ISPRS Journal of Photogrammetry and Remote Sensing 167 (2020) 123–153

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

# HyperLi-Net: A hyper-light deep learning network for high-accurate and high-speed ship detection from synthetic aperture radar imagery

Tianwen Zhang, Xiaoling Zhang*, Jun Shi, Shunjun Wei

*School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China*

ARTICLE INFO

*Keywords:*
HyperLi-Net
Deep learning
Ship detection
Synthetic Aperture Radar (SAR)
High-speed
High-accurate

ABSTRACT

Ship detection from Synthetic Aperture Radar (SAR) imagery is attracting increasing attention due to its great value in ocean. However, existing most studies are frequently improving detection accuracy at the expense of detection speed. Thus, to solve this problem, this paper proposes HyperLi-Net for high-accurate and high-speed SAR ship detection. We propose five external modules to achieve high-accuracy, i.e., Multi-Receptive-Field Module (MRF-Module), Dilated Convolution Module (DC-Module), Channel and Spatial Attention Module (CSA-Module), Feature Fusion Module (FF-Module) and Feature Pyramid Module (FP-Module). We also adopt five internal mechanisms to achieve high-speed, i.e., Region-Free Model (RF-Model), Small Kernel (S-Kernel), Narrow Channel (N-Channel), Separable Convolution (Separa-Conv) and Batch Normalization Fusion (BN-Fusion). Experimental results on the SAR Ship Detection Dataset (SSDD), Gaofen-SSDD and Sentinel-SSDD show that HyperLi-Net's accuracy and speed are both superior to the other nine state-of-the-art methods. Moreover, the satisfactory detection results on two Sentinel-1 SAR images can reveal HyperLi-Net's good migration capability. HyperLi-Net is build from scratch with fewer parameters, lower computation costs and lighter model that can be efficiently trained on CPUs and is helpful for future hardware transplantation, e.g. FPGAs, DSPs, etc.

## 1. Introduction

Synthetic Aperture Radar (SAR) is a reliable tool for ocean monitoring for all-day and all-weather working ability (Kanjir and Greidanus, 2018). Ships in SAR images as valuable marine targets have been a focus of ocean monitoring. Recent years, SAR ship detection is attracting increasing attention for its great value in ocean, e.g. traffic control (Meyer et al., 2006), fishery management (Petit et al., 1992), environment protection (Nunziata et al., 2013), disaster rescue (Koyama et al., 2016), etc. On the civilian side, it is helpful for water transport management, ship rescue, illegal fishing, illegal oil pollution dumping, etc. On the military side, it plays an important role in ensuring battlefield initiative. In 1978, the United States launched the first civil SAR satellite (Born et al., 1979) to carry out ocean exploration opening up the era of SAR ship detection. As the launches of ERS-1 and RadarSat-1, increasing efforts have been devoted to this, but compared to optical ship detection, SAR ship detection is still behindhand (Zhang and Zhang, 2019) because SAR images cannot be understood intuitively and are often accompanied by speckle noise making interpretation more challenging.

So far, many SAR ship detection methods or algorithms have emerged, but exsiting most studies are frequently improving detection accuracy at the expense of detection speed. Moreover, our survey finds that there are few researches focusing on detection speed. One possible reason is that the overall time to acquire SAR images, from radar echo signal acquisition, focus and specific imaging algorithm processing, is often more than a few minutes in the fastest case, so adding to computation cost a few seconds/milliseconds is a minor concern for full-link SAR application (Zhang and Zhang, 2019), so it is easy to conscious accuracy importance. However, SAR application consists of images' acquisition and their interpretation. Imaging algorithms are the former's focus meanwhile image processing techniques are the latter's focus, and in fact it is valuable for the latter to improve detection speed where the millisecond-level speed improvement is a great progress (Zhang and Zhang, 2019).

In short, it is unsatisfactory to sacrifice speed for accuracy because some real-time occasions need both high-accuracy and high-speed, e.g. emergency military deployment, rapid maritime rescue, etc. Thus, to solve this problem, HyperLi-Net is proposed that is established on five external modules and five internal mechanisms. See Fig. 1.

From Fig. 1a, the five external modules for high-accurate SAR ship detection are as follows:

(1) Module 1: MRF-Module (Multi-Receptive-Field Module). Extract all-around image information.

* Corresponding author.
*E-mail addresses:* twzhang@std.uestc.edu.cn (T. Zhang), xlzhang@uestc.edu.cn (X. Zhang), shijun@uestc.edu.cn (J. Shi), weishunjun@uestc.edu.cn (S. Wei).

https://doi.org/10.1016/j.isprsjprs.2020.05.016
Received 3 January 2020; Received in revised form 14 May 2020; Accepted 19 May 2020
Available online 21 July 2020

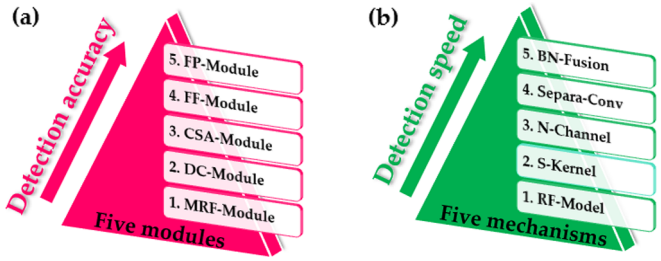



**Fig. 1.** Basic Design Concept of HyperLi-Net. **(a)** Five External Modules; **(b)** Five Internal Mechanisms.

(2) Module 2: DC-Module (Dilated Convolution Module). Expand MRF-Module's receptive-field (Hubel and Wiesel, 1959).
(3) Module 3: CSA-Module (Channel and Spatial Attention Module). Distinguish important or inessential features.
(4) Module 4: FF-Module (Feature Fusion Module). Fuse shallow and deep features.
(5) Module 5: FP-Module (Feature Pyramid Module). Detect multi-scale ships.

From Fig. 1b, the five internal mechanisms for high-speed SAR ship detection are as follows:

(1) Mechanism 1: RF-Model (Region-Free Model). Avoid Region Of Interests (ROIs) generation.
(2) Mechanism 2: S-Kernel (Small Kernel). Use smaller kernels to reduce network parameters.
(3) Mechanism 3: N-Channel (Narrow Channel). Use fewer kernels to further reduce network parameters.
(4) Mechanism 4: Separa-Conv (Separable Convolution). Use Separa-Conv (Sifre, 2014)instead of traditional convolution (Lecun et al., 1998).
(5) Mechanism 5: BN-Fusion (Batch Normalization Fusion). Fuse BN (Ioffe and Szegedy, 2015) into Separa-Conv in the detection model.

Experimental results on the SAR Ship Detection Dataset (SSDD) (Li et al., 2017; Li et al., 2019; Li et al., 2019; Li et al., 2019)c), Gaofen-SSDD and Sentinel-SSDD show that HyperLi-Net's accuracy and speed are both superior to the other nine state-of-the-art detectors, e.g. Faster R-CNN (Ren et al., 2017), R-FCN (Dai et al., 2016), YOLO (Redmon et al., 2015; Redmon and Farhadi, 2016; Redmon and Farhadi, 2018), SSD (Liu et al., 2015) and RetinaNet (Lin et al., 2017). Moreover, its good migration ability is verified by results on Sentinel-1 SAR images. HyperLi-Net is build from scratch with fewer parameters (103,754), lower computation costs (203,754 FLOPs) and lighter model (0.69 MB) and it can also avoid pre-training on ImageNet (He et al., 2018). HyperLi-Net can promote real-time application, improve full-link SAR application efficiency convenient for follow-up ship classification, slash expenses from GPUs and contribute to future hardware transplantation (FPGAs/DSPs have < 10 MB on-chip memory limit (Iandola et al., 2016).).

The main contributions of our work are as follows:

(1) HyperLi-Net is proposed for high-speed and high-accurate SAR ship detection.
(2) Five modules and five mechanisms are proposed to ensure HyperLi-Net's good detection performance.

The rest of this paper is arranged as six parts. Section 2 presents related work. Section 3 introduces HyperLi-Net. Section 4 describes experiments. Results are shown in Section 5. Ablation studies are introduced in Section 6. Finally, a summary of this paper is made in Section 7.

## 2. Related work

Exsiting SAR ship detection methods can be divided into two categories, i.e., traditional concrete-feature-based methods and modern abstract-feature-based methods. See taxonomy in Fig. 2.

### 2.1. Traditional Concrete-Feature-Based methods

Traditional concrete-feature-based methods detect ships by handcraft features that are theoretically easy-explained. Some common features are backscatter, polarization, geometric, Histogram of Oriented Gradient (HOG) (Dalal and Triggs, 2005), Haar (Mita et al., 2005), Local Binary Pattern (LBP) (Shen, 2015) and Scale-Invariant Feature Transform (SIFT) (Lowe, 2004).

