detection models based on YOLO, which are trained on different image scales to achieve better detection results for objects of disparate sizes. Chang *et al.* [18] designed a YOLOv2-reduced network for real-time SAR ship detection, which lowers computational complexity but still lacks theoretical explanation. Zhang *et al.* [19] devised a depthwise separable CNN (DS-CNN) and further merged it into YOLOv3 framework to speed up detection. Although multiscale detection, feature concatenation, and anchor box mechanism are adopted to improve detection accuracy, their model still involves partial heavy convolution operators, declining the detection speed. They further proposed HyperLi-Net [20] by integrating five internal mechanisms and five external modules, which is more lightweight and accurate. While these approaches are capable of simplifying network structure, they may neglect that the balance between detection accuracy and inference speed should be guaranteed. Substantial efforts have been devoted to the speedup and compression of CNNs for deploying a deep model on resource-limited devices.

Manuscript received September 20, 2020; revised November 18, 2020; accepted November 26, 2020. Date of publication December 7, 2020; date of current version January 6, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61901500 and in part by the Natural Science Foundation of Hunan Province under Grant 2020JJ5674. *(Corresponding author: Ronghui Zhan.)*

The authors are with the National Key Laboratory of Science and Technology on Automatic Target Recognition, College of Electronic Science, National University of Defense Technology, Changsha 410073, China (e-mail: 337035302@qq.com; zhanrh@nudt.edu.cn; wangwei_nudt@hotmail.com; zhangjun@nudt.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2020.3041783

At present, model compression [21] is mainly divided into four types, including pruning, quantization, low-rank factorization, and knowledge distillation (KD). Model pruning [22] is a method to trim the network by cutting off redundant connections between weights or neurons of adjacent layers, and fine-tuning is subsequently attached to the pruned model for improving the performance. Then, quantization converts the weights of the neural network from 32-b floating point to low-bit representation, thereby greatly reducing the size of the neural network model [23]. The next one is low-rank factorization, which uses matrix or tensor decomposition techniques to estimate and decompose the convolution kernel and thereby reducing network parameters. First, introduced by Hinton *et al.* [24], KD exploits the knowledge extracted from the high-performance teacher model as guidance for improving the accuracy of a small student model. Moreover, it can be used as a new training strategy to elevate the network performance. Though recent works such as [25]–[27] have proved its effectiveness on object detection, there exist few applications adaptable to the direct compression of large object detectors.

Therefore, in this article, we proposed to learn a simple but efficient SAR ship detector through the combination of network pruning and KD, which would facilitate the deployment of SAR ship detection in practical applications and thereby improve the model generalization ability. Here, a well-performed detector YOLOv3 is applied as our reference detection framework for its remarkable inference speed but with the following modifications. To strengthen the feature representation ability of targets with large aspect ratio, an asymmetric convolution module (ACM) is incorporated into the backbone network. Then, a densely connected feature pyramid network is devised to fuse features of different layers and enhance the semantic information for each scale feature layer, which further boost the performance of multiscale ship detection especially for small targets. Next, motivated by the network slimming algorithm proposed by Liu *et al.* [28], we prune the backbone of the modified detector to obtain a compact feature extraction model.

To further improve the detection performance of pruned models, we propose feature map inter-relationship guided KD, a novel model compression approach for efficiently training a slim but effective ship detector. Specifically, the original novel detector denoted as DC-ACM YOLOv3 with complicated backbone is defined as the teacher, while the target slim detector with rather small backbone obtained by network slimming is defined as the student. As a higher level of vision task, object detection predicts both classification and localization information of each instance; so distilling the soft target of classification branch does not lead to promising results. Moreover, the extreme imbalance between foreground and background instances makes it intractable to capture the most representative knowledge from the teacher network. It is intuitive to directly imitate the backbone features of the teacher model in object detection; however, blindly transferring all learned features from the teacher model brings unnecessary computation of unrelated features and further restricts the performance gain of the student model. The problem becomes tougher especially when detecting inshore ships in SAR images since the interference of backscattering points are stronger and many complex surroundings may cause more false alarms. To address this dilemma, a novel graph-based feature KD strategy is devised to perform imitations at the most discriminative features from different levels. Meanwhile, the feature inter-relationship graph is also constructed for generating the sufficient knowledge so as to inspect the generalization ability over the teacher model.

Focusing on how to build a slim SAR real-time ship detector with high accuracy, the main contributions of this article can be summarized as follows.

1) For paying more attention to objects with multishapes and multiscales, a novel backbone network is designed to enrich features with more semantics and diverse shapes of receptive fields. With less computation burden and more powerful feature representation, the performance of YOLOv3 on small scale and multishape targets can be elevated.

2) Different from the most network pruning approaches which are attached with fine-tuning stage to make up for performance decline, we present a novel KD strategy for model deployment on spaceborne or missile-borne platform, which eliminates the temporary performance degradation and further boost the detection performance.

3) Considering that the selection of transferred knowledge should be more discriminative for object detection, we derive a feature inter-relationship graph-based distillation framework to adapt the feature interaction from different levels. Additionally, a novel attention-based block is proposed to strengthen distilled features according to the object-related regions, which largely eliminates the interference of complex cluttered background.

II. Related Work

Here, we specifically discuss the works of literature that are closely related to our work, including network pruning and KD methods.

1) Network Pruning: As a feasible method of removing the redundant branches in the networks, network pruning can be categorized into weight pruning and structured pruning. The first type of method prunes the unimportant connections with small weights in the trained networks [29]; however, the speedup can only be achieved by dedicated sparse matrix operation libraries, and it is unfriendly to implementation on hardware platforms. In comparison, structured pruning [30] is more likely to reach a well-performed tradeoff between flexibility and ease of implementation. Recent works [31] remove channels with small incoming weights entirely instead of individual weights and then fine-tune the network to regain accuracy. These methods [28], [32], [33] also utilize sparsity regularization on different levels of structures such as filters, channels, or layers in CNNs. Among them, channel pruning based on scaling factors in batch normalization (BN) layers is the most effective and widely used method [28], which is applied to prune CNN-based network in the image classification task. However, the ways of measuring the importance of channels' lack of universality and flexibility, thus, seems to be less practical in SAR ship detection applications. Furthermore, most of the existing pruning methods mainly focus on classification, while object detection requires not only semantic information but also localization information.

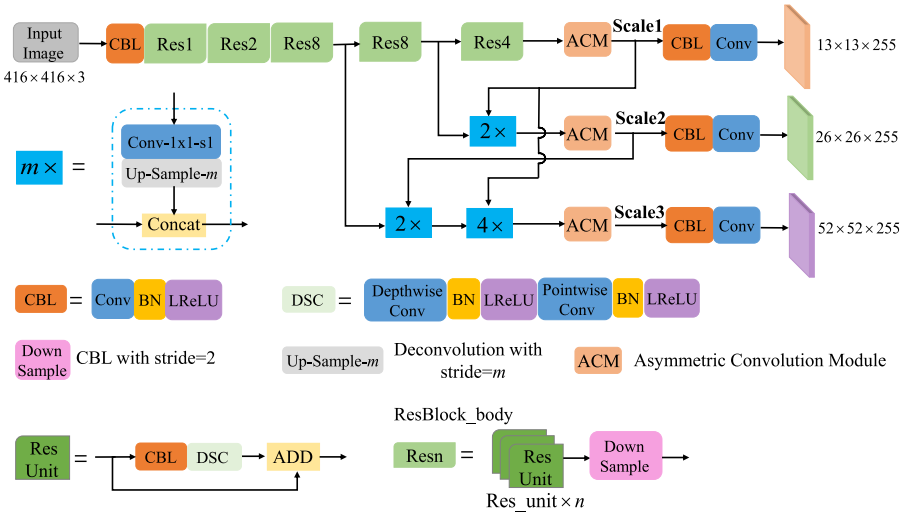


Fig. 1. Architecture of the self-designed teacher network in YOLOv3, which includes the feature extraction network (called as Darknet-ARes) and the dense multiscale prediction module. The colored submodules such as CBL, DSC, and Res Unit are illustrated by more detailed compositions as shown in the bottom part of the figure.

As for object detection, Shih *et al.* [34] use the pruning method proposed in [28] to prune the backbone of the two-stage network. Referring to the effective channel pruning method [35], some approaches [36], [37] apply the global and local pruning scheme to obtain a slim network model with good performance. Both layerwise and channelwise pruning are utilized in [38] to lighten the model parameters.

2) Knowledge Distillation: One promising direction of mitigating the high computational burden in deep CNNs is to transfer knowledge in the cumbersome model into a small model. KD methods assume the knowledge as a mapping function learned from inputs to outputs and then transfer it by training the student model under the supervision of the teacher's outputs. KD is originally used in the task of image classification [24], where the student is supervised by both softened labels and ground truth labels simultaneously. Romero *et al.* [25] propose FitNet, which distills a thinner student model by imitating both the softened class scores and intermediate feature representations of the teacher model. Following this, some subsequent works [39], [40] have also tried to mimic the intermediate layers of the teacher network to that of the student network. Zagoruyko and Komodakis [41] devise attention transfer on the basis of attention maps to improve the performance of the student network. Yim *et al.* [42] even distill the knowledge of pairwise similarity maps between neighboring layers. To summarize, current distillation frameworks either lack particular design for object detectors or fail to select the most representative part for distillation. Different from the above works, our distillation framework is deliberately designed for SAR ship detection, which considers the feature inter-relationship between different levels and avoids the interference of less informative background area.

III. Proposed Methodology

In this work, we develop a simple-to-implement SAR ship detection method through network pruning and make a further step in detection accuracy via a feature inter-relationship graph-guided KD strategy. First, we devise a novel detector DC-ACM YOLOv3 as a promising reference detector, on which our method is established for real-time SAR ship detection. Then, DC-ACM YOLOv3 is initially trained with channelwise sparsity regularization. Channel pruning is performed on convolutional layers to remove the less informative feature channels and, thus, form an object detector with less model volume and computational costs. The original burdensome model and the corresponding pruned one are selected as the teacher and the student model, respectively. Finally, KD techniques are introduced to help the training of the slim student object detector. Each module would be elaborated in the following sections.

A. Pipeline of the Modified YOLO-Based Detector

This section first reviews the structure of YOLOv3 and then describes the proposed network architecture named densely connected and ACM-assisted YOLOv3 (DC-ACM YOLOv3). The pipeline of the proposed detector is indicated in Fig. 1. Specifically, YOLOv3 down-samples the input image by five times corresponding to five stages in the backbone part and selects three specific scales of feature layers for final predictions. Three detection heads are separately built on top of these feature maps and are responsible for detecting objects with different sizes. More details about YOLOv3 can be traced in [16].