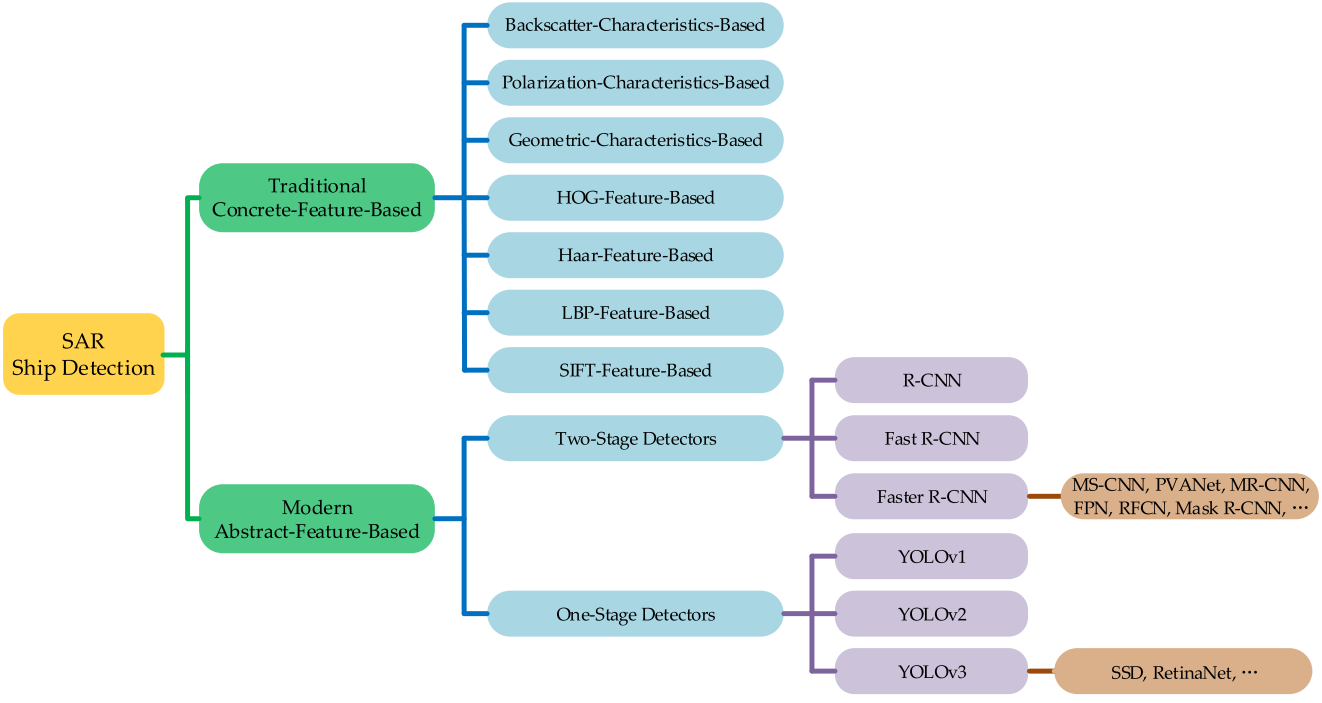



**Fig. 2.** Taxonomy of SAR Ship Detection Methods.

#### 2.1.1. Backscatter-Characteristics-Based methods

Backscatter-characteristics-based methods (Gui, 2011; An et al., 2014; Biao et al., 2015) are the most fundamental ones. Normally, ships made of metal materials have stronger backscatter characteristics, so their gray values are higher. Constant False Alarm Rate (CFAR) (Gui, 2011; An et al., 2014; Biao et al., 2015) was proposed to establish a statistical distribution model of background clutter, by which the most appropriate threshold is gained for ships. Thereinto, some widely-used ones are rayleigh-distribution (Benachenhou et al., 2013), lognormal-distribution (Guida et al., 1993), weibull-distribution (Anastassopoulos and Lampropoulos, 1995); *k*-distribution (Erfanian, 2009), etc. However, establishing an accurate statistical model is a hard task and these models are vulnerable to ocean currents and hard-predictable marine environment causing poor migration ability. Moreover, the need to solve complex distribution equations causes their detection speed quite unsatisfactory (Zhang and Zhang, 2019; Zhang et al., 2019). Especially, they also has great difficulties for detecting ships near ports and reefs (Wang et al., 2018).

#### 2.1.2. Polarization-Characteristics-Based methods

Polarization-characteristics-based methods (Zhang et al., 2019; Atteia and Collins, 2013; Zhang et al., 2019; Gao et al., 2018) can comprehensively describe the scattering difference between ships and sea clutter. Recent years, with the launches of PolSAR satellites, e.g. TerraSAR-X, Gaofen-3, Sentinel-1, etc., PolSAR ship detection studies are rapidly increasing. Compared to single-polarization SAR, PolSAR contains full scattering information, so the detection accuracy obtains significantly improved. However, for PolSAR, establishing an all-sided and abundant polarization scattering feature library is heavy and time-consuming. For another thing, for the ship detection in large-region images, large amount of calculation will be generated, leading to slower detection speed.

#### 2.1.3. Geometric-Characteristics-Based methods

Geometric-characteristics-based methods (Yin et al., 2013; Jiang et al., 2012; Wang et al., 2017; Wang et al., 2016; Zhu et al., 2017) utilize some intuitive geometric features to detect ships, e.g. length, width, aspect ratio, perimeter, area, contour, etc. These features are visual descriptions of ships with strong representativeness, which play an important role in target detection and recognition. In practical application, these methods usually employ a variety of templates to match pixels in SAR images to search for ships (one template corresponds to one geometric feature), by which the desirable ship detection results have obtained on some occasions. However, establishing an adequate template library is time-consuming and labor-intensive, and it often excessively relies on expert experience, causing poor generalization ability. Moreover, they need to calculate all image pixels bringing huge computation costs.

#### 2.1.4. HOG-Feature-Based methods

HOG-feature-based methods (Song et al., 2016; Gan et al., 2016; Lin et al., 2018) detect ships by calculating the gradient direction histogram of images' local areas with the advantage of geometric invariance. They were first applied in pedestrian detection (Hoang et al., 2014) and then were popularized into ship detection including optical images (Dong et al., 2019) and SAR images (Song et al., 2016). Their advantage is that they can detect ships even though ship edge's specific location is unknown. However, for HOG detectors, feature description operators are very sensitive to speckle noise due to gradient property. Worse still, both the long-time feature operators' generation and the much computation from huge feature dimension jointly result in poor real-time performance.

#### 2.1.5. Haar-Feature-Based methods

Haar-feature-based methods (Tello et al., 2006; Schwegmann et al., 2017; Ai et al., 2019) can effectively describe images' gray-level change and differences between different pixel sub-modules. For a SAR image, the numerous Haar features can successfully detect ships by changing the size and position of feature templates. However, the number of Haar features is very large, so it is a huge work to calculate all Haar features of an image. Obviously, for a large-size and wide-area image from the spaceborne SAR, these Haar-based methods have a prodigious challenge in ensuring real-time.

#### 2.1.6. LBP-Feature-Based methods

LBP-feature-based methods (Yin et al., 2012; Song et al., 2014; Yang et al., 2017) have the merits of gray invariance and rotation invariance. In these methods, variance and gray co-occurrence matrix serves to describe ships texture information. Compared with HOG detectors and Haar detectors, the feature dimension of LBP detectors gets partly decreased, which improves the detection speed to a certain extent. Unfortunately, detection accuracy gets sacrificed when increasing detection speed. In addition, LBP detectors have difficulty in detecting ships with different sizes because only a small area within a fixed radius is covered.

#### 2.1.7. SIFT-Feature-Based methods

SIFT-feature-based methods (Agrawal and Mangalraj, 2016; Zhou and Lina, 2015; Akagündüz, 2013) search for ships by matching some key feature points that are not affected by noise and affine-transformation. These methods employ difference of gaussians (Bundy and Wallen, 1984) to calculate extremum and then remove unstable points by judging the edge main curvature, so as to obtain the key points with strong anti-noise ability. These key points' gradient is characterized with direction histogram. However, SAR images occasionally have few key points, resulting in limited application scenarios for SIFT detectors.

To sum up, the above traditional methods are mature in feature design but complex in algorithm, weak in migration and cumbersome in manual design (Zhang et al., 2019). Moreover, their speed is too slow to satisfy real-time application (Yang et al., 2019). Recent years, modern abstract-feature-based (deep learning) methods are emerging, making these problems be smoothly solved.

### 2.2. Modern Abstract-Feature-Based methods

Modern abstract-feature-based methods detect ships by extensive abstract features (Lecun et al., 1998; LeCun et al., 2015) that are extracted by deep Convolutional Neural Networks (CNNs) (Koushik, 2016) without human intervention, but they are theoretically hard-explained. Even so, their detection performance has surpassed traditional methods (Krizhevsky et al., 2017). Recent years, the explosive growth of SAR images significantly promotes the extensive application of deep learning in SAR ship detection. Consequently, at present, deep learning is attracting more and more scholars given its simplicity, high efficiency and high accuracy (Zhang, 2020). This paper also follows the deep learning path to study SAR ship detection. So far, many deep learning object detectors have been proposed that can be divided into two categories (Gui et al., 2019)-two-stage and one-stage. See taxonomy in Fig. 2.

#### 2.2.1. Two-Stage detectors

Two-stage detectors assign detection tasks into two stages (Zhang et al., 2019)-ROIs acquisition and ROIs classification. They are designed around regions, so they are regarded as being based on Region Model (R-Model). R-CNN (Girshick et al., 2014) first applied CNNs to object detection, opening an era in which CNNs lead object detection field. R-CNN used Selective Search (SS) (Uijlings et al., 2013) to get ROIs that are inputted AlexNet (Krizhevsky et al., 2017) to extract features. Finally, SVM served to classify ROIs to obtain detection results. Compared to traditional methods, R-CNN greatly improved mean Average Precision (mAP) on Pascal VOC dataset (Everingham et al., 2014); but it is low-efficient for overlapping region's repeated computation. Thus, its

detection speed is rather slow (Girshick et al., 2014) making it hard to apply in industry. To solve this problem, Fast R-CNN was proposed (Girshick and Fast, 2015) that generate ROIs by on VGG's features maps (Simonyan and Zisserman, 2014) and then these ROIs are converted into fixed size by ROI pooling (Girshick and Fast, 2015). Finally, they are fed into Full Connection (FC) layers for classification and location. For the avoidance of feature repeated calculation, Fast R-CNN reduces detection time. However, its ROI acquisition still takes up most detection time, declining detection efficiency. Thus, Faster R-CNN emerged (Li et al., 2019) that replaced tranditional candidate region methods with Region Proposal Networks (RPN) (Li et al., 2019), high-efficient in generating ROIs. Finally, it achieved end-to-end training and detection, so its detection speed is 10 times faster than Fast R-CNN (Li et al., 2019). Faster R-CNN has become a two-stage detectors' mainstream representative (Zhao et al., Nov. 2019). After this, many its improvements are proposed by-1) using stronger backbone networks, e.g. MS-CNN (Cai et al., 2016), PVANet (Hong et al., 2016), etc.; 2) designing more precise RPNs, e.g. MR-CNN (Gidaris and Komodakis, 2015), FPN (Lin et al., 2017), etc.; 3) improving ROIs classification, e.g. R-FCN (Dai et al., 2016), Mask R-CNN (He et al., 2020), etc. However, these two-stage detectors inherently need to obtain region recommendation boxes in advance, leading to heavy computation. As a result, there are some intrinsic technical bottlenecks in improving speed, so they scarcely meet real-time requirement (Zhang et al., 2019). Hence, one-stage detectors emerged to improve detection speed further.

So far, many scholars from the SAR remote sensing community have appiled these two-stage detectors for SAR ship detection. Li et al. (Li et al., 2017) released the first open SAR Ship Detection Dataset (SSDD) and then based on this dataset, an improved Faster R-CNN by feature fusion, transfer learning, hard negative mining and other implementation details for SAR ship detection was proposed. Most importantly, they also pointed out that deep-learning-based SAR ship detection methods must be the focus of future research, which is undoubtedly correct from current research status. After that, Li et al. also (Li et al., 2019) used Generative Adversarial Network (GAN) (Goodfellow et al., 2014) and Online Hard Examples Mining (OHEM) (Shrivastava et al., 2016) to further improve accuracy. Moreover, when Li et al. (Li et al., 2019) applied Fast R-CNN, a smoothing operator was added on the original gradient operator to tackle the speckle noise of SAR images to get more accurate ROIs. Ai et al. (Ai et al., 2019) proposed a Multi-Scale Rotation-Invariant Haar-Like (MSRI-HL) Feature Integrated Convolutional Neural Network (MSRIHL-CNN) ship detection algorithm of the multiple-target environment in SAR imagery. However, their method requires a lot of human participation and does not achieve end-to-end training and detection (Jiao et al., 2018). Gui et al. (Gui et al., 2019) designed a Multilayer Fusion Light-Head Detector (MFLHD) for ship detection based on Faster R-CNN, considering multiscale performance, detection efficiency and hard-easy example balance. Jiao et al. (Jiao et al., 2018) proposed a densely connected (Huang et al., 2016) multiscale neural network based on Faster-RCNN framework to solve multiscale and multi-scene SAR ship detection. They densely connected one feature map to every other feature map from top to down and generate proposals from each fused feature map and also used focal loss to reduce the weight of easy examples. Cui et al. (Cui et al., Nov. 2019) proposed a novel multi-scale SAR ship detectior named Dense Attention Pyramid Network (DAPN) by using Convolutional Block Attention Module (CBAM) (Woo et al., 2018), but the detection accuracy of inshore scene ships is modest. Deng et al. (Deng et al., 2018) proposed a unified and effective method for detecting multi-class objects (airplane, ship, etc.) in multi-modal remote sensing images with large scales variability, but they used pretrained models on ImageNet (He et al., 2018), reducing training efficiency. Thus, to solve this problem, Deng et al. also (Deng et al., 2019) proposed a learning deep SAR ship detector from scratch, more effective than existing algorithms for detecting the small and densely clustered ships. Zhao et al. (Lin et al., 2019) designed a Squeeze Excitation Rank Faster R-CNN (SER Faster R-CNN) for SAR ship detection to improve accuracy, inspired by SE-Net (Hu et al., 2017). However, their classification network architecture needs further improvement. Wei et al. (Wei et al., 2020) adopted a high-resolution backnone network (Sun and Zhao, 2019) to extract ship features based on Cascade R-CNN (Cai and Vasconcelos, 2017), but their detection model is too huge to applied in practical. Zhao et al. (Zhao et al., 2019) developed a coupled CNN for small and densely clustered SAR ship detection consisting of an Exhaustive Ship Proposal Network (ESPN) for ship-like region generation from multiple layers with multiple receptive fields and an Accurate Ship Discrimination Network (ASDN) for false alarm elimination. Moreover, Zhao et al. also (Zhao et al., 2018) used guided visual attention method to design a novel cascade coupled convolutional neural network for the full use of image spatial information. Especially, Digital Elevation Model (DEM) data are adopted to remove ship-like targets on land in their detection model. However, their model comes at the expense of tedious human intervention, reducing detection efficiency. Chen et al. (Chen et al., 2019) used Generalized Intersection Over Union (GIoU) (Rezatofighi et al., 2019) loss to reduce the scale sensitivity of the network. In their model, Soft Non-Maximum Suppression (Soft-NMS) (Bodla et al., 2017) was also introduced to reduce the number of missed detections for ship targets in the presence of severe overlap, but the training and inference speed is bound to decrease. Chen et al. (Chen et al., 2020) embedded RPN into feature pyramid to enhance the small-object/multi-object ship detection performance in complex scenarios. Moreover, the k-means clustering algorithm is used to optimize anchor boxes, but adds complex mathematical calculations, declining detection speed. Kang et al. (Kang et al., 2017) proposed a contextual region-based CNN with multilayer fusion for SAR ship detection, but some weak and tiny ships remain undetected and false alarms on land are hard to rule out.

#### 2.2.2. One-Stage detectors

One-stage detectors achieve detection tasks based on direct position regression (Zhang et al., 2019). These one-stage detectors do not produce ROIs, so they are regarded as being based on Region-Free Model (RF-Model). You Only Look Once (YOLO) is the first one-stage detector (Redmon et al., 2015) that uses a single CNN to realize end-to-end object detection. Different from two-stage detectors, YOLO treated object detection task as a regression problem and directly used CNNs to predict the coordinates and class confidence of bounding boxes from a whole image, whose computation cost gets decreased further, improving detection speed. However, its detection performance on neighboring-small targets is not unsatisfactory, because each grid is responsible for predicting only one target. Thus, YOLOv2 (Redmon and Farhadi, 2016) was proposed to solve this problem which uses a more robust backbone Darknet-19 (Redmon and Farhadi, 2016) to extract more representative features. However, YOLOv2's accuracy is still far inferior to two-stage detectors. Finally, YOLOv3 (Redmon and Farhadi, 2018) was proposed to enhance detection performance further. In YOLOv3, a novel backbone Darknet-53 (Redmon and Farhadi, 2018) was proposed to extract features, by which on the premise of maintaining the speed advantage, the accuracy is improved and especially the detection performance for small objects is enhanced. So far, YOLOv3 has become a mainstream representative of one-stage detectors (Zhao et al., Nov. 2019), but it has the disadvantage of low positioning accuracy (Zhao et al., Nov. 2019). Thus, many improvements have been proposed, e.g. SSD (Liu et al., 2015), RetinaNet (Lin et al., 2017), etc. SSD combined the advantages of YOLO and Faster R-CNN to achieve a good balance between accuracy and speed. RetinaNet proposed focal loss (Lin et al., 2017) to solve extreme imbalance between positive and negative samples, which can improve accuracy. However, their detection speed is slightly inferior to YOLO (Zhang et al., 2019; Wang et al., 2018). Overall, the above one-stage detectors have faster detection speed than two-stage detectors, because they regress targets' coordinates without generating ROIs. However, their detection accuracy is often inferior to two-stage detectors for their simple and crude

grid division mechanism.

So far, many scholars from the SAR remote sensing community have appiled these one-stage detectors for SAR ship detection. Chang et al. (Chang et al., 2019) proposed a reduced YOLOv2 based on high performance GPU computing for real-time SAR ship detection. They thought there are many redundant convolution layers in YOLOv2, but this idea is only a subjective assumption. Although their experiments proved the validity of the idea, there was still a lack of strong theory. Zhang et al. (Zhang et al., 2019) used Depthwise Separable Convolution Neural Network (DS-CNN) for high-speed SAR ship detection based on YOLOv3. Multi-scale detection mechanism, concatenation mechanism and anchor box mechanism are used to improve detection accuracy. However, their model still contains patial heavy traditional convolution layer, declining detection speed. Li et al. (Li et al., 2019) improved the raw SSD by upsampling feature fusion for both ship detection and direction estimation. Moreover, they also adopted the mechanism of increasing training weight of difficult samples, making learning more effective. However, their accuracy improvement is not obvious, but brings more detection delay. Wang et al. (Wang et al., 2018) studied transfer learning based on SSD to improve detection accuracy given limited training samples, but their model still has a modest false alarm probability for inshore ships, declining the overall detection performance. After that, Wang et al. also (Wang et al., 2019) proposed a modified SSD for SAR ship detection. Considering that the bounding-box sizes that include the ships are relatively small, they removed some high-level layers to construct the new mode, by which the accuracy is not affected too much meanwhile increasing speed. Yang et al. (Yang et al., 2020) performed deep multi-scale feature fusion on SSD by DarkNet-53 (Redmon and Farhadi, 2018) to detect multi-scale ships in complex SAR images. Moreover, they also used focal loss to slove the huge imbalance between easy samples and hard ones. However, their model missed many small ships. Liu et al. (Liu et al., 2020) combined the characteristics of SAR image with less feature information, adopted the idea of multi-feature layer fusion and proposed a more appropriate loss function to improve ship detection performance based on RetinaNet. Moreover, they also used data augmentation and transfer learning to improve the robustness and convergence speed of model. However, their model sacrificed the detection accuracy with more computational costs. Wang et al. (Wang et al., 2019) used RetinaNet for automatic ship detection in multi-resolution Gaofen-3 imagery. However, those ships near the harbor are rather difficult for their model to detect. Mao et al. (Mao et al., 2020) proposed an efficient and low-cost SAR ship detector by simplifying U-Net (Ronneberger et al., 2015) that no longer depends on region proposals and also avoid configuring any anchors, more simply and efficiently. However, the detection accuracy is sacrificed and some ships parking side by side on harbor are often miss-detected in their model. In addition, their detection score threshold are set as 0.2, which may lead to a high false alarm probability for inshore scene. Gao et al. (Gao et al., 2019) proposed a novel Split Convolution Block (SCB) to enhance the feature representation of small targets and embedded a spatial attention block (SAB) into feature pyramid network to reduce the loss of spatial information, greatly improving detection accuracy, but their model increased inference time.

To sum up, the above modern-abstract-based methods or deep learning methods are rarely involved with the analysis of traditional SAR information or ship features. Instead, deep neural networks can automatically extract abstract features to avoid huge feature engineering. More importantly, these deep neural networks are useful in various fields (including ImageNet and SAR), which can adaptively learn the target features and scarcely consider the influence of data domain differences. Therefore, recent years, more and more scholars (Zhang and Zhang, 2019; Zhang et al., 2019; Wang et al., 2018; Ai et al., 2019; Zhang, 2020; Gui et al., 2019; Wei et al., 2020; Gao et al., 2019; Zhang et al., 2019; Li et al., 2017; Li et al., 2019; Li et al., 2019; Li et al., 2019; Jiao et al., 2018; Huang et al., 2016; Cui et al., Nov. 2019; Deng et al., 2018; Deng et al., 2019; Lin et al., 2019; Zhao et al., 2019; Zhao et al., 2018; Chen et al., 2019; Chen et al., 2020; Kang et al., 2017; Chang et al., 2019; Wang et al., 2019; Yang et al., 2020; Liu et al., 2020; Wang et al., 2019; Mao et al., 2020) in the SAR ship detection community are devoting themselves to ship detection based on deep learning.

In addition, nowadays, two-stage detectors has higher accuracy but lower speed, and on the contrary, one-stage detectors has higher speed but lower accuracy (Zhao et al., Nov. 2019; Cui et al., Nov. 2019). Therefore, how to achieve faster ship detection speed under the premise of maintaining high accuracy is still an urgent problem to be addressed.

## 3. HyperLi-Net

### 3.1. Network architecture

Fig. 3 shows the overall network architecture of HyperLi-Net. Table 1 shows the detailed description of HyperLi-Net. Table 2 shows the parameter configuration of HyperLi-Net.

From Fig. 3 and Table 1, HyperLi-Net mainly consists of 5 types of externel modules that are used to improved detection accuracy (i.e., MRF-Module, DC-Module, CSA-Module, FF-Module and FP-Module) and 4 backbones that are used to extract features (i.e., Backbone-1, Backbone-2, Backbone-3 and Backbone-4).

Inspired by Inception structure (Szegedy et al., 2014), we designed MRF-Module that is at the imput-end of HyperLi-Net. In MRF-Module, three types of convolution kernels ($1 \times 1$,$3 \times 3$,$5 \times 5$) are used (i.e. Separa-Conv-1, Separa-Conv-2 and Separa-Conv-3 in Table 1), which is same as the work of Szegedy et al. (Szegedy et al., 2014) who has confirmed the effectiveness of such practice. However, MRF-Module is slightly different from their raw Inception module (Szegedy et al., 2014), because we removed the MaxPooling layer that may loss much image information (Long et al., 1411). To be clear, we do not consider larger kernels (e.g. $7 \times 7$, $9 \times 9$, $11 \times 11$, etc.) to add more convolution layers. For one thing, larger kernels will result in more parameters and calculations. For another thing, we will propose extra DC-Module to further expand receptive field without increasing parameters and calculations. Moreover, the strides of the above three convolution layers in MRF-Module are all set as 2 in order to achieve feature dimension reduction (i.e. from $L$ to $L/2$). The detailed introduction of MRF-Module's design ideas can be found in Section 3.3.1 and its ablation study can be found in Table 14 of Section 6.1.1.

Inspired by Multi-Scale Context Aggregation (Yu and Koltun, 2015), we designed DC-Module to expand MRF-Module receptive field without increasing too much computation cost. DC-Module is also at the imput-end of HyperLi-Net that is parallel to MRF-Module. Similar to MRF-Module, three types of convolution kernels ($1 \times 1$,$3 \times 3$,$5 \times 5$) are used in DC-Module (i.e. Separa-Conv-4, Separa-Conv-5 and Separa-Conv-6 in Table 1). Similarly, the strides of the above three convolution layers in DC-Module are also all set as 2 in order to achieve feature dimension reduction (i.e. from $L$ to $L/2$). In addition, the dialted rate $d$ in DC-Module is set as 2 according to our ablation study in Section 6.1.2. The detailed introduction of DC-Module's design ideas can be found in Section 3.3.2 and its ablation study can be found in Table 15 of Section 6.1.2.