Although the backbone Darknet-53 exhibits good feature extraction capability in YOLOv3, the performance is restricted under the application scenario of SAR ship detection, in which targets are with multiscales and multiaspect ratios. Furthermore, the balance between accuracy and speed is also demanding.

Dealing with the above limitations, a more powerful and lightweight backbone is designed to extract more representative features, thus ensuring the precision of small ship detection and improving the detection efficiency. A more optimal feature extraction network named Darknet-ARes, where A means

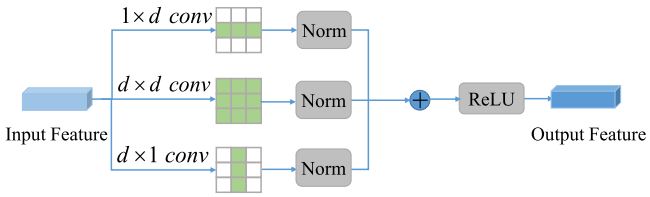


Fig. 2. Architecture of the asymmetric convolution module.

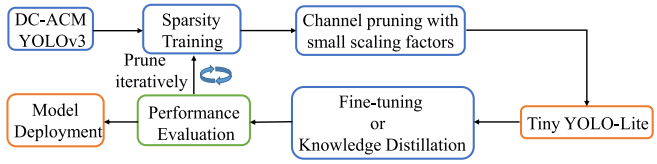


Fig. 3. Channelwise sparsity training and network pruning for Tiny YOLO-Lite.

asymmetric, is designed to partly reduce the computational complexity and parameter size of the network.

To maintain the multiscale object detection capability of the YOLOv3 network, the modified structure retains the framework of five stages in the backbone with three output branches. The original backbone Darknet-53 in YOLOv3 uses five residual modules (Res$n$ where n means the number of Res Unit) for feature extraction, while Darknet-ARes is streamlined by replacing the second convolution module in residual bottlenecks with 3×3 depthwise separable convolution (DSC), as shown in Fig. 1.

FPN-style [43] architecture is introduced in YOLOv3 to make the deeper network contain both rich semantic features and abundant spatial features, which has improved the accuracy of small target detection, while exhibiting poor performance on medium- or large-sized objects. To further improve the capability of multiscale detection, we draw the inspiration of dense connection in [44] and integrate a novel feature fusion scheme into the last three stages of Darknet-ARes, thus achieving multiscale dense prediction.

For each feature level in the backbone, instead of merely fusing two successive feature levels, we merge features from other levels as well. The blue modules ($m\times$) in Fig. 1 illustrate the dense prediction scheme, in which the three-scale feature maps used for prediction head are densely connected after being up-sampled. Among them, $m\times$ represents up-sampling the features with stride m and the up-sample layer is implemented by a deconvolution operation with stride m. Our model integrates the outputs of three different scales by up-sampling and concatenation through top-down pathway and dense connections. Therefore, the low-resolution but semantically rich features are adequately fused with high-resolution but locationally aware features. In this way, each level is able to leverage both high-level semantic and low-level fine-grained features, thereby being beneficial for the detection of multiscale targets.

For the purpose of more efficiently and effectively capturing long-range dependencies, He *et al.* [45] exploited spatial pyramid pooling (SPP) at the end of the backbone network, thus to enlarge the receptive fields of backbone features and to aggregate the most informative context features. However, the spatial pyramid module only processes square receptive fields and the performance would degenerate when detecting ships with large aspect ratios. In this article, the ACM is designed to enrich different shapes of receptive fields with minimal computation costs. As can be seen from Fig. 2, two additional convolution branches with kernel $d \times 1$ and $1 \times d$ are added in parallel with $d \times d$ convolutional layer to enrich receptive fields. The proposed ACM is attached at the end of the feature map processed by dense connection, thus enhancing the representation capability of features with rectangular receptive fields.

B. Sparsity Training Channel Pruning

Due to the limited memory and computing power of embedded devices, real-time ship detection of in-orbit missile-borne or air-borne SAR is extremely challenging. Here, we follow the pruning scheme illustrated below and construct a more lightweight and efficient SAR ship detector through channel pruning of convolutional layers in DC-ACM YOLOv3 network model. To this end, we obtain Tiny YOLO-Lite by following the procedure depicted in Fig. 3.

Aiming at providing a simple but effective scheme to achieve channel-level sparsity in CNNs, sparsity regularization is enforced in the training objective. First, a scaling factor for each channel is assigned to represent the channel's importance. Then, channelwise sparsity training is performed to discriminate important channels from unimportant ones. Finally, we set a global threshold to determine the channels to be removed. Next, the selection of scaling factor and the procedure of sparsity training will be illustrated.

1) Scale Factors in BN Layers: As a powerful operator which ensures fast convergence and improved generalization, BN layer [46] is usually connected after a convolutional layer in most conventional network structures. In particular, BN layer normalizes the feature activations using mini-batch statistics, which can be formulated as follows:

$$z_{\text{out}} = \gamma \frac{z_{\text{in}} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta \tag{1}$$

where μ and σ are the mean and standard deviation value of input features in a mini-batch and γ and β denote trainable scaling factor and bias parameter.

Following the common practice, the BN layers in our backbone network are inserted after the convolutional layers with trainable factors, of which the scaling factors are adopted as indicators of channel importance.

Specifically, some special connections between layers are required to be treated carefully. During pruning process, we first construct a pruning mask for all the convolutional layers according to the global threshold $\hat{\gamma}$ and the local safety threshold θ. For a route layer, the pruning masks of its input layers are concatenated in sequence and the output mask is taken as its pruning mask. For shortcut layer which is similar with residual bottleneck in ResNet, to match the channel numbers of layers connected by this layer, we iterate through the pruning mask of all connected layers and perform OR operation on these pruning

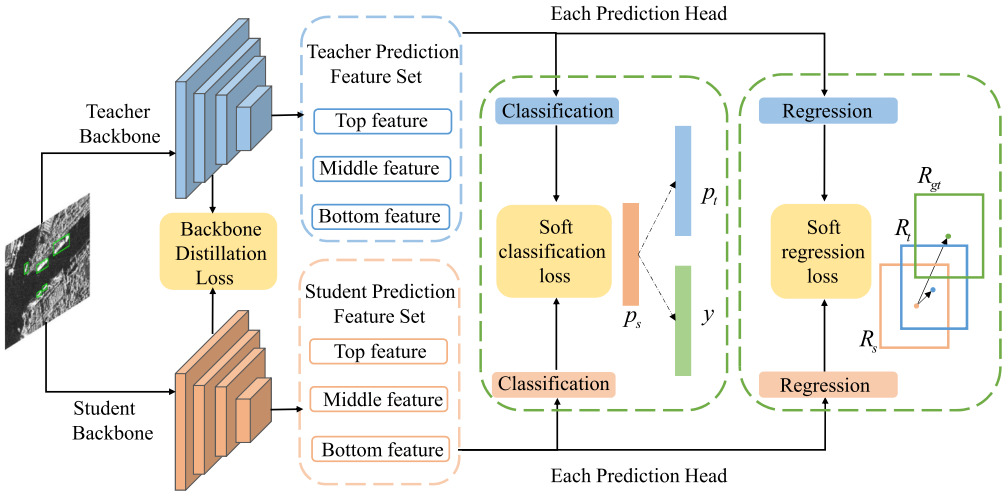


Fig. 4. Overview of the proposed ship detection distillation framework. The distillation method consists of three modules: distillation of backbone feature, classification head, and bounding box regression head for each prediction level, as depicted in the yellow parts. Blue parts denote the schematic diagram of DC-ACM YOLOv3 and they remain unchanged during distillation process, while orange parts are the student network supervised by the teacher network.

masks to generate the final pruning mask for these connected layers.

2) Training With Channelwise Sparsity Constraint: During the process of network slimming, we jointly perform training of network weights and scaling factors, with L1 regularization imposed on the latter in channelwise scheme. Specifically, the training objective of channel pruning network is given by

$$L = \text{loss}_{\text{detector}} + \lambda \sum_{\gamma \in \Gamma} f(\gamma) \tag{2}$$

where the first term denotes the normal training loss of a CNN detector, $f(\gamma) = |\gamma|$ represents L1 normalization which is widely applied in achieving sparsity, and λ balances the two terms. In our implementation, subgradient method was utilized to optimize the nonsmooth L1 penalty term as Liu *et al.* [28] did. After channelwise sparsity training, a global threshold $\hat{\gamma}$ is set to pick out the feature channels to be pruned. Given the prune ratio n, $\hat{\gamma}$ can be denoted as the nth percentile of all γ values. Besides, a local safety threshold θ is also defined in a layerwise manner. We prune the feature channels whose scaling factors are less than the minimum value of $\hat{\gamma}$ and θ. In detail, the channels with near-zero scaling factors are pruned by removing all the input and output connections and corresponding weights.

C. Knowledge Distillation Strategy

After training with channelwise sparsity-guided regularization, we obtain a narrow network in which most scaling factors are clipped to zero. Although network pruning helps obtain a lightweight model and fine-tuning helps make up for some decline in the accuracy, it does not offer satisfactory performance on the pruned model and the upper limit of the performance gain is largely dependent on that of the original model. In this article, a novel KD strategy focusing on feature inter-relationship, called FIR-KD, is adopted to achieve higher detection accuracy than the original unpruned network. To introduce the FIR-KD scheme in detail, this section first illustrates the idea of FIR-KD by analyzing the weakness of existing methods. In the second part, the feature inter-relationship established in the graph structure (FIR-Graph) and the attention-enhanced mechanism in it are described.

1) Overall Framework and Ideas of FIR-KD: The overall framework of the proposed KD method is demonstrated in Fig. 4. The upper blue network represents the teacher network, while the lower orange one denotes the student network. Three modules called distillation of backbone feature, classification head, and bounding box regression head constitute the proposed distillation framework, and three different levels of knowledge are adaptively transferred from cumbersome network to the lightweight one.

Existing works tried knowledge transfer of extracted features by adding full feature imitation, but we find this kind of imitation will bring minor performance gain for student model and it is ambiguous how to deal with feature dimensions inconsistency when performing distillation. In this work, we developed a simple but effective feature imitation method utilizing feature inter-relationship KD, called FIR-KD, which can provide more sufficient and general information about the distribution and mutual relationship between different feature levels.

Furthermore, directly doing full feature imitation will inevitably introduce large amount of noise from irrelevant areas, especially for SAR ship detection where the cluttered background is diversified and overwhelming. Inspired by the attention mechanism which is proven to be helpful for boosting the performance of SAR ship detection [11], [47], we embed a novel attention-based block to guide the feature representation used for knowledge transfer, which not only spotlights the pivotal pixels but also restrains the uninformative ones.

2) Details of Attention-Based FIR-Graph: In this part, the feature affinity graph called FAG is first constructed for representing adequate knowledge between different levels of feature maps. Subsequently, each feature in the graph structure is strengthened by attention module, namely strip pooling based attention module (SPAM). Finally, the overall loss is formulated

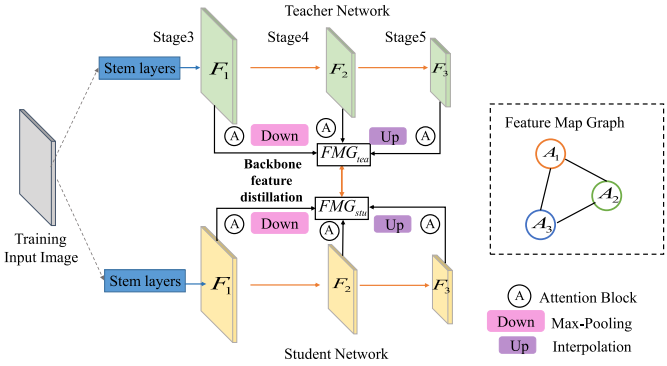


Fig. 5. Structure of FIR-Graph.

on the basis of general detection loss functions and serves as the supervisor for training the slim student network.

Given a training instance x_i, the feature inter-relationship graph will be calculated in both the teacher and the student network. The graph structure can capture the similarities between features of different scales, which are then taken as the high-level backbone feature knowledge to be distilled from the teacher network.

Let $f_l(x_i)$ denote the features of x_i extracted from lth stage, which means the intermediate feature maps of three levels in the backbone network. The feature inter-relationships are formulated as the adjacent matrix of the features from different scales, referring to as A_l. An example of FAG is shown in the right part of Fig. 5. Then a feature affinity graph of the ith training sample denoted as FAG_i, which reflects the backbone feature space, is expressed as

$$\text{FAG}_i = (V_n, E_n) = (f_l(x_i), A_l) \tag{3}$$

where V_n is a set of nodes, each of which denotes the feature distributions of each predicted level. In this case, $n = 3$ since only output feature maps of three scales constitute the whole graph. Each pair of nodes is connected by the edge set E_n, of which each element manifests the feature relationship matrix, A_l. We obtain the relationship matrix by calculating the Euclidean distance between the features from two independent layers, which can be formulated as follows:

$$A_l(m,n) = \|f_m(x_i) - f_n(x_i)\|_2^2, m, n = 1, 2 \ldots, l. \tag{4}$$

Next, the attention module is introduced into the feature affinity graph to boost the salient features and suppress the background clutters. Based on that, a finer representation of features with higher potential to reflect pivotal pixels can be generated.

The concept of strip pooling is first presented in [48], which utilized a band shape pooling window to perform pooling along the horizontal or the vertical dimension. Owing to the long and narrow kernel shape, the object regions with banded shape are easier to be described and long-range dependencies between discretely distributed regions can be built. Then, we integrate both horizontal and vertical strip pooling operations to the spatial attention block [49]. Thus, an attention-guided feature

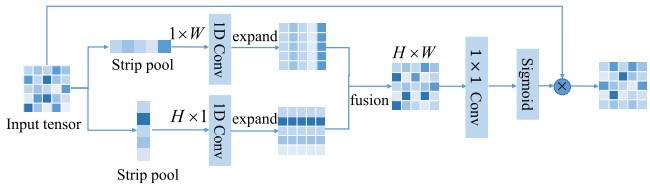


Fig. 6. Schematic illustration of strip pooling based attention module.

inter-relationship graph is built, which serves as the final form of feature distillation.

Once features are extracted by the backbone under three stages with $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$ of the input image size, those features are fed into a *conv* layer and SPAM sequentially. The diagram of SPAM is shown in Fig. 6, in which the *conv* layer is used to unify the channel numbers for each feature level. To exploit the spatial attention map $A_{\text{sam}}(X_i) \in \mathbb{R}^{W_f \times H_f \times 1}$ as the feature importance descriptor given input feature map $X_i \in \mathbb{R}^{W_f \times H_f \times C}$, the SPAM first generates two pooled features $P_{\text{hori}}, P_{\text{ver}}$ along the channel axis in two parallel pathways. In order to keep the same spatial size of two pooled features from horizontal and vertical direction, they are fed into a 1-D convolutional layer with kernel size 3 for adjusting the current location and its adjacent features. Then, it is followed by a convolutional layer and normalized by the sigmoid function. The computation process can be computed as follows:

$$A_{\text{sam}}(X_i) = \sigma(f^{1\times1}(P_{\text{hori}} \circ P_{\text{ver}})) \tag{5}$$

where σ denotes the sigmoid function and $f^{1\times1}$ is a 1×1 convolution. In our implementation, $\circ$ denotes the elementwise addition. Finally, the attention-guided feature map X_{sam} is obtained by

$$X_{\text{sam}} = A_{\text{sam}}(X_i) \otimes (X_i) \tag{6}$$

where $\otimes$ refers to elementwise multiplication operation. For DC-ACM-YOLOv3, the output of the last three modified residual blocks is denoted as feature maps $\{F1, F2, F3\}$ and this makes up a feature pyramid. Since feature maps in the pyramid are not of the same resolution, we have to normalize the feature maps to the same resolution, which is then utilized as the input of each vertex of FAG. Therefore, we employ the max-pooling and linear interpolation to simultaneously adjust the resolution and achieve the preservation of feature information.

Here, given the multilevel feature sets $\{F_1, F_2, F_3\}$, we normalize them to the same resolution as F_2. That is, F_1 and F_3 are down-sampled and up-sampled to the same scale as F_2, resulting in F_1^*, F_2^*, and F_3^*. Thus, the formulation of the affinity feature map distillation can be redefined as

$$A_l(m,n) = \|\text{Att}(F_m^*) - \text{Att}(F_n^*)\|_2^2, m, n = 1, 2 \ldots, l \tag{7}$$

where Att(•) means the attention block attached to feature maps from different levels. Since the attention module is only attached when training the student network, the parameters of this part are trained from scratch.

In the above process, the feature importance of each position in the output tensor can be reflected by feature reweighting,

which preserves the salient features and suppresses the background noise. Note that the attention module only participates in enhancing features to be mimicked when training the student network. Compared to full feature imitation, the attention-guided feature KD considers the strengthened feature of object-related regions instead of the whole feature map, avoiding most unnecessary information to be mimicked from the teacher model.

The whole structure is optimized with different imitation losses, in which the classification and regression loss keep the same format as that in normal detection training loss. As shown in Fig. 4, we add the supervision on the intermediate feature map of backbone network, classification head, and bounding box regression head. Integrating the above three distillation terms generates the overall training targets of the student model, which can be defined as follows:

$$L_{\text{all}} = \beta L_{A_l} + L_{\text{cls}} + L_{\text{reg}} \tag{8}$$

where β is the balance parameter of affinity feature map distillation and L_{A_l}, L_{cls}, and L_{reg} constitute the distillation part from three aspects.

We will show in the experimental section that such feature transfers are beneficial to improve the performance of student model.

IV. Experiments and Discussions

Our experiments are implemented based on the deep learning framework PyTorch and carried out on a PC with Intel (R) i7-8700 K CPU, NVIDIA GTX-1080Ti GPU with CUDA 10.0, and cuDNN v7. The PC operating system is Ubuntu 16.04.

In the following experiments, we set IOU $= 0.5$ as the detection threshold, which means if the overlap between a predicted bounding box and a ground truth is higher than 0.5, then the targets in this bounding box are successfully detected. For generating the final prediction results, the objectiveness confidence and nonmaximum suppression threshold are set as 0.4 and 0.5, respectively.

A. Datasets

In order to verify the effectiveness and superiority of the proposed methods, this article conducts experiments and analyses on two SAR ship datasets: SSDD [50] and AIR-SARShip-1.0 [51]. The SSDD dataset contains SAR images with different resolutions, polarization modes, or sensor types, or under different sea conditions, scenarios, and so on. Additionally, ships in SSDD also have many sizes, from the smallest scale of 7×7 to the largest scale of 211×298, which is convincing to evaluate the multiscale detection performance of ship detectors under complex scenarios. It contains 1160 images and 2456 ship instances in total, where the proportion of the training, validation, and test sets are split with the ratio of 7:2:1. The ablation studies are implemented on this dataset to identify the contributions of each improvement. Besides, a high-resolution SAR dataset called HRSID [52] is also adopted for the visualization of detection results due to the characteristic of small and different shapes of objects in large detection scenes. Apart from this, we apply AIR-SARShip-1.0 dataset to evaluate the generalization of our method. It comprises 31 single-polarized and high-resolution SAR images acquired from GF-3 satellite. Considering the restricted GPU memory, each large-scale image is cropped into 512×512 image blocks with the overlap of 256 pixels for testing.

B. Model Definition

1) Baseline Models: Several models based on YOLO-series, i.e., YOLOv3, YOLOv3-tiny, YOLOv3-mobilenet, and YOLOv3-SPP, are selected as baseline models. YOLOv3-tiny and YOLOv3-mobilenet are YOLOv3-based methods with different backbones, both of which are lightweight and are much faster with less accuracy than YOLOv3. YOLOv3-SPP represents a revised version of YOLOv3, which inserts one SPP module in front of the first detection header.

2) DC-ACM YOLOv3: The self-designed detector based on YOLOv3 is implemented by incorporating asymmetric convolutional block to the end of the fused feature maps used for dense prediction. The main difference of our model from the baseline model lies in a novel backbone network design and the capability of multiscale prediction realized by dense connection.

3) Tiny YOLO-Lite: Implemented by channelwise pruning and KD, Tiny YOLO-Lite denotes the final lightweight model with slim network structure and high detection accuracy. After channelwise pruning, several tiny models are obtained by setting different pruning ratios, and the local safety threshold θ is empirically set as 90th percentile of all $|\gamma|$ in each layer to keep at least 10% of channels unpruned in a single layer. Finally, KD strategy further facilitates the performance of pruned model and results in the final model called Tiny YOLO-Lite.