Inspired by YOLOv3 (Redmon and Farhadi, 2018), we designed 4 backbones and each backbone reduces the feature maps' size by half (from $L/4$ to $L/32$) in order to learn deep semantic features, where $L$ denotes the image size that is set as 160 according to our ablation study in Table 4 of Section 4.4.1. Therefore, the feature maps' size is 40 ($L/4$) in Backbone-1, 20 ($L/8$) in Backbone-2, 10 ($L/16$) in Backbone-3, and 5 ($L/32$) in Backbone-4. We cannot set more backbones behind Backbone-4 because the feature maps' size of Backbone-4 is 5 that can not be divided by 2. In HyperLi-Net, Backbone-1 is only used to extract shallow features but it is not responsible for final detection due to its weak feature representation in the network front-end. Backbone-2, Backbone-3 and Backbone-4 are responsible for both feature extraction

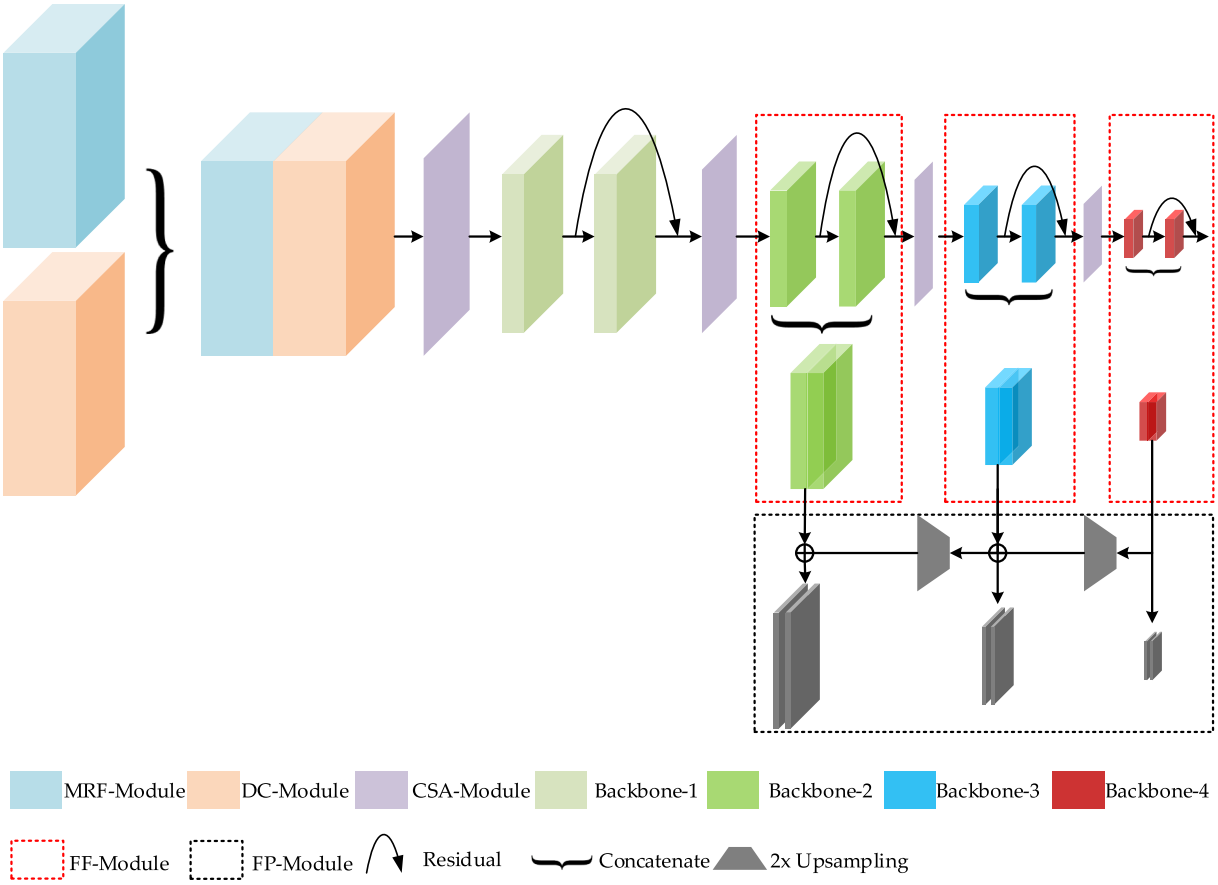



**Fig. 3.** Network Architecture of HyperLi-Net.

and final detection. There are $Z$ layers in every backbone where Z is a hyper parameter that is set as 2 according to our ablation study in Table 5 of Section 4.4.2. Therefore, in each backbone, the first layer's stride is set as 2 to reduce the feature maps' size by half, and we do not use MaxPooling to complete such because it may loss much image information (Long et al., 1411); i.e., HyperLi-Net belongs to a Fully Convolutional Network (FCN) (Long et al., 1411). Finally, inspired by ResNet (He et al., 2015), we also adopt Residual Learning to improve detection accuracy in backnones according to our ablation study in Table 6 of Section 4.4.3. As a result, there are 4 residual path in total in HyperLi-Net, respectively corresponding to 4 backbones.

Inspired by CBAM (Woo et al., 2018), we designed CSA-Module. We set 4 CSA-Modules because there are 4 backbones (i.e., Backbone-1, Backbone-2, Backbone-3 and Backbone-4) and these 4 CSA-Modules are placed at the input-end of each backbone. These 4 CSA-Modules are almost the same as CBAM (Woo et al., 2018) except that we replaced the original Trad-Conv with Separa-Conv. In addition, CSA-Module does not change the feature maps' size. The detailed introduction of CSA-Module's design ideas can be found in Section 3.3.3 and its ablation study can be found in Table 17 of Section 6.1.3.

Inspired by DenseNet (Huang et al., 2016), we designed FF-Module. There are 3 FF-Modules in HyperLi-Net, because there are 3 backbones used for detection, i.e., Backbone-2, Backbone-3 and Backbone-4 (as mentioned before, Backbone-1 is only used to extract shallow features but it is not responsible for final detection due to its weak feature representation in the network front-end). The detailed introduction of FF-Module's design ideas can be found in Section 3.3.4 and its ablation study can be found in Table 18 of Section 6.1.4.

Inspired by FPN (Lin et al., 2017), we designed FP-Module. FP-Module is at the end of HyperLi-Net whose feature maps are directly used for the final detection. Moreover, we also adopted upsampling to achieve scale fusion (Upsampling-1, Upsampling-2), whose times are both set as 2, respectively from $L/32$ to $L/16$ and from $L/16$ to $L/8$. In Table 1, Add-1 and Add-2 denote elementwise addition. Inspired by YOLOv3 (Redmon and Farhadi, 2018), we set the dimension of convolution layers (i.e. Separa-Conv-15, Separa-Conv-16 and Separa-Conv-17 in Table 1) as $1 \times 1 \times 18$@1. Namely, we set their kernel sizes as 1 to flattening feature maps to avoid heavy full connection layers with huge parameter quantity, inspired by Network in Network (Lin et al., 2013), we set their kernel number as 18 to adapt to ship detection task (See the detailed reason in Section 3.3.5.), and we set their srides as 1 because the feature maps' size of Backbone-4 is 5 that can not be divided by 2. The detailed introduction of FP-Module's design ideas can be found in Section 3.3.5 and its ablation study can be found in Table 19 of Section 6.1.5.

From Table 1, HyperLi-Net also adopted five types of internal mechanisms that are used to improve detection speed (i.e., RF-Model, S-Kernel, N-Channel, Separa-Conv and BN-Fusion).

Inspired by YOLO (Redmon et al., 2015; Redmon and Farhadi, 2016; Redmon and Farhadi, 2018), we adopted the idea of RF-Model to design HyperLi-Net, because as mentioned before, RF-Model is universally faster than R-Model. From Fig. 3, there are no RPNs used to extract ROIs in HyperLi-Net, so training time can be shortened and detection speed can be improved. The detailed introduction of RF-Model's design ideas can be found in Section 3.4.1 and its ablation study can be found in Section 6.2.1.

Inspired by VGG (Simonyan and Zisserman, 2014), we used small kernels to construct convolution layers, i.e., S-Kernel. From Table 1 and Table 2, the kernels' sizes $k$ in backbones are all set as 3 to reduce the number of parameters, which is the same as VGG. The detailed introduction of S-Kernel's design ideas can be found in Section 3.4.2 and its ablation study can be found in Table 21 of Section 6.2.2.

Inspired by EfficentNet (Tan and Le, 1905), we also used narrower channels to construct convolution layers, i.e., N-Channel. From Table 1 and Table 2, the channel widths of all convolution layers $N_{kernel}$ are set as 32, which is universally fewer than other object detectors (e.g.,