C. Training details

1) Normal Training: Following the default configuration in Darknet-53 [16], we train the YOLOv3-based methods using stochastic gradient descent with the momentum of 0.9 and weight decay of 0.0005. The initial learning rate is set as 0.001, which is altered by a cosine annealing scheduler during sinusoid training. For both dataset, we train our models for a total of 100 epochs, with a batch size of 8. For the comparison methods, we set the size of input image as 416 for YOLOv3-tiny and 608 for YOLOv3 and YOLOv3-SPP.

2) Sparsity Training: Since more adequate training is required to maintain the sparsity of the pruned network, we perform sparse training of the pruned DC-ACM YOLOv3 for 200 epochs. When training with channelwise sparse regularization, the penalty factor λ, which controls the tradeoff between normal detection loss and sparsity regularization loss, is set as different values for achieving the best performance of sparsity. All the remaining settings are kept the same as in normal training.

3) Fine-Tuning or KD Training: After the pruning process, we obtain a narrower and more compact model, which is then fine-tuned or trained again with the assistance of KD strategy. Both of them use the same optimization setting as in training.

Due to the good sparsity of the model generated in the sparsity training stage, our pruned model can converge fast and achieve

even better results than the original model after introducing KD strategy.

D. Evaluation Metrics

In this article, the commonly used evaluation indicators such as average precision (AP), precision–recall curve, and F1 score are utilized to reflect the holistic performance of SAR ship detector. Plotted by calculating the precision–recall pair under different confidence thresholds, the PR curve reveals the relationship between precision and recall, and the larger the area it covers, the better the detection results can be obtained. The precision indicates the correctness of the detection results, computed by the fraction of positive samples that are correctly identified. Whereas, the recall reveals the completeness of the detection results, calculated by the ratio of true positives in all the detection results. These two metrics can be computed as follows:

$$\text{Recall} = \frac{N_{\text{TP}}}{N_{\text{TP}} + N_{\text{FN}}} \tag{9}$$

$$\text{Precision} = \frac{N_{\text{TP}}}{N_{\text{TP}} + N_{\text{FP}}} \tag{10}$$

where N_{TP}, N_{FP}, and N_{FN} denote the number of the correctly detected ships, the false alarms, and the undetected ship targets. F1 which combines the precision and recall metrics into a single measurement is formulated as follows:

$$F1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{11}$$

The F1 score measures the detection performance of the model with a single threshold, while the AP metric evaluates the integral detection performance under a set of threshold values. It is measured by the area under the PR curve, which can be defined as follows:

$$\text{AP} = \int_0^1 p(r)dr \tag{12}$$

where p and r denote the precision and recall, respectively. In addition, to validate the effectiveness of Tiny YOLO-Lite for real-time SAR application, other indicators such as model volume, parameter size, floating point operations (FLOPs), and inference time ($T \sim ms$), which are also represented by frames per second (FPS), are introduced for evaluation.

E. Qualitative and Quantitative Analyses of Results

1) Backbone Impact and YOLOv3 Baseline: We first investigate the impact of two novel submodules in DC-ACM YOLOv3 by successively adding them to the baseline backbone network. Here, we view the original backbone structure Darknet-53 without the self-designed modules as our baseline and evaluate the following design choices: 1) densely connected pyramid (DCP)—the proposed method that rebuilds the feature pyramid with dense connections and we denote it as DCP; 2) ACM—the proposed ACM which is attached at the output of the fused feature from different levels.

Fig. 7 shows the superiority of DCP in the detection of small targets compared with the typical feature pyramid structure. As shown in Fig. 7(b), several small-scale ships are neglected in both open sea area and inshore area. In contrast, the network with DCP exhibits a better performance when dealing with small chips. In terms of detecting inshore ships with small scale, it is apparent that some densely distributed ships are easier to be detected under DCP structure. Seen from the fourth column in Fig. 7, when ships are arranged end to end on the shore, they can be well distinguished if DCP is introduced, while two neighboring ships are taken as one target in the baseline model. This is advantageous because each feature level is given a much richer information flow from all its bottom layers rather than only the adjacent layer. Note DCP has negligible extra consuming time and only costs 0.5 ms more time for an input image of size 416×416, making it have an ignorant impact on the speed of the whole network. Since multiscale feature fusion in the format of dense connection is beneficial for generating more discriminative features especially for small targets, DCP is employed as the auxiliary strategy to boost the performance of Darknet-ARes.

Furthermore, we also analyze some typical detection results to verify the effectiveness of ACM block. Various shapes of ships are ubiquitously located in the inland area, as demonstrated in Fig. 8. It is distinct that some missing detections exist in the inshore area, such as ships in the first and second columns of Fig. 8. This may be because the general convolutions with fixed kernel size show limited capability of extracting enough receptive fields for targets with relatively large aspect ratio. Besides, take ships in the last two columns of Fig. 8(b), for instance; although some parts of ships are involved in the predicted bounding boxes, the boundary areas are inadequately represented and object parts far from the center are not properly enclosed. We argue that this phenomenon lies in the insufficient semantic representation of the areas between objects and the complex surroundings such as docks, buildings, and other man-made facilities since the targets are prone to be affected by interference with similar scattering characteristics. Enabled with more appropriate feature representation brought by specific convolutional operations with asymmetric kernel sizes, more accurate localization which reflects the actual shape of ships can be obtained. Nevertheless, there is still some missing detections in terms of densely packed ships, which will be later discussed.

2) Ablation Experiments: To verify the importance of each component, we show a comprehensible ablation study in this section. The baseline of the ablation experiment is the proposed DC-ACM YOLOv3. We gradually add sparsity training, channelwise pruning and KD according to the order of network compression.

a) Effect of channel sparsity training: During the sparsity training, we visualize the scaling factors in all BN layers of our backbone network to trace the change in the distribution of scaling factors under different λ values. Although sparsity training is effective in reducing the scaling factors, finding the most optimal penalty factor λ in (2) is good for sparsification process and conducive to subsequent channel pruning. As can be observed in Fig. 9, the number of smaller scaling factors increases gradually, while the number of larger factors decreases with the training continues.

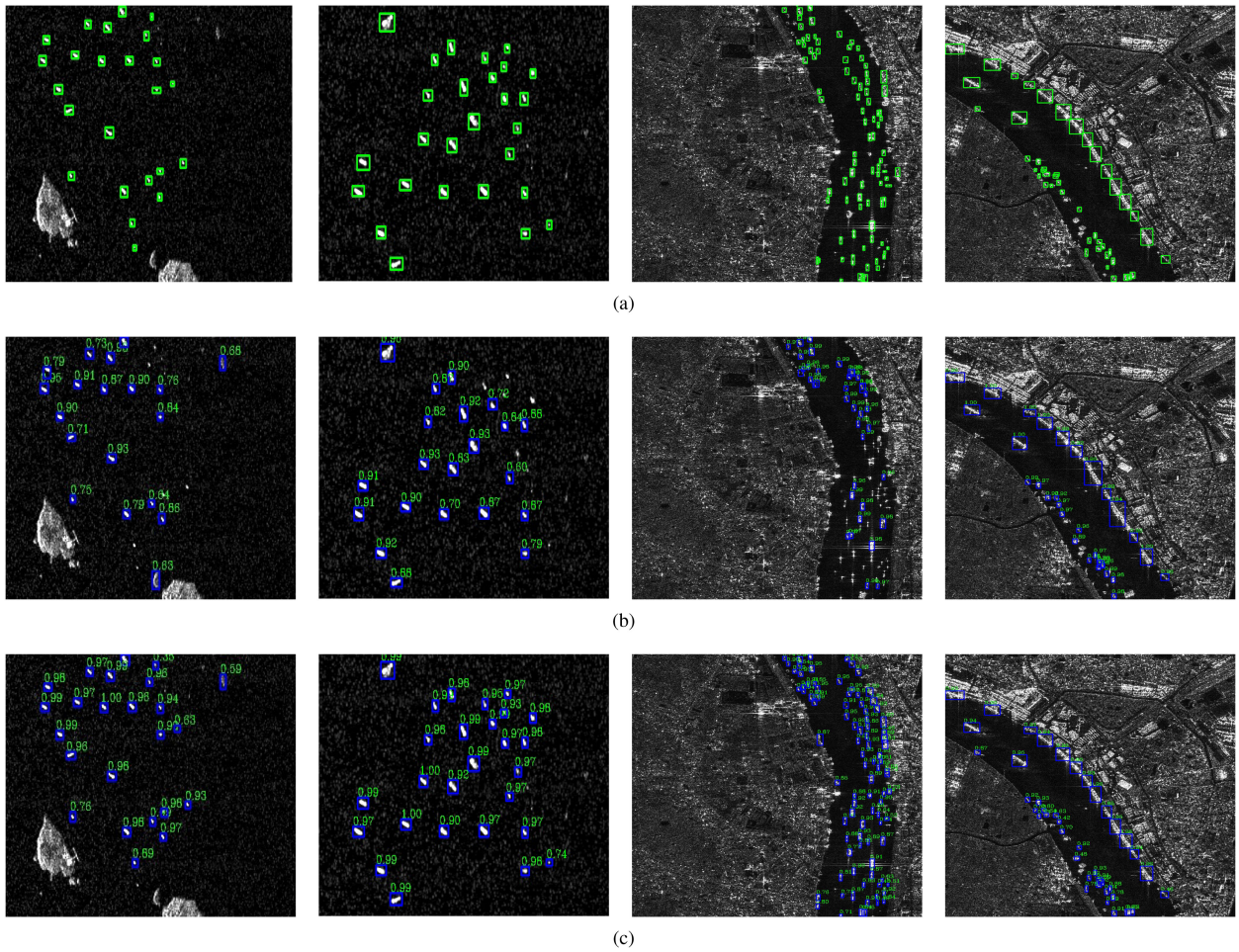


Fig. 7. Comparison of detection results under different methods on SSDD and HRSID chip images. (a) Ground truths. (b) Results under ordinary pyramidal structure. (c) Results of densely connected FPN structure.