**Table 1**
Detailed Description of HyperLi-Net.

| Block | Name | Layer @ Stride | Output Size | Ablation Study |
|---|---|---|---|---|
| MRF-Module | Separa-Conv-1 | $1 \times 1 \times N_{kernel}$ @2 | $(L/2) \times (L/2) \times N_{kernel}$ | Section 6.1.1 |
| | Separa-Conv-2 | $3 \times 3 \times N_{kernel}$ @2 | $(L/2) \times (L/2) \times N_{kernel}$ | |
| | Separa-Conv-3 | $5 \times 5 \times N_{kernel}$ @2 | $(L/2) \times (L/2) \times N_{kernel}$ | |
| | Concate-1 | – | $(L/2) \times (L/2) \times 3N_{kernel}$ | |
| DC-Module | Separa-Conv-4 | $1 \times 1 \times N_{kernel}$ @2, $d = 2$ | $(L/2) \times (L/2) \times N_{kernel}$ | Section 6.1.2 |
| | Separa-Conv-5 | $3 \times 3 \times N_{kernel}$ @2, $d = 2$ | $(L/2) \times (L/2) \times N_{kernel}$ | |
| | Separa-Conv-6 | $5 \times 5 \times N_{kernel}$ @2, $d = 2$ | $(L/2) \times (L/2) \times N_{kernel}$ | |
| | Concate-2 | – | $(L/2) \times (L/2) \times 3N_{kernel}$ | |
| | Concate-3 | – | $(L/2) \times (L/2) \times 6N_{kernel}$ | |
| CSA-Module | CSA-Module-1 | – | $(L/2) \times (L/2) \times N_{kernel}$ | Section 6.1.3 |
| | CSA-Module-2 | – | $(L/4) \times (L/4) \times N_{kernel}$ | |
| | CSA-Module-3 | – | $(L/8) \times (L/8) \times N_{kernel}$ | |
| | CSA-Module-4 | – | $(L/16) \times (L/16) \times N_{kernel}$ | |
| Backbone-1 | Separa-Conv-7 | $k \times k \times N_{kernel}$ @2 | $(L/4) \times (L/4) \times N_{kernel}$ | – |
| | Separa-Conv-8 | $k \times k \times N_{kernel}$ @1 | $(L/4) \times (L/4) \times N_{kernel}$ | |
| | Residual-1 | – | $(L/4) \times (L/4) \times N_{kernel}$ | |
| Backbone-2 | Separa-Conv-9 | $k \times k \times N_{kernel}$ @2 | $(L/8) \times (L/8) \times N_{kernel}$ | |
| | Separa-Conv-10 | $k \times k \times N_{kernel}$ @1 | $(L/8) \times (L/8) \times N_{kernel}$ | |
| | Residual-2 | – | $(L/8) \times (L/8) \times N_{kernel}$ | |
| Backbone-3 | Separa-Conv-11 | $k \times k \times N_{kernel}$ @2 | $(L/16) \times (L/16) \times N_{kernel}$ | |
| | Separa-Conv-12 | $k \times k \times N_{kernel}$ @1 | $(L/16) \times (L/16) \times N_{kernel}$ | |
| | Residual-3 | – | $(L/16) \times (L/16) \times N_{kernel}$ | |
| Backbone-4 | Separa-Conv-13 | $k \times k \times N_{kernel}$ @2 | $(L/32) \times (L/32) \times N_{kernel}$ | |
| | Separa-Conv-14 | $k \times k \times N_{kernel}$ @1 | $(L/32) \times (L/32) \times N_{kernel}$ | |
| | Residual-4 | – | $(L/32) \times (L/32) \times N_{kernel}$ | |
| FF-Module | FF-Module-1 | – | $(L/8) \times (L/8) \times 2N_{kernel}$ | Section 6.1.4 |
| | FF-Module-2 | – | $(L/16) \times (L/16) \times 2N_{kernel}$ | |
| | FF-Module-3 | – | $(L/32) \times (L/32) \times 2N_{kernel}$ | |
| FP-Module | Separa-Conv-15 | $1 \times 1 \times 18$ @1 | $(L/32) \times (L/32) \times 18$ | Section 6.1.5 |
| | 2 × Upsampling-1 | – | $(L/16) \times (L/16) \times 18$ | |
| | Add-1 | – | $(L/16) \times (L/16) \times 18$ | |
| | Separa-Conv-16 | $1 \times 1 \times 18$ @1 | $(L/16) \times (L/16) \times 18$ | |
| | 2 × Upsampling-2 | – | $(L/8) \times (L/8) \times 18$ | |
| | Add-2 | – | $(L/8) \times (L/8) \times 18$ | |
| | Separa-Conv-17 | $1 \times 1 \times 18$ @1 | $(L/8) \times (L/8) \times 18$ | |

**Table 2**
Parameter Configuration of HyperLi-Net.

| Hyper Parameter | Definition | Value | Ablation Study |
|---|---|---|---|
| $L$ | Input Image Size | 160 | Section 4.4.1 |
| $Z$ | Layer Number in Backbones | 2 | Section 4.4.2 |
| *Residual* | Residual Learning in Backbones | ✓ | Section 4.4.3 |
| *Pre-Training* | Pre-Training on ImageNet | ✗ | Section 4.4.4 |
| *CPU and GPU* | Train on CPUs and GPUs | ✓ | Section 4.4.5 |
| $d$ | Dialted Rate in DC-Module | 2 | Section 6.1.2 |
| $k$ | Kernel Size in S-Kernel | 3 | Section 6.2.2 |
| $N_{kernel}$ | Channel Width in N-Channel | 32 | Section 6.2.3 |

Faster R-CNN, YOLO, SSD, etc.) bringing fewer parameters. The detailed introduction of N-Channel's design ideas can be found in Section 3.4.3 and its ablation study can be found in Table 22 of Section 6.2.3.

Inspired by MobileNet (Howard et al., 2017), we used Separa-Conv to replace Trad-Conv to reduce model size. From Table 1, all convolution layers are set as Separa-Conv. The detailed introduction of Separa-Conv's design ideas can be found in Section 3.4.4 and its ablation study can be found in Table 23 of Section 6.2.4.

Inspired by our previous work (Zhang et al., 2019); we adopted BN-Fusion. We found that when training, BN is helpful for learning acceleration and accuracy improvement in deep neural networks, but when performing inference, it does nothing but increases computation costs. Thus, we fused BN into Separa-Conv by beforehand mathematical calculation in HyperLi-Net, i.e., BN-Fusion, which can improve detection speed. The detailed introduction of BN-Fusion's design ideas can be found in Section 3.4.5 and its ablation study can be found in Table 24 of Section 6.2.5.

### 3.2. Execution process

Fig. 4 shows HyperLi-Net's execution process. Algorithm 1 is

**Table 3**
Detailed Descriptions of Three Datasets. # denotes number.

| | SSDD | Gaofen-SSDD | Sentinel-SSDD |
|---|---|---|---|
| **Sensors** | Sentinel-1, RadarSat-2, TerraSAR-X | Gaofen-3 | Sentinel-1 |
| **Place** | Visakhapatnam, India; Yantai, China | Bohai Sea, China; Zhejiang, China | Korean Strait, South Korea; Tokyo, Japan |
| **Polarization** | HH, VV, HV, VH | HH, VV, HV, VH | HH, VV, HV, VH |
| **Resolution** | 1 m–10 *m* | 3 m, 5 m, 8 m, 10 m, 25 *m* | from 1.7 m × 4.3 m to 3.6 m × 4.9 m, 20 m × 22 *m* |
| **Scene** | Inshore, offshore | Inshore, offshore | Inshore, offshore |
| **#(Images)** | 1,160 | 20,000 | 20,000 |
| **Avg. size of Images** | 500 × 500 | 160 × 160 | 160 × 160 |
| **#(Ships)** | 2,358 | 29,250 | 22,925 |
| **Avg. #(Ships) per image** | 2.03 | 1.46 | 1.17 |
| **Smallest ships** | 7 × 7 | 4 × 5 | 1 × 17 |
| **Biggest ships** | 211 × 298 | 81 × 130 | 57 × 49 |

**Table 4**
Ablation Study on $L$. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Image Size ($L$) | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Training Time |
|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | |
| 32 | 57.72% | 47.28% | 15.65% | 57.72% | 84.35% | 51.66% | 4.11 ms | 243 | 1.11 h |
| 64 | 85.33% | 14.67% | 12.29% | 85.33% | 87.71% | 83.98% | 4.27 ms | 234 | 1.67 h |
| 96 | 92.93% | 7.07% | 10.47% | 92.93% | 89.53% | 92.12% | 4.32 ms | 231 | 2.22 h |
| 128 | 94.02% | 5.98% | 6.99% | 94.02% | 93.01% | 93.48% | 4.41 ms | 227 | 2.78 h |
| **160** | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 3.33 h |
| 192 | 95.65% | 4.35% | 7.37% | 95.65% | 92.63% | 94.26% | 4.63 ms | 216 | 4.44 h |
| 224 | 93.48% | 6.52% | 8.99% | 93.48% | 91.01% | 92.46% | 4.79 ms | 209 | 5.56 h |
| 256 | 94.57% | 5.43% | 8.42% | 94.57% | 91.58% | 92.89% | 4.98 ms | 201 | 7.22 h |
| 288 | 96.20% | 3.80% | **6.35%** | 96.20% | **93.65%** | 95.70% | 5.24 ms | 191 | 8.89 h |
| 320 | 94.57% | 5.43% | 6.45% | 94.57% | 93.55% | 94.28% | 5.32 ms | 188 | 11.67 h |

**Table 5**
Ablation Study on Z. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| $Z$ | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| 1 | 94.57% | 5.43% | 12.56% | 94.57% | 87.44% | 92.76% | 4.28 ms | 234 | 96,330 | 189,418 | 0.62 MB |
| 2 | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| 3 | 92.93% | 7.07% | 7.07% | 92.93% | 92.93% | 92.27% | 4.65 ms | 215 | 111,178 | 217,722 | 0.77 MB |
| 4 | 93.48% | 6.52% | **6.01%** | 93.48% | **93.99%** | 92.87% | 4.90 ms | 204 | 118,602 | 231,874 | 0.84 MB |
| 5 | **96.74%** | **3.26%** | 7.77% | **96.74%** | 92.23% | 95.74% | 5.10 ms | 196 | 126,026 | 246,026 | 0.91 MB |

**Table 6**
Ablation Study on Residual. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Residual Learning? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 95.65% | 4.35% | 9.74% | 95.65% | 90.26% | 95.13% | 4.48 ms | 223 | 103,754 | 203,570 | 0.69 MB |
| ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | **90.36%** | **96.08%** | 4.51 ms | 222 | **103,754** | **203,570** | **0.69 MB** |

HyperLi-Net's pseudo code that processes a SAR image to be detected. In Algorithm 1, the input SAR image denotes $\boldsymbol{I} \in \mathbb{R}^{L\times L\times 3}$, where $L$ is the image size, 3 is the channel number (SAR images are converted RGB images with 3 channels.), $\otimes$ denotes convolution and © denotes concatenate. HyperLi-Net has 3 outputs, i.e., $F_{L/32}$ of Separa-Conv-15, $F_{L/16}$ of Separa-Conv-16 and $\boldsymbol{F}_{L/8}$ of Separa-Conv-17.

In addition, the pseudo code of training HyperLi-Net is the same as YOLO (Redmon et al., 2015; Redmon and Farhadi, 2016; Redmon and Farhadi, 2018), so considering limited pages, we will not redundantly show it any more.

From Fig. 4 and Algorithm 1, for a SAR image, the overall execution workflow of HyperLi-Net is as follows:

Step 1.Input a SAR image whose dimension is $L \times L \times 3$ (in HyperLi-Net, $L = 160$).

MRF-Module and DC-Module parallelly process this SAR image. Their outputs' dimension is $80 \times 80 \times 96$.

Step 2.Concatenate the outputs of MRF-Module and DC-Module as the next layer's input.

Step 3.Perform the 1st channel attention and spatial attention (CSA-Module-1).

Step 4.Backbone-1 performs feature extraction (including residual learning). Its outputs' dimension is $40 \times 40 \times 32$.

Step 5.Perform the 2nd channel attention and spatial attention (CSA-Module-2).

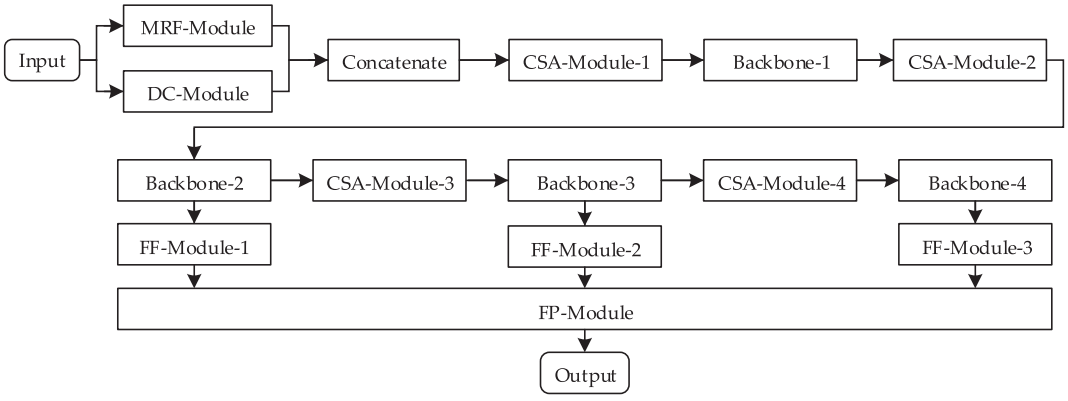



**Fig. 4.** Execution Process of HyperLi-Net.

Step 6.Backbone-2 performs feature extraction. Its outputs' dimension is 20 × 20 × 32. Step 7.The outputs of Separa-Conv-9 and Separa-Conv-10 are inputted into FF-Module-1 to achieve feature fusion for small detection scale ($L/8$).

Step 8.Perform the 3rd channel attention and spatial attention (CSA-Module-3).

Step 9.Backbone-3 performs feature extraction. Its outputs' dimension is 10 × 10 × 32. The outputs of Separa-Conv-11 and Separa-Conv-12 are inputted into FF-Module-2 to achieve feature fusion for medium detection scale ($L/16$).

Step 10.Perform the 4th channel attention and spatial attention (CSA-Module-4).

Step 11.Backbone-4 performs feature extraction. Its outputs' dimension is 5 × 5 × 32. The outputs of Separa-Conv-13 and Separa-Conv-14 are inputted into FF-Module-3 to achieve feature fusion for big detection scale ($L/32$).

Step 12.Perform FP-Module operation to get the final output, i.e., $\boldsymbol{F}_{L/32}$ of Separa-Conv-15, $\boldsymbol{F}_{L/16}$ of Separa-Conv-16 and $\boldsymbol{F}_{L/8}$ of Separa-Conv-17.

**Algorithm 1:** HyperLi-Net processes a SAR image.

**Input:** $I \in \mathbb{R}^{L\times L\times 3}$

**Output:** $F_{L/32} \in \mathbb{R}^{(L/32)\times(L/32)\times 18}, F_{L/16} \in \mathbb{R}^{(L/16)\times(L/16)\times 18}, F_{L/8} \in \mathbb{R}^{(L/8)\times(L/8)\times 18}$

**Begin**

**do** MRF-Module Operation

$A \leftarrow I \otimes K_{Separa-Conv-1}, B \leftarrow A \otimes K_{Separa-Conv-2}, CB \otimes K_{Separa-Conv-3}, D \leftarrow A©B©C$

**end**

**do** DC-Module Operation

$A' \leftarrow I \otimes K_{Separa-Conv-4}$,

$B' \leftarrow A' \otimes K_{Separa-Conv-5}, C' \leftarrow B' \otimes K_{Separa-Conv-6}, D' \leftarrow A'©B'©C'$

**end**

$F \leftarrow D©D'$

**do** CSA-Module-1 Operation

$F \leftarrow$ CSA - Module - 1($F$)

**end**

**do** Backbone-1 Operation

$F_1 \leftarrow F \otimes K_{Separa-Conv-7}, F_2 \leftarrow F_1 \otimes K_{Separa-Conv-8}, F \leftarrow F_1 + F_2$

**end**

**do** CSA-Module-2 Operation

$F \leftarrow$ CSA - Module - 2($F$)

**end**

**do** Backbone-2 Operation

$F_3 \leftarrow F \otimes K_{Separa-Conv-9}, F_4 \leftarrow F_3 \otimes K_{Separa-Conv-10}, F \leftarrow F_3 + F_4$

**end**

**do** CSA-Module-3 Operation

$F \leftarrow$ CSA - Module - 3($F$)

**end**

**do** Backbone-3 Operation

$F_5 \leftarrow F \otimes K_{Separa-Conv-11}, F_6 \leftarrow F_5 \otimes K_{Separa-Conv-12}, F \leftarrow F_5 + F_6$

**end**

**do** CSA-Module-4 Operation

$F \leftarrow$ CSA - Module - 4($F$)

**end**

**do** Backbone-4 Operation

$F_7 \leftarrow F \otimes K_{Separa-Conv-13}, F_8 \leftarrow F_7 \otimes K_{Separa-Conv-14}, F \leftarrow F_7 + F_8$

**end**

**do** FF-Module-1 Operation

$F_{L/8} \leftarrow F_3©F_4$

**end**

**do** FF-Module-2 Operation

$F_{L/16} \leftarrow F_5©F_6$

**end**

**do** FF-Module-3 Operation

$F_{L/32} \leftarrow F_7©F_8$

**end**

**do** FP-Module Operation

$P \leftarrow UpSampling^{2\times}(F_{L/32}), F_{L/32} \leftarrow F_{L/32} \otimes K_{Separa-Conv-15}$,

$F_{L/16} \leftarrow F_{L/16} + P, F_{L/16} \leftarrow F_{L/16} \otimes K_{Separa-Conv-16}$

$Q \leftarrow UpSampling^{2\times}(F_{L/16})$,

$F_{L/8} \leftarrow F_{L/8} + Q, F_{L/8} \leftarrow F_{L/8} \otimes K_{Separa-Conv-17}$

**end**

**end**

### 3.3. Five external modules

#### 3.3.1. Module 1: MRF-Module

MRF-Module has multi receptive fields from the use of three different sizes kernels (1 × 1, 3 × 3, 5 × 5) that can compensate for S-Kernel's defect of limited receptive field. These three types of kernels are set at the input-end of HyperLi-Net, which can extract SAR images' adequate information, as is shown in Fig. 5a.

From Fig. 5a, the receptive field becomes larger with the increase of kernel size. Thus, if several kernels with different receptive fields are simultaneously utilized in HyperLi-Net, more comprehensive information and more powerful features will be extracted. For example, some small ships can be observed by 1 × 1 kernel, some medium ships can be observed by 3 × 3 kernel, and some big ships can be observed by 5 × 5 kernel. Finally, the convolution outputs $O_{1\times1}$, $O_{3\times3}$, and $O_{5\times5}$ from 1 × 1, 3 × 3 and 5 × 5 kernels are concatenated together as the input of next layer, which can be expressed as:

$$O_{MRF-Module} = O_{1\times1}©O_{3\times3}©O_{5\times5} \tag{1}$$

where $O_{MRF\text{-}Module}$ represents the output of MRF-Module and © denotes the concatenate operation.

The ablation study of MRF-Module is discussed experimentally in Section 6.1.1.

#### 3.3.2. Module 2: DC-Module

DC-Module is used for expanding receptive field further on the basis of MRF-Module, as is shown in Fig. 5b. From Fig. 5b, dilated convolution operation can increase receptive field by injecting regular holes into the standard convolution operation (Yu and Koltun, 2015), which can be expressed as:

$$RF_{dilated} = d{\cdot}RF_{standard} - 1 \tag{2}$$

where $d$ is the dilated rate, $RF_{standard}$ is the receptive field of standard convolution, and $RF_{dilated}$ is the receptive field of dilated convolution.

For example, in Fig. 5b, if $d = 2$, the receptive field of dilated 1 × 1 kernel remains unchanged, the receptive field of dilated 3 × 3 kernel is the same as standard 5 × 5 kernel, and the receptive field of dilated 5 × 5 kernel is the same as standard 9 × 9 kernel. In DC-Module, according to our experimental analysis, we set $d = 2$, because for one thing, the value of $d$ cannot be too large, otherwise the extracted features will have serious discontinuities (too many holes) leading to missing features (Yu and Koltun, 2015). For another thing, our experiments show that the detection accuracy reaches the best when $d = 2$. Finally, the convolution outputs $O'_{1\times1}$, $O'_{3\times3}$, and $O'_{5\times5}$ are concatenated together, which can be expressed as:

$$O_{DC-Module} = O'_{1\times1}©O'_{3\times3}©O'_{5\times5} \tag{3}$$

where $O'_{1\times1}$ represents the outputs of dilated 1 × 1 kernel, $O'_{3\times3}$ represents that of dilated 3 × 3 kernel, $O'_{5\times5}$ represents that of dilated 5 × 5 kernel, and $O_{DC\text{-}Module}$ represents that of DC-Module.

Moreover, compared with standard convolution, dilated convolution increases neither parameter quantity nor calculation quantity, because kernel size remains unchanged which means that the number of pixels involved in convolution operarion is constant.

The ablation study of DC-Module is discussed experimentally in Section 6.1.2.

#### 3.3.3. Module 3: CSA-Module

CSA-Module can improve the representativeness of useful features, that is, focus on important features and suppress inessential features, which is beneficial to information flow in networks (Woo et al., 2018). Compared with SENet (Hu et al., 2017) that only pays attention to channel, CSA-Module has better detection performance for the attention of both channel and spatial.

Fig. 6a shows the baisc architecture of CSA-Module. From Fig. 6a, a

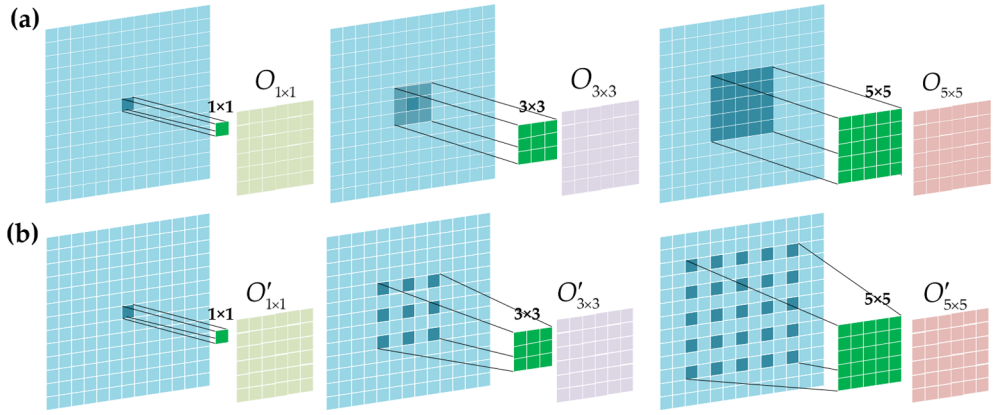



**Fig. 5.** Multi-Receptive-Field Module (MRF-Module) and Dilated Convolution (Module DC-Module). **(a)** MRF-Module; **(b)** Module DC-Module.

CSA-Module consists of a CA-Module and a SA-Module, inspired from the experience from Reference (Woo et al., 2018). It can be expressed as:

$$F' = M_{channel}(F) \odot F,\ F'' = M_{spatial}(F') \odot F' \tag{4}$$

where $\boldsymbol{F}$ is the input feature map, $\boldsymbol{F'}$ is the output feature map processed by CA-Module, $\boldsymbol{F''}$ is the output feature map processed by SA-Module, $M_{channel}()$ denotes processed by CA-Module, $M_{spatial}()$ denotes processed by SA-Module and $\odot$ denotes elementwise multiplication.

**CA-Module** As a common sense in the deep learning community, a feature matrix will be obtained after images processed by several convolution layers, where the channel number of this matrix is the number of convolution kernels $N_{kernel}$ ($N_{kernel}$ = 32 in HyperLi-Net). Then, these 32 channels in HyperLi-Net are not all valuable for information transmission. Thereinto, some channels may play an extremely significant role in the whole detection process, but some channels may have little value, even may have a negative effect in some extreme circumstances. Given this, CA-Module can successfully solve this problem. In fact, CA-Module in popular understanding is to retain effective channels meanwhile suppress invalid channels.