With the increase of λ, the scaling factors become more clustered near zero. When $\lambda = 0$, there exists no sparsity regularization and the distribution is relatively flat. When $\lambda = 5 \times 10^{-4}$, the sparsity of scaling factors cannot be fully guaranteed and the model starts failing with underfitting. When $\lambda = 0.001$, nearly all scaling factors fall into the region near zero. This process can be interpreted as a feature selection happening in the intermediate layers of deep CNNs, where we only retain the channels with nontrivial scaling factors. After sparsity training, a slightly performance drop can be captured, which will be further compensated by subsequent training stage. In our implementation, the DC-ACM YOLOv3 is trained with $\lambda = 10^{-3}$ to achieve channel sparsity.

b) Effect of channelwise pruning: In our experiments, we obtain several models under different pruning ratios. A variety of global thresholds corresponding to the prune ratios ranging from 0.5 to 0.97 are adopted to perform aggressive channel pruning, where prune ratios denote the percentile of pruned channels. The relationships between AP versus parameter size and versus FLOPs are demonstrated in Fig. 10 for explicit comparison.

For instance, compared with the unpruned model DC-ACM YOLOv3, channelwise pruning with pruning ratios of 0.5, 0.8, and 0.95 actually decreases the model parameters size by 65.22%, 96.54%, and 99.18% and reduces FLOPs by 53.07%, 80.87%, and 92.65% (when the input size is 416×416). Under the moderate pruning ratio 0.6, the pruned model volume shrinks to 39.4 M, which is approximate to that of YOLOv3 tiny. This again verifies the effectiveness of channel pruning scheme in compressing the model size. On the other hand, although channel pruning enables fewer training parameters in the network, the detection accuracy exhibits a sharp decrease from 0.8907 to 0.8851, 0.8456, and 0.3698, respectively. When the prune ratio increases to 0.9, the detection performance is getting worse to 0.7075 AP, which is much poorer than that of YOLOv3-tiny (0.8241). One of the reasons for this phenomenon might be that too shallower structure will degenerate the detection performance and the loss of accuracy caused by pruning should be compensated by fine-tuning stage. In addition, the inference time under the input size of 416×416 evaluated on a NVIDIA GTX1080Ti GPU is reduced by 38.48%, 42.33%, and 43.90% accordingly. Owing to the pruning of less important channels, the slim detector under the prune ratio 0.95 runs about 1.8 times faster than the original model. In our implementation, we choose the prune ratio as 0.95 for balance of speed and accuracy.

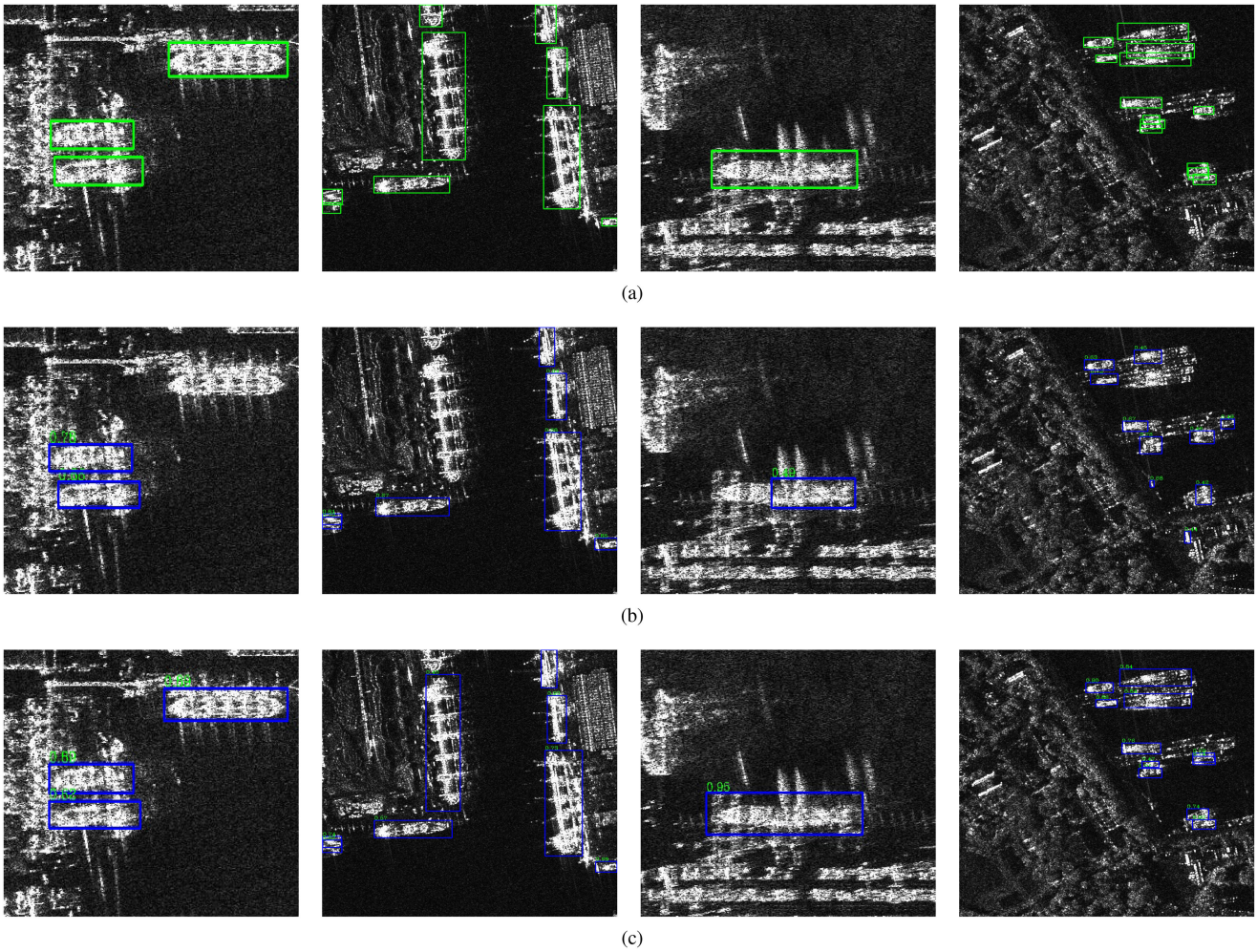


Fig. 8. Comparison results of the detector without and with ACM in the backbone network. The blue rectangles are the detection results and the green numbers above the detection boxes represent the confidence score. (a) Ground truths. (b) Results of the baseline. (c) Results of the baseline with the addition of ACM block.

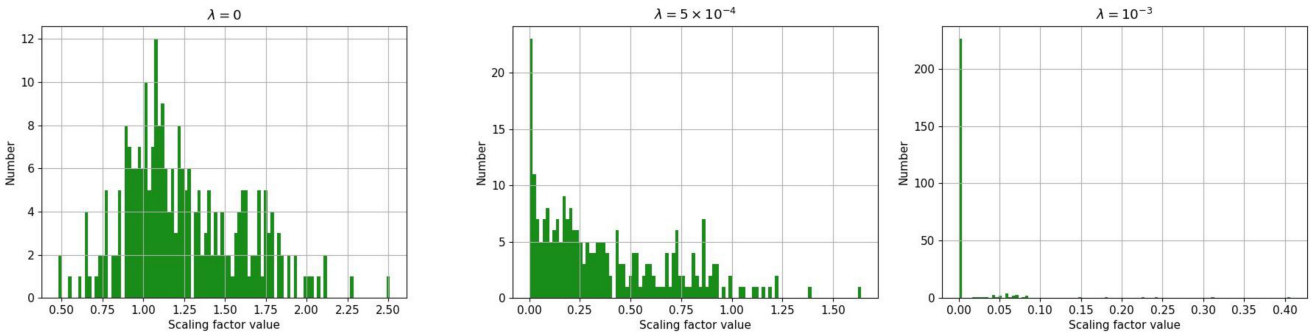
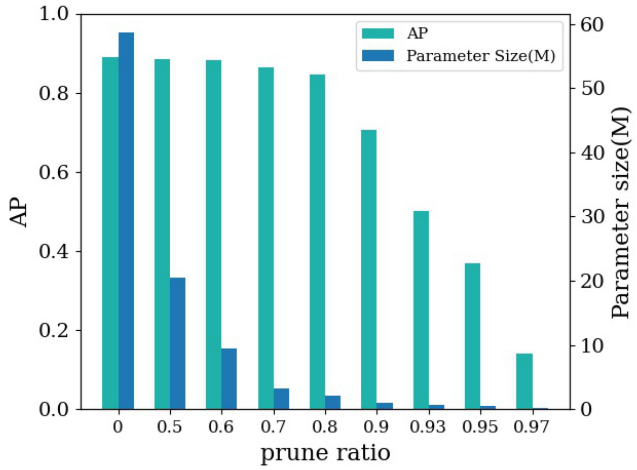


Fig. 9. Distributions of scaling factors in a trained DC-ACM YOLOv3 under different degrees of channelwise sparsity regularization (controlled by λ). $\lambda = 0$ means the base training process. Too small λ shows little impact on sparsity while the increase of λ is conducive for making scaling factors more sparser.

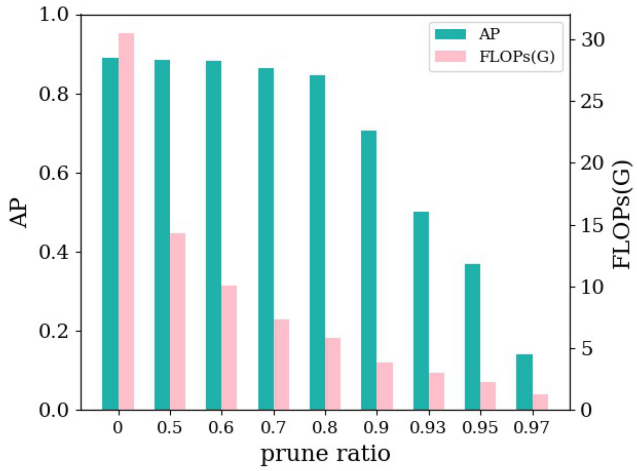
c) Effect of knowledge distillation: As can be seen from Fig. 10, model pruning leads to potential performance degradation especially when the prune ratio is high; however, this accuracy gain compensated by fine-tuning on the pruned network is restricted. In this part, KD strategy is applied to retrain the lightweight detection model obtained in the previous stage. To validate the effectiveness of feature map graph-based KD, we conduct experiments with different configurations to reveal the specific role of each component in improving the detection performance. Several distillation components contain 1) full feature imitation denoted as FFI, which serves as the baseline KD strategy; 2) feature inter-relationship imitation under graph structure denoted as FIR; 3) attention-strengthened feature imitation in which attention refers to the SPAM, and this is denoted as SPAM-FI; and the combination of 2) and 3) denoted as SPAM-FIR. The overall comparative evaluations are reported

TABLE I
EFFECT OF MODULES PROPOSED IN KD STRATEGY IN TWO DIFFERENT SCENES. BOLD VALUES MEAN THE BEST AP UNDER THE INSHORE SCENE AND OFFSHORE SCENE

Scene	Offshore				Inshore			
KD Strategy	FFI	FIR	SPAM-FI	SPAM-FIR	FFI	FIR	SPAM-FI	SPAM-FIR
Attention Module	×	×	✓	✓	×	×	✓	✓
Inter-relationship Graph	×	✓	×	✓	×	✓	×	✓
Precision	0.954	0.956	0.943	0.968	0.686	0.628	0.729	0.712
Recall	0.970	0.975	0.980	0.975	0.838	0.792	0.808	0.838
F1	0.962	0.966	0.961	0.972	0.754	0.701	0.766	0.770
AP	0.9676	0.9794	0.9774	**0.9799**	0.7833	0.7941	0.8010	**0.8247**

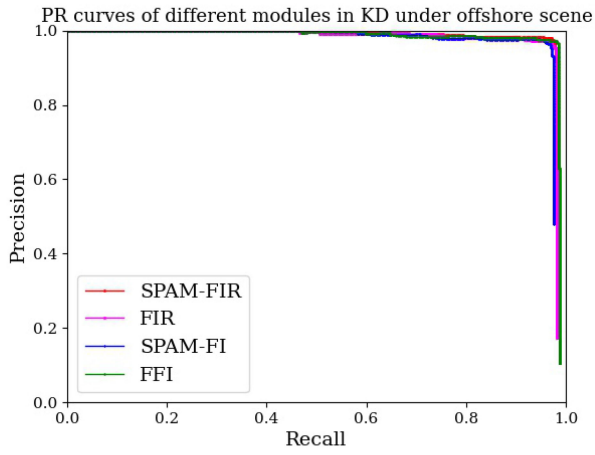


(a)

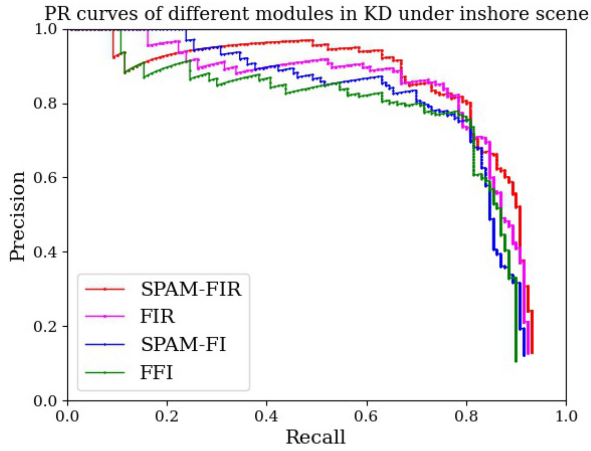


(b)

Fig. 10. Histogram of the relationship between model property and average precision under different pruning ratios. (a) Parameter size versus average precision. (b) FLOPs versus average precision.



(a)



(b)

Fig. 11. Precision–recall (PR) curves of different components in KD tested on offshore and inshore ships. (a) PR curves of FFI, FIR, SPAM-FI, and SPAM-FIR tested on offshore ships. (b) PR curves of FFI, FIR, SPAM-FI, and SPAM-FIR tested on inshore ships.

in Table I, and the corresponding PR curves of different settings tested on inshore and offshore ships are presented in Fig. 11(a) and (b), respectively.

As can be seen in Table I, SPAM-FIR consistently outperforms the other methods. With the combination of feature inter-relationship graph structure and attention mechanism, the KD strategy SPAM-FIR displays different degrees of improvement on the FFI baseline, such as 1.0%, 1.23% higher F1, AP on offshore scene and more distinct discrepancy of 1.6%, 4.14% higher F1, AP on inshore scene.

In terms of offshore ships, the PR curves of FIR versus SPAM-FIR and FFI versus SPAM-FI are similar to each other, while the improved ones are all slightly better than that of FFI. This means different KD strategies have insignificant impact when detecting those offshore ships. Nevertheless, apparent difference can be observed among the PR curves of these strategies when

TABLE II
EVALUATION RESULTS OF BASELINE MODELS WITH DIFFERENT BACKBONES AND PRUNED MODELS WITH KD STRATEGY. THE MOST LIGHTWEIGHT MODEL VOLUME AND PARAMETER SIZE ARE MARKED AS BOLD

Model Name	YOLOv3-Darknet53	YOLOv3-SPP	YOLOv3-tiny	YOLOv3-mobilenet	DC-ACM YOLOv3(Teacher)	Pruned DC-ACM YOLOv3(student)	Pruned DC-ACM YOLOv3(+finetune)	Tiny YOLO-Lite(Student+KD)
Precision	0.936	0.928	0.835	0.973	0.855	0.042	0.773	0.896
Recall	0.938	0.944	0.827	0.886	0.936	0.831	0.939	0.938
AP	0.944	0.947	0.814	0.872	0.934	0.369	0.924	0.946
F1-score	0.937	0.936	0.831	0.927	0.984	0.079	0.848	0.917
Inference time(ms)	30.8	30.9	14.0	18.8	26.8	2.5	3.8	3.9
FPS	32.48	32.40	71.18	53.27	37.24	**397.48**	258.26	258.61
Parameter Size(M)	58.67	59.68	8.27	22.68	47.23	**0.61**	**0.61**	**0.61**
Model Volume(M)	246.4	250.7	34.8	95.5	202.3	**2.8**	**2.8**	**2.8**

TABLE III
COMPARISON OF EVALUATION RESULTS UNDER DIFFERENT OBJECT DETECTORS ON SSDD. NOTE THAT THE COMPUTATION COST REPRESENTED BY FLOPS IS CALCULATED AT THE INPUT SIZE OF 416×416 WHEN YOLO-SERIES METHODS ARE EVALUATED, WHILE THE INPUT SIZE IS SET AS 512×512 WHEN OTHER METHODS ARE INVOLVED. THE BOLD VALUE ALSO MEANS THE BEST PERFORMANCE

Detection Method	AP	F1-score	FLOPs(G)	FPS	Inference Time(ms)	Parameter Size(M)	Model Volume(M)
Faster R-CNN	0.827	0.831	49.37	16.0	62.5	40.2	323.0
FPN	0.893	0.839	63.25	15.6	64.2	41.1	330.2
SSD	0.904	0.893	59.86	20.1	49.8	36.0	225.1
RetinaNet	0.921	0.862	61.22	18.0	55.5	37.7	290.0
RefineDet	0.932	0.898	64.05	17.7	56.4	39.3	299.6
YOLOv3	0.944	0.937	30.51	32.5	30.8	58.7	246.4
DAPN	0.898	0.841	62.87	14.2	70.4	38.8	270.3
DS-CNN	0.917	0.906	9.54	77.3	12.9	3.2	13.7
HR-SDNet	0.943	**0.938**	91.29	11.4	87.7	69.2	557.5
Tiny YOLO-Lite(Ours)	**0.946**	0.917	**1.94**	**258.6**	**3.9**	**0.6**	**2.8**

detecting inshore ships [see Fig. 11(b)]. The precision rate under the SPAM-FIR strategy is always higher than those of FFI, FIR, and SPAM-FI especially when the recall rate is higher than 0.8.

The significant improvement of our method mainly comes from two aspects: 1) The feature map graph structure considers the relationship between different levels of feature maps, which not only ensures more sufficient knowledge can be extracted from the teacher but also means the imitation will be less influenced by the size inconsistency of feature map scales. 2) The attention module used in the graph structure effectively reinforces the feature of object-related area for more accurate imitation and, thus, suppresses the undesirable background clutter while maximally retaining the target information.

Furthermore, we also implement some YOLO-series methods based on backbones with different model volumes, such as YOLOv3-tiny, YOLOv3-mobilenet, and some improved versions of YOLOV3 such as YOLOv3-SPP for fair comparison. The backbone network in the above models is pretrained using the ImageNet dataset, while the proposed detector using the self-designed backbone is trained from scratch. The DC-ACM YOLOv3 based on Darknet-ARes is used as the teacher network to examine the difference of performance according to the teacher architecture, while the lightweight model pruned from the teacher network is utilized as the student networks. With the participation of KD strategy, the final tiny model is denoted as Tiny YOLO-Lite.

Table II shows the comparison results in terms of accuracy, speed, computational cost, and model volume, in which the evaluation metrics of directly using teacher and student without KD are also depicted as baseline. Compared with the original

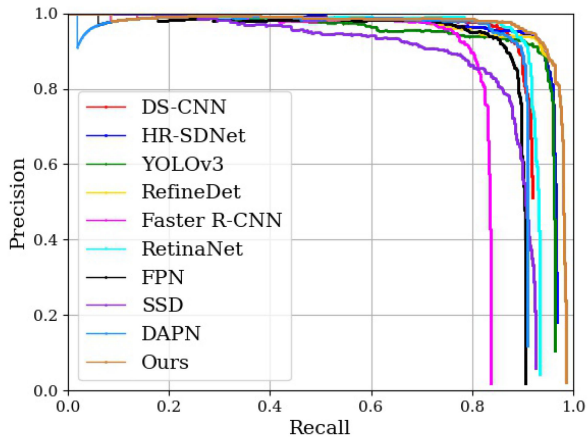


Fig. 12. Comparison of different CNN-based methods on SSDD.