Fig. 6b shows the structure of CA-Module which can be expressed as:

$$M_{channel}(F) = sigmod\{MLP[GAvgPooling(F)] + MLP[GMaxPooling(F)]\} \tag{5}$$

where $\boldsymbol{F}$ is the input feature map of CA-Module, *GAvgPooling*() denotes global average pooling, *GmaxPooling*() denotes global max pooling, *MLP*() denotes multi-layer perceptron, and *sigmod*($x$) is an activation function defined by:

$$sigmod(x) = 1/(1 + e^{-x}) \tag{6}$$

From Fig. 6b, the output of CA-Module can effectively describe the importance level of $N$ channels. Briefly, $\boldsymbol{F}_{channel} = (\hat{1}_{\pm 1}, \hat{1}_{\pm 2}, \cdots, \hat{1}_{\pm N})^T$, where $\hat{1}_{\pm i}$ ($i$ = 1,2,…,$N$-1,$N$) represents the importance level of the $i$-th channel. Finally, the original input feature $\boldsymbol{F}$ dot-multiplies this channel attention vector $F_{channel}$ to obtain the final output channel-refined feature $\boldsymbol{F'}$.

**SA-Module** Also as a common sense, for a SAR image, the different regions of the image in space are also not all valuable for information transmission. For example, for a ship in port under a complex background, the port facilities in images are bound to have obvious side effects on ship detection. Therefore, we need to concentrate on valuable regions and suppress worthless ones in SAR images. Given this, similar to the human eye's attention principle, SA-Module can seek out important and valuable information in space. Fig. 6c shows the structure of SA-Module which can be expressed as:

$$M_{spatial}(F') = sigmod\{f^{7\times 7}[GAvgPooling(F'), GMaxPooling(F')]\} \tag{7}$$

where $\boldsymbol{F'}$ is the input feature map which has been processed by previous CA-Module, and $f^{7\times 7}$ represents Separa-Conv operation with a 7 × 7 size kernel, inspired by the experience of Reference (Woo et al., 2018).

From Fig. 6c, the output of SA-Module can effectively describe importance level in $H \times W$ space, where $H$ is the height of feature map and $W$ is the width of feature map. Briefly, $\boldsymbol{F}_{spatial} = (\beta_{i,j})_{1\le i\le H, 1\le j\le W}$, where $\beta_{i,j}$ represents the importance level of the region with coordinate ($i$,$j$) in the $H \times W$ space. Finally, the input channel-refined feature $\boldsymbol{F'}$ dot-multiplies this spatial attention vector $\boldsymbol{F}_{spatial}$ to obtain the final output of CSA-Module $\boldsymbol{F''}$.

The ablation study of CSA-Module is discussed experimentally in Section 6.1.3.

### 3.3.4. Module 4: FF-Module

FF-Module can realize shallow and deep features fusion, making the final features used for detection more robust and contextual. Fig. 7 shows structure of FF-Module, which can be expressed as:

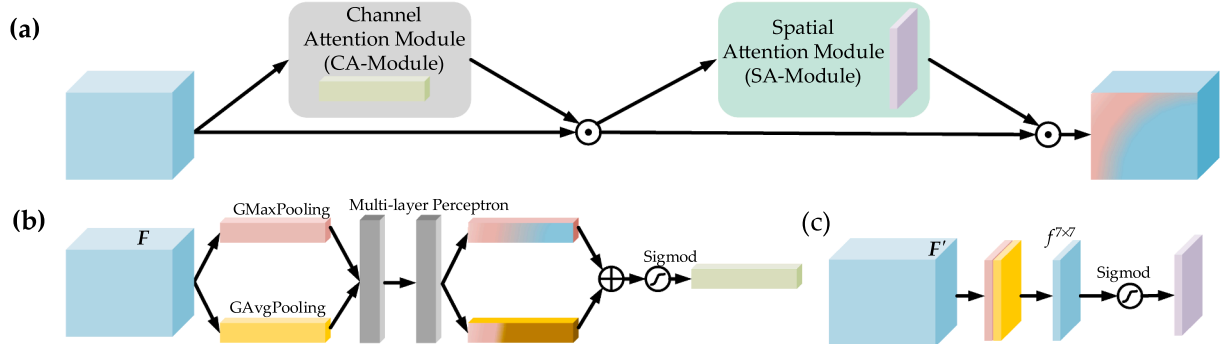



**Fig. 6.** Channel and Spatial Attention Module (CSA-Module). **(a)** Overall Architecture; **(b)** CA-Module; **(c)** SA-Module.

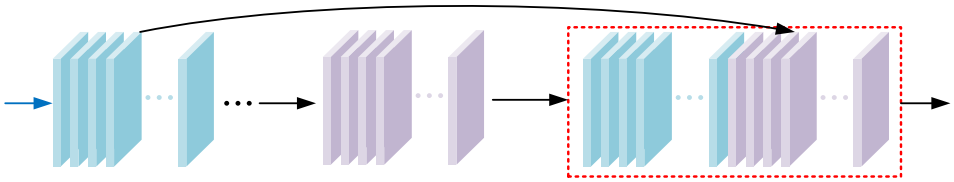


**Fig. 7.** Feature Fusion Module (FF-Module).

$$F_{FF-Module} = F_{shallow} \copyright F_{deep} \tag{8}$$

where $\boldsymbol{F}_{shallow}$ denotes shallow features, $\boldsymbol{F}_{deep}$ denotes deep features, and $\boldsymbol{F}_{FF\text{-}Module}$ denotes shallow and deep features fusion.

From Fig. 7, features from left to right will become deeper and more abstract. The fusion features $\boldsymbol{F}_{\text{FF-Module}}$ can fully combine contextual information (Lee and Kwon, 2017), which can improve detection accuracy.

The ablation study of FF-Module is discussed experimentally in Section 6.1.4.

#### 3.3.5. Module 5: FP-Module

In HyperLi-Net, three detection scales are designed, i.e. big-scale ($L$/32), medium-scale ($L$/16), and small-scale ($L$/8), which jointly constitute a FP-Module. Big-scale is designed for detecting big-size ships, medium-scale is designed for detecting medium-size ships, and small-scale is designed for detecting small-size ships. To be clear, three detection scales are sufficient because satisfactory detection results have obtained in our experiments.

Fig. 8a shows structure of FP-Module by up-sampling scale fusion which can be expressed as:

$$\begin{aligned} &D_{big} = f^{1\times1\times18}(F_{L/32}),\ \ D_{medium} = f^{1\times1\times18}(F_{L/16} + UpSa(F_{L/32})),\\ &D_{small}\ = f^{1\times1\times18}(F_{L/8} + UpSa(F_{L/16})) \end{aligned} \tag{9}$$

where $\boldsymbol{F}_{L/32}$, $\boldsymbol{F}_{L/16}$, and $\boldsymbol{F}_{L/8}$ respectively represents the feature maps of big-scale ($L$/32) layer, medium-scale ($L$/16) layer, and small-scale ($L$/8) layer, $\boldsymbol{D}_{big}$ is the output of big-scale detection, $\boldsymbol{D}_{medium}$ is the output of medium-scale detection, $\boldsymbol{D}_{small}$ is the output of small-scale detection, and $UpSa()$ is 2 × upsampling. (Fig. 8b is the down-sampling scale fusion.)

Moreover, $f^{1\times1\times18}$ is the final Separa-Conv layer with $1 \times 1 \times 18$ dimension and 18 comes from $B\times(5 + C)$ where 5 is the number of predictive parameters ($x$, $y$, $w$, $h$, $score$), $B$ is the number of bounding boxes ($B = 3$ introduced in Section 3.4.1) and $C$ is the number of classifications ($C = 1$, i.e. ship). Moreover, from Fig. 8, our proposed FP-Module not only realizes multi-scale pyramid ship detection, but also integrates the outputs of different scales by 2 × up-sampling to further inprove the detection accuracy. The effectiveness of this practice has been confirmed by previous studies (Lin et al., 2017). Certainly, this scale fusion can also be achieved by 2 × down-sampling.

In HyperLi-Net, we adopt up-sampling scale fusion. For one thing, the features from $L$/32 scale are more representative and contextual, and if they are transmitted to the bottom of pyramid, features in $L$/16 scale and $L$/8 scale will become more robust. For another thing, according to our experimental analysis, up-sampling scale fusion possesses better detection accuracy than down-sampling.

The ablation study of FP-Module is discussed experimentally in Section 6.1.5.

### 3.4. Five internal mechanisms

#### 3.4.1. Mechanism 1: RF-Model

We design HyperLi-Net according to the basic idea of one-stage detectors namely based on RF-Model. Next, we will take Faster R-CNN (Li et al., 2019) and YOLOv1 (Redmon et al., 2015) as examples to introduce R-Model and RF-Model respectively. Fig. 9a shows the architecture of Faster R-CNN. Faster R-CNN uses VGG-16 (Simonyan and Zisserman, 2014) to extract ship's features and the output of VGG-16 is respectively inputted into the first stage and the second stage. In the first stage, RPN is employed to predict bounding box proposals based on the sliding feature maps by adjusting the size of $k$ anchor boxes. Then, a softmax classifier is used to judge the type of ROIs and a bounding box regressor is used to refine the position of ROIs. In the second stage, ROIs are used to crop features that are subsequently fed to Fast R-CNN (Girshick and Fast, 2015) for classification and bounding box regression. Fig. 9b shows the architecture of YOLOv1. YOLOv1 divides the entire image into a $S \times S$ grid and each grid cell predicts $B$ bounding boxes with confidence score, and $C$ class probabilities. After processed by GoogLeNet (Szegedy et al., 2014) and FC layers, these predictions are encoded as a $S \times S \times (B \times 5 + C)$ tensor.

Fig. 10 shows the ship detection workflow of HyperLi-Net. A detailed description is provided as follows:

**Step 1:** Input SAR images to be detected. See Fig. 10a.

**Step 2:** Resize images into $L \times L$. See Fig. 10b.

In order to ensure the same feature dimension of images, we resize original images into $L \times L$ by resampling. Here, $L$ is a variable that can be adjusted to make a trade-off between accuracy and speed. In HyperLi-Net, according to our experimental analysis, we set $L = 160$, because the detection accuracy reaches the highest according to our ablation study in Table 4 of Section 4.4.1.

**Step 3:** Divide images into $S \times S$ grids. See Fig. 10c.

The images are divided into $S \times S$ grids where $S = L/32, L/16, L/8$. For example, if $L = 160$, then $S = 5,10,20$. HyperLi-Net detects ships for three times, respectively under $5 \times 5$ grids, $10 \times 10$ grids, and $20 \times 20$ grids. Inspired by the experience of FPN (Lin et al., 2017), big-scale ($L$/32) is designed for detecting big-size ships, medium-scale ($L$/

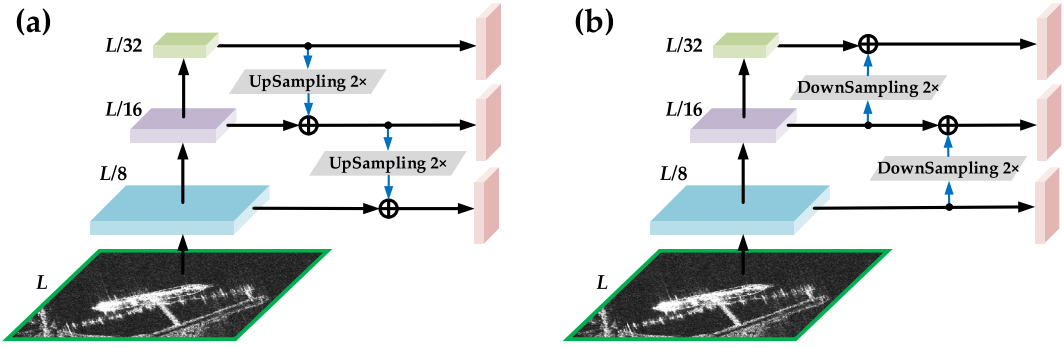



**Fig. 8.** Feature Pyramid Module (FP-Module). **(a)** Scale Fusion by Up-sampling; **(b)** Scale Fusion by Down-sampling.