YOLOv3-based methods (see the second and third columns of Table II), the proposed tiny model enjoys 0.264% better AP than YOLOv3, while 0.084% slightly lower AP than YOLOv3-SPP. In contrast with some relative lightweight backbone models, Tiny YOLO-Lite shows about 16.30% and 8.53% more excellent performance than YOLOv3-tiny and YOLOV3-mobilenet in detection accuracy, respectively. Meanwhile, 92.66% and 97.30% fewer trainable parameters are lessened and 2.63 and 3.85 times faster inference speed can be guaranteed. When DC-ACM YOLOv3 is pruned without KD, although the fastest inference speed can be ensured, the precision rate decreases sharply accordingly. This is because a too small network shows limited capability of feature representation. Admittedly, fine-tuning

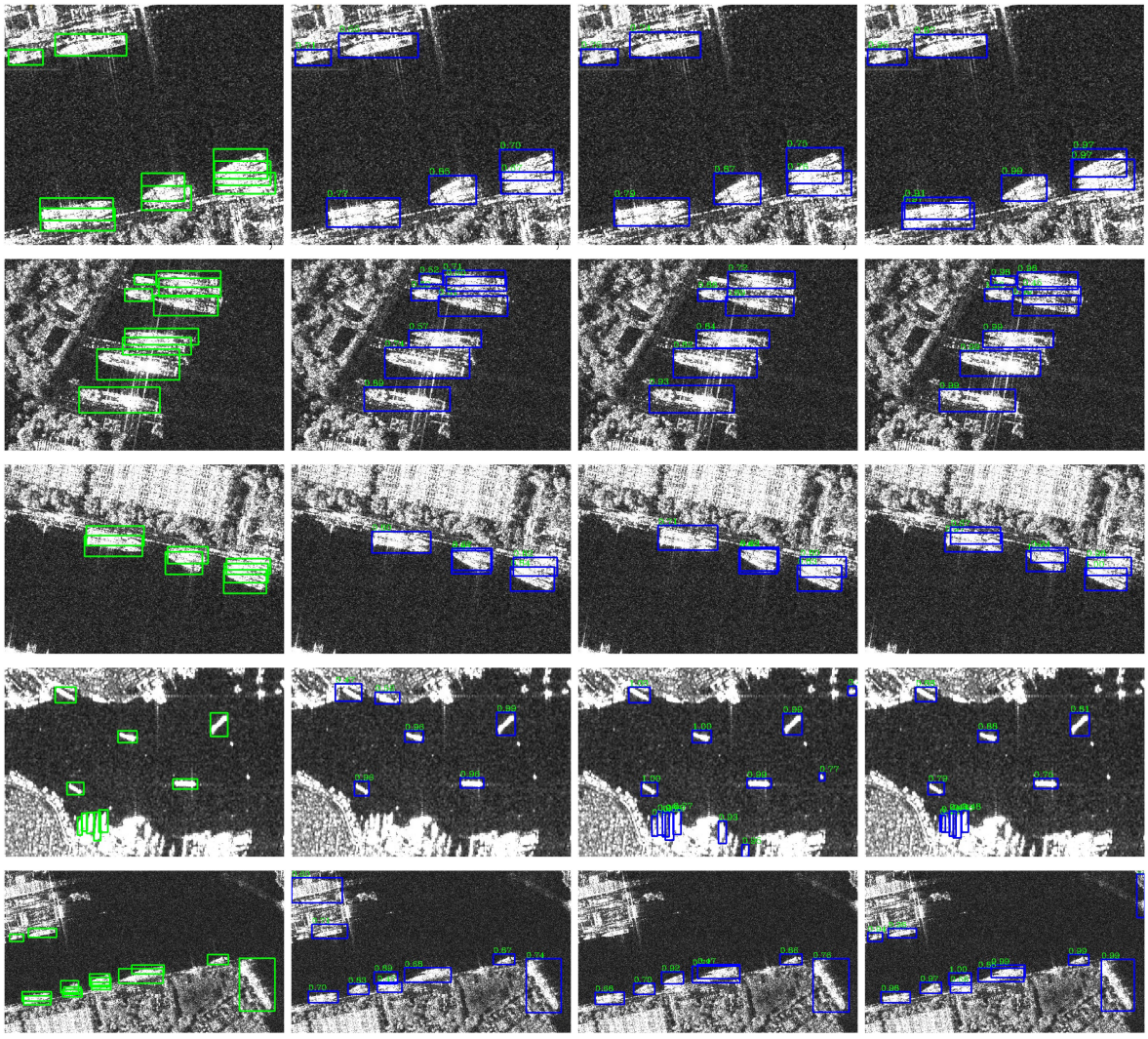


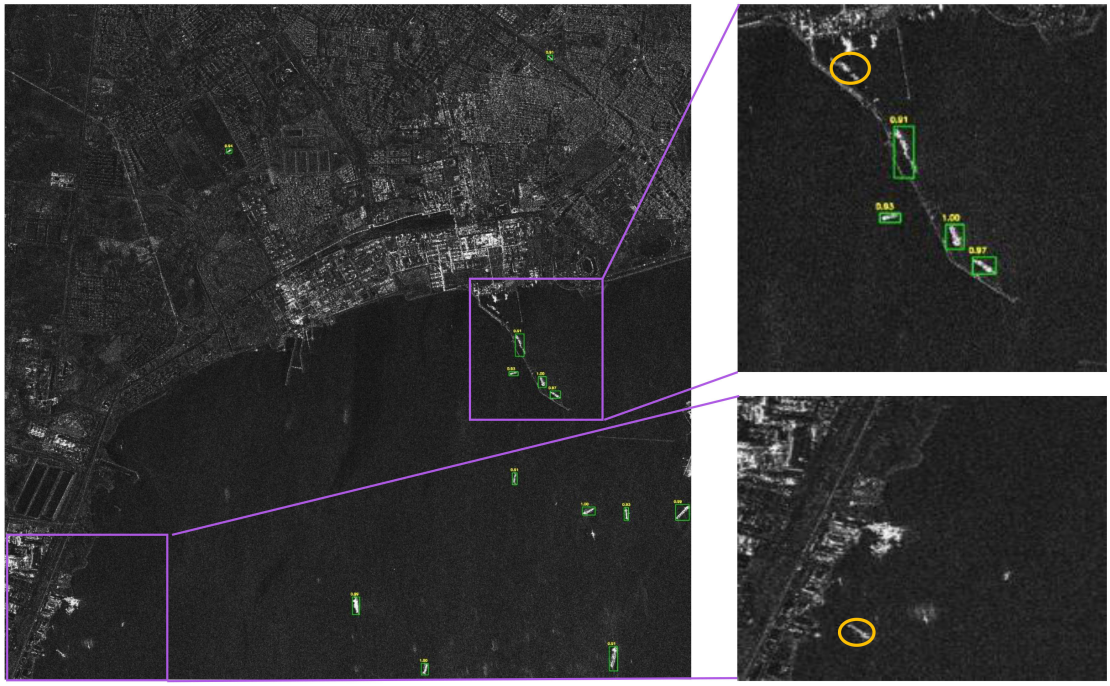
Fig. 13. Comparison of ground truths and visual detection results under different CNN-based methods on SSDD inshore ships. Rectangles with green color mark the ground truths, and those added with green number mark the prediction results with confidence scores. (a) Ground truths. (b–d) Results of DS-CNN, YOLOv3-mobilenet, and our method, respectively.

stage helps elevate the detection performance to a certain degree; however, KD strategy can further achieve up to 2.2% accuracy gain than using fine-tuning alone. In addition, the proposed tiny model can achieve better results than that of both the teacher and student baseline, especially in the aspect of reduced computation cost, compressed model size, and raised inference speed. This indicates that employing feature inter-relational information transferred from a deep teacher model to a lightweight student will significantly make up for the performance loss caused by pruning and even attain 1.24% performance enhancement in terms of AP, which is even beyond the teacher baseline.

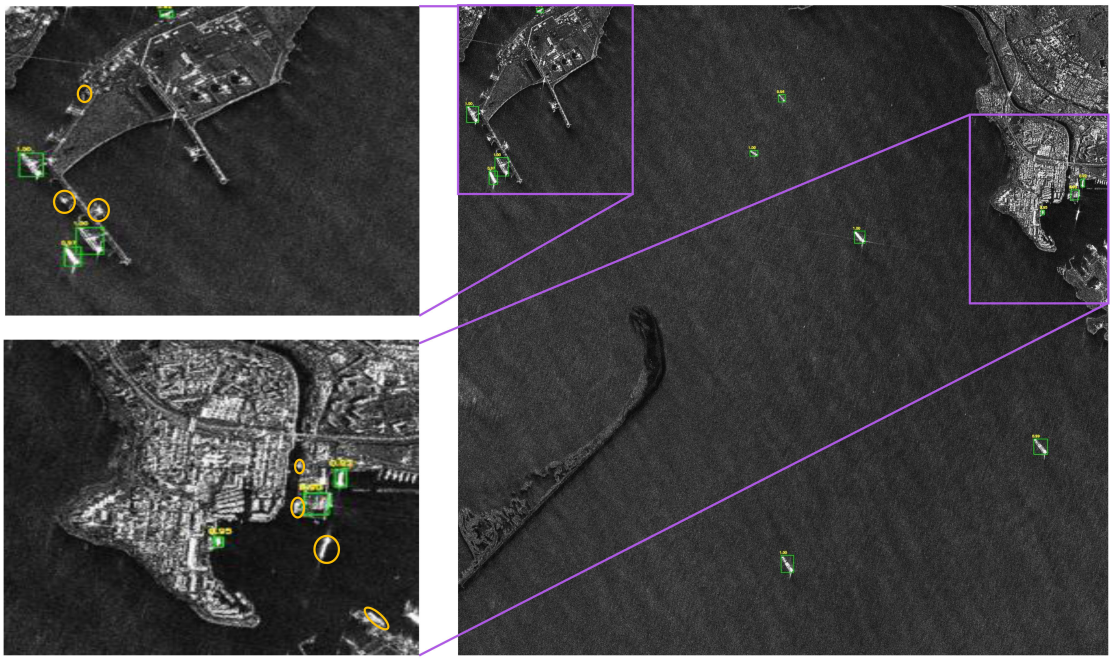
F. Comparison of State-of-the-Art Methods

In this section, SSDD test set is adopted to verify the superiority of the proposed method compared with other state-of-the-art CNN-based methods. As shown in Table III, the detection performance of classical CNN-based methods in terms of speed, accuracy, and computational complexity are presented in the first six rows, while the seventh to ninth rows denote the results of some newest methods specially designed for SAR ship detection.

What stands out in this table is the slow decrease in the performance of Tiny YOLO-Lite model with large compress ratio of model parameters. The AP and F1-score of our method cannot reach the optimal at the same time; nevertheless, the best performance attained in other evaluation indexes can compensate it. More prominently, with only a tiny model of 2.8 M, the detection speed of the proposed method is faster than all the others by several or even tens of times. Thus, a good balance between speed and accuracy can be well ensured in Tiny YOLO-Lite. Finally, from the perspective of lightweight attribute, we compare computational cost, parameter quantity, as well as model volume of different methods. From Table III, compelling results are achieved by the proposed model in comparison with



(a)



(b)

Fig. 14. Results on AIR-SARShip-1.0. The green rectangles represent the detection results. The orange ellipses denote the missing ships. (a) Image 1. (b) Image 2.

the unpruned counterpart YOLOv3, including ∼93.6% decrease of FLOPs, ∼98.9% decline of parameter size, and only $\sim 0.01\times$ model volume.

Although DS-CNN, which is mainly composed of depthwise convolutions and other regular strategies used in YOLOv3, also enables high-speed SAR ship detection, our model still surpasses it in terms of 2.9% higher AP, 1.1% improved F1-score and even about 3.35 times faster speed, 0.19 times smaller model size. Therefore, Tiny YOLO-Lite indeed achieves the lightest and the minimum calculated SAR ship detection framework which combines the advantage of both high accuracy and fast speed.

More intuitive performance comparisons such as PR curves are demonstrated in Fig. 12. Apparently, our method improves both the precision and recall rate considerably, reaching the best

performance in terms of speed and accuracy. Overall speaking, our approach can meet the requirements of accurate detection in real-time applications and is suitable for scenarios where the hardware resources on the satellite are constrained.

Fig. 13 displays the detection results of inshore ships tested in complicated scenarios. The ships are intensively packed in a complex background and we only choose some lightweight models for fair comparison. Comprehensive detection results indicate that when the scene is complicated, both YOLOv3-mobilenet and DS-CNN have the phenomenon of missing detection (from the first three rows) and false alarms (from the last two rows). Some two parallel parking ships are detected as one ship, or the prediction boxes of them are severely overlapped. This is partly due to the unclear boundary between densely parked ships, which leads to semantic feature under-representation of the whole ship. In the proposed detector, we can obtain the predicted boxes with higher quality and more compact representation, which is extremely beneficial for distinguishing the parallel parking ships distinctively. Additionally, more false alarms such as isles and man-made facilities appear in the context of inland area. The reason may be that the adjacent interference with similar scattering intensity degrades the performance, while the powerful feature attentive mechanism in KD strategy helps our model spotlight pivotal pixels while restraining uninformative ones and, thus, discriminate ships against the background scatters more easily. To conclude, owing to the collective effect of channel pruning and KD, our method is simple and computationally efficient to achieve the outstanding detection results.

G. Generalization Ability Verification

In order to confirm the robustness and generalization ability of the proposed method, we perform ship detection in two large-scene SAR ship images in AIR-SARShip-1.0 dataset. The following figure shows their detection results. From Fig. 14, most ships in these two images can be successfully detected. In the first image, there are three missing detection cases meanwhile two false alarm cases. In the second image, there are seven missing detection cases while only one false alarm case. Both of them are quite complex and contain almost all types of missing detections and false alarms, which fully reflects the generalization ability of the proposed detector. Although the detection results are slightly inferior than that on the trained dataset, the migration capability is prospective to be improved.

V. Conclusion

Due to the large model size, SAR ship object detector based on deep learning cannot be deployed on hardware devices with constrained computational resources. In this article, a lightweight object detector named Tiny YOLO-Lite is proposed to achieve efficient SAR ship detection with both high speed and high accuracy. First, we redesign a novel detection framework DC-ACM YOLOv3 according to the multiscale and multiaspect ratio characteristic of SAR ships. Then, channel pruning scheme is applied to drastically reduce the number of parameters and computations. To compensate the accuracy loss caused by network compression, a novel KD strategy is introduced to maintain a competitive detection performance. Our feature distillation method is a general framework which attentively imitates informative features and suppresses noninformative ones in the format of feature inter-relationship graph and can be universally applicable for many modern detectors. To this end, a slim SAR ship detection network with only 2.8 M model size is obtained, which is only 1% of the original YOLOv3 in terms of model volume. Tiny YOLO-Lite is built from scratch with significant fewer network parameters, lower computation costs, and lighter model volume. Experiments conducted on two SAR ship benchmark datasets have proven the superiorities of our method compared with other state-of-the-art CNN-based methods. In the future, more efficient distillation methods without time-consuming training will be explored in the combination of pruning methods to make the model compression more diversified and easier for transplantation.

References

[1] F. Sharifzadeh, G. Akbarizadeh, and Y. S. Kavian, "Ship classification in SAR images using a new hybrid CNMLP classifier," *J. Indian Soc. Remote Sens.*, vol. 47, no. 4, pp. 551–562, 2019.

[2] F. Samadi, G. Akbarizadeh, and H. Kaabi, "Change detection in SAR images using deep belief network: A new training approach based on morphological images," *IET Image Process.*, vol. 13, no. 12, pp. 2255–2264, 2019.

[3] M. Zalpour, G. Akbarizadeh, and N. Alaei-Sheini, "A new approach for oil tank detection using deep learning features with control false alarm rate in high-resolution satellite imagery," *Int. J. Remote Sens.*, vol. 41, no. 6, pp. 2239–2262, 2020.

[4] Z. Tirandaz, G. Akbarizadeh, and H. Kaabi, "PolSAR image segmentation based on feature extraction and data compression using weighted neighborhood filter bank and hidden Markov random field-expectation maximization," *Measurement*, vol. 153, 2020, Art. no. 107432.

[5] S. Ren *et al.*, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[6] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.

[7] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE Conf. Comput. Vis Pattern Recognit.*, 2018, pp. 6154–6162.

[8] J. Redmon *et al.*, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.

[9] W. Liu *et al.*, "SSD: Single shot multibox detector," in *European Conference on Computer Vision*, Berlin, Germany: Springer, Oct. 2016, pp. 21–37.

[10] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–328, Feb. 2020.

[11] Z. Cui *et al.*, "Dense attention pyramid networks for multi-scale ship detection in SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 8983–8997, Nov. 2019.

[12] C. Chen *et al.*, "A deep neural network based on an attention mechanism for SAR ship detection in multiscale and complex scenarios," *IEEE Access*, vol. 7, pp. 104848–104863, 2019.

[13] Z. Cui *et al.*, "Ship detection in large-scale SAR images via spatial shuffle-group enhance attention," *IEEE Trans. Geosci. Remote Sens.*, vol. 99, pp. 1–13, 2020.

[14] S. Wei *et al.*, "Precise and robust ship detection for high-resolution SAR imagery based on HR-SDNET," *Remote Sens.*, vol. 12, no. 1, 2020, Art. no. 167.

[15] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis Pattern Recognit.*, Jul. 2017, pp. 7263–7271.

[16] J. Redmon and A. Farhadi, "YOLOV3: An incremental improvement," 2018, *arXiv:1804.02767*.

[17] A. Van Etten, "You only look twice: Rapid multi-scale object detection in satellite imagery," 2018, *arXiv:1805.09512*.

[18] Y.-L. Chang *et al.*, "Ship detection based on YOLOV2 for SAR imagery," *Remote Sens.*, vol. 11, no. 7, 2019, Art. no. 786.

[19] T. Zhang *et al.*, "Depthwise separable convolution neural network for high-speed SAR ship detection," *Remote Sens.*, vol. 11, no. 21, 2019, Art. no. 2483.

[20] T. Zhang *et al.*, "Hyperli-Net: A hyper-light deep learning network for high-accurate and high-speed ship detection from synthetic aperture radar imagery," *ISPRS J. Photogrammetry Remote Sens.*, vol. 167, pp. 123–153, 2020.

[21] L. Deng *et al.*, "Model compression and hardware acceleration for neural networks: A comprehensive survey," *Proc. IEEE*, vol. 108, no. 4, pp. 485–532, Apr. 2020.

[22] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. ICLR*, 2016.

[23] X. Lin, C. Zhao, and W. Pan, "Towards accurate binary convolutional neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 345–353.

[24] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[25] A. Romero *et al.*, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*.

[26] G. Chen *et al.*, "Learning efficient object detection models with knowledge distillation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 742–751.

[27] Q. Li, S. Jin, and J. Yan, "Mimicking very efficient network for object detection," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6356–6364.

[28] Z. Liu *et al.*, "Learning efficient convolutional networks through network slimming," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2736–2744.

[29] S. Han *et al.*, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.

[30] H. Li *et al.*, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*.

[31] P. Molchanov *et al.*, "Pruning convolutional neural networks for resource efficient inference," 2016, *arXiv:1611.06440*.

[32] J. Ye *et al.*, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," 2018, *arXiv:1802.00124*.

[33] Z. Huang and N. Wang, "Data-driven sparse structure selection for deep neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 304–320.

[34] K.-H. Shih *et al.*, "Real-time object detection via pruning and a concatenated multi-feature assisted region proposal network," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 1398–1402.

[35] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1389–1397.

[36] Z. Li *et al.*, "Learning slimming SSD through pruning and knowledge distillation," in *Proc. Chin. Automat. Congr.*, 2019, pp. 2701–2705.

[37] P. Zhang, Y. Zhong, and X. Li, "SlimYOLOv3: Narrower, faster and better for real-time UAV applications," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2019, pp. 37–45.

[38] Z. Wang *et al.*, "Efficient YOLO: A lightweight model for embedded deep learning object detection," in *Proc. IEEE Int. Conf. Multimedia Expo. Workshops.*, 2020, pp. 1–6.

[39] T. Chen, I. Goodfellow, and J. Shlens, "Net2Net: Accelerating learning via knowledge transfer," 2015, *arXiv:1511.05641*.

[40] Z. Wang, Z. Deng, and S. Wang, "Accelerating convolutional neural networks with dominant convolutional kernel and knowledge pre-regression," in *Eur. Conf. Comput. Vis.*, Berlin, Germany: Springer, 2016, pp. 533–548.

[41] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," 2016, *arXiv:1612.03928*.

[42] J. Yim *et al.*, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4133–4141.

[43] T.-Y. Lin *et al.*, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.

[44] G. Huang *et al.*, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.

[46] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[47] F. Gao *et al.*, "Enhanced feature extraction for ship detection from multi-resolution and multi-scene synthetic aperture radar (SAR) images," *Remote Sens.*, vol. 11, no. 22, 2019, Art. no. 2694.

[48] Q. Hou *et al.*, "Strip pooling: Rethinking spatial pooling for scene parsing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4003–4012.

[49] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[50] J. Li, C. Qu, and J. Shao, "Ship detection in SAR images based on an improved faster R-CNN," in *Proc. BIGSARDATA*, Beijing, China, 2017, pp. 1–6.

[51] S. Xian *et al.*, "Air-SARShip–1.0: High resolution SAR ship detection dataset," *J. Radars*, vol. 8, no. 6, pp. 852–862, 2019.

[52] S. Wei *et al.*, "HRSID: A high-resolution SAR images dataset for ship detection and instance segmentation," *IEEE Access*, vol. 8, pp. 120234–120254, 2020.