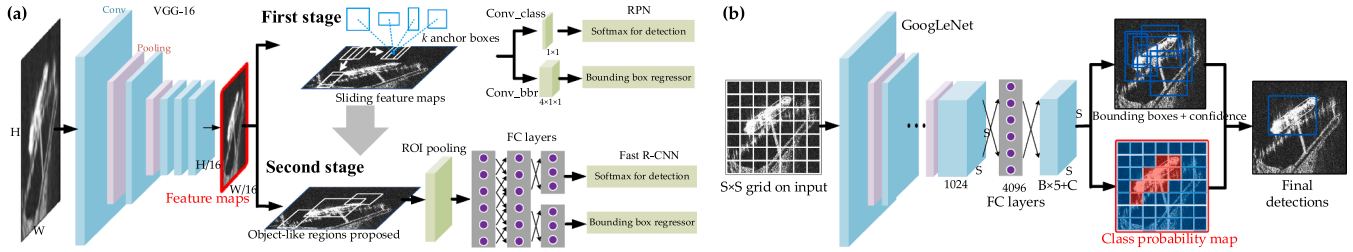



**Fig. 9.** Region Model (R-Model) and Region-Free Model (RF-Model). **(a)** Faster R-CNN (R-Model); **(b)** YOLOv1 (RF-Model).

16) is designed for detecting medium-size ships, and small-scale ($L/8$) is designed for detecting small-size ships.

**Step 4:** Locate ships' centre. See Fig. 10d.

The model can automatically locate ships' centre based on the experience learned during training, namely cell P for ship 1 and cell Q for ship 2. If the centre of a ship falls into a grid cell, this grid cell is responsible for detecting this ship. For example, in Fig. 10d, cell P is responsible for detecting ship 1, and cell Q is responsible for detecting ship 2.

In addition, in order to facilitate the introduction of loss function in Section 4.2, here, we also defined the probability that cell $i$ contains ships as follows (Redmon et al., 2015):

$$P_{cell_i}^{ship} = \begin{cases} 1, \ if \ \ cell \ i \ contains \ ships \\ 0, \ if \ \ cell \ i \ does \ not \ contains \ ships \end{cases} \tag{10}$$

For example, in Fig. 10d, cell A, B, C, D, E, F, P and Q all contain ships or a part of ships, so $P_{cell_A}^{ship}=1$, $P_{cell_B}^{ship}=1$, $P_{cell_C}^{ship}=1$, $P_{cell_D}^{ship}=1$, $P_{cell_E}^{ship}=1$, $P_{cell_F}^{ship}=1$, $P_{cell_P}^{ship}=1$, and $P_{cell_Q}^{ship}=1$. For other cells, the probability is zero.

**Step 5:** Generate $B$ bounding boxes. See Fig. 10e.

$B$ bounding boxes are generated based on cell P and cell Q obtained from Step 4. Here, $B$ is a variable that can be adjusted to make a trade-off between accuracy and speed. Furthermore, these $B$ bounding boxes may have different sizes due to the use of different features. In HyperLi-Net, according to our experimental analysis, we set $B = 3$ given that satisfactory detection results have obtained. Certainly, it is also feasible to set more bounding boxes, which may improve accuracy but will also deservedly bring about heavier computing burden.

Five predictive parameters ($x$, $y$, $w$, $h$, $score$) are used to describe the location of these bounding boxes, where ($x$, $y$) is the coordinate of the top left vertex, $w$ is the width, $h$ is the height, and score is the confidence value defined by (Redmon et al., 2015):

$$score = P_{cell_i}^{ship} \cdot IOU \tag{11}$$

where IOU is Intersection Over Union defined by:

$$IOU = (B_P \cap B_G)/(B_P \cup B_G) \tag{12}$$

where $B_P$ is the predictive box and $B_G$ is the ground truth box.

**Step 6:** Non-Maximum Suppression (NMS) (Hosang et al., 2017). See Fig. 10f.

We get $B$ bounding boxes for ship 1 and $B$ bounding boxes for ship 2. Here, we rank these $B$ bounding boxes from large to small according to their scores. Then, the bounding box with the maximum score is retained while the other $B$-1 ones with their IOU $< 0.5$ are suppressed.

**Step 7:** Coordinate mapping (Detection results). See Fig. 10g.

The detection bounding boxes with five predictive parameters ($x$, $y$, $w$, $h$, $score$) belong to $L \times L$ image in Fig. 10f. Therefore, we need to map the coordinates of the predictive box to the original images in Fig. 10g.

The ablation study of RF-Model is discussed experimentally in Section 6.2.1.

### 3.4.2. Mechanism 2: S-Kernel

S-Kernel refers that kernels' size used in HyperLi-Net is small. Suppose the input dimension of a convolution layer is $L \times L \times N_{input}$, where $L$ is the width and height and $N_{input}$ is the channel number of input. For a $k \times k \times N_{kernel}$ kernel, where $k$ is the width and height, and $N_{kernel}$ is the number of kernels, the parameter quantity for a convolution operation is:

$$Parameter_{conv} = (k^2 \cdot N_{input} + b) \cdot N_{kernel} \tag{13}$$

where $b$ is the bias parameter quantity (generally $b = 1$ for one kernel). From Eq. (13), the parameter quantity decreases with kernel size decreases ($k\downarrow$). For example, as shown in Fig. 11a, there are three kernels with different sizes $1 \times 1$, $3 \times 3$ and $5 \times 5$. From Fig. 11a, the larger the kernel is, the larger the receptive field is, but parameter quantity increases.

In AlexNet (Krizhevsky et al., 2017), some big kernels are used, e.g, $5 \times 5$ and $11 \times 11$, which can absorb more information but lead to huge model (238 MB) (Iandola et al., 2016). Obviously, such a big model is difficult to apply in practice. In fact, it has been abandonded by almost detectors (Szegedy et al., 2015). In addition, since $1 \times 1$ kernels merely can observe only one pixel without adequate receptive field, they are usually excluded. In particular, in order to ensure that the two sides of the image are still symmetrical when zeropadding, $k$ must be odd numbers. Therefore, we choose $3 \times 3$ kernels in the backbone of HyperLi-Net.

The ablation study of S-Kernel is discussed experimentally in Section 6.2.2.

### 3.4.3. Mechanism 3: N-Channel

N-Channel refers that the kernel's channel used in HyperLi-Net is narrow. From Eq. (13), equally, the parameter quantity also decreases with the number of kernel decreases ($N_{kernel}\downarrow$) (from 10 parameters for $N_{kernel} = 1$ to 30 parameters for $N_{kernel} = 3$, if $N_{input} = 1$, $k = 3$, as is shown in Fig. 11b). In HyperLi-Net, we set $N_{kernel} = 32$, universally fewer than other object detectors bringing fewer parameters. Especially, distinguishing from other object detectors, the vaule of $N_{kernel}$ in HyperLi-Net remains unchanged with the deepening of layers. The purpose of this practice is to facilitate adjustment in our experiments so

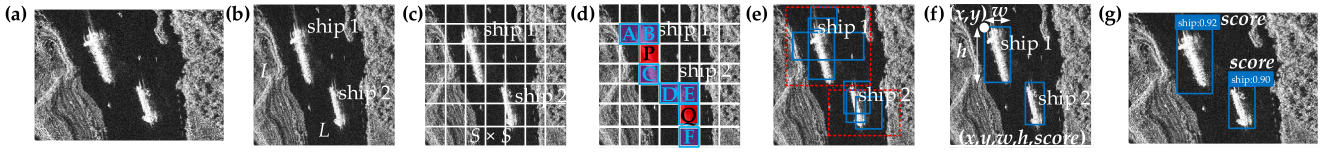



**Fig. 10.** SAR Ship Detection Workflow of HyperLi-Net. **(a)** Step 1: Original Images; **(b)** Step 2: Resize Images; **(c)** Step 3: Divide Images; **(d)** Step 4: Locate Ships' Centers; **(e)** Step 5: Generate Bounding Boxes; **(f)** Step 6: Non-Maximum Suppression; **(g)** Step 7: Detection Results.

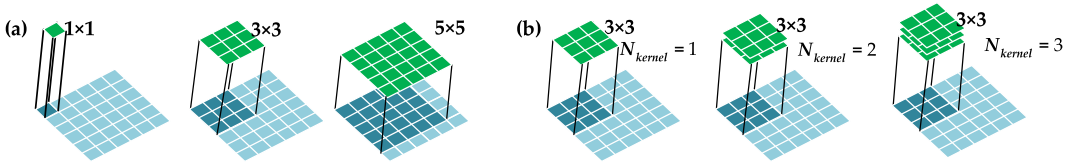



**Fig. 11.** Small Kernel (S-Kernel) and Narrow Channel (N-Channel) **(a)** S-Kernel; **(b)** N-Channel.

as to determine the appropriate value of $N_{kernel}$. Furthermore, the reason why we do not set $N_{kernel} < 32$ is that the detection accuracy gets greatly declined if we do so.

The ablation study of N-Channel is discussed experimentally in Section 6.2.3.

### 3.4.4. Mechanism 4: Separa-Conv

Separa-Conv was proposed by L. Sifre in 2014 (Sifre, 2014) consisting of a Depthwise Convolution (D-Conv) and a Pointwise Convolution (P-Conv). Compared to Trad-Conv (Lecun et al., 1998), it has fewer parameters because it decouples channel and spatial correlations (Zhang et al., 2019; Chollet, 2016). So far, its validity have been proved by many previous studies (Zhang and Zhang, 2019; Zhang et al., 2019; Chollet, 2016). Especially, it has a tremendous application prospect in mobile communication and embedded devices for its lighter model and faster speed. However, current most detectors all utilize Trad-Conv to construct models, leading to heavy computational cost and slow speed. On the contrary, our proposed HyperLi-Net specially employs Separa-Conv to reduce parameter quantity.

Fig. 12 is the diagrammatic sketch of Trad-Conv and Separa-Conv. From Fig. 12a, in Trad-Conv, each kernel ($K_1$,$K_2$,$K_3$) must convolute all three channel inputs to obtain three channel outputs ($O_1$,$O_2$,$O_3$). For example, for the output $O_1$, the process of Trad-Conv can be expressed as follows:

$$O_1 = I_1 \otimes K_1 + I_2 \otimes K_1 + I_3 \otimes K_1 \tag{14}$$

where $I_1$, $I_2$ and $I_3$ are the inputs and $\otimes$ denotes the convolution operation.

From Fig. 12b, Separa-Conv first perfoms D-Conv operation, and then performs P-Conv operation. For D-Conv, each kernel ($K_1$,$K_2$,$K_3$) convolutes only one channel to obtain temporary outputs ($J_1$,$J_2$ ,$J_3$). Then, for P-Conv, each $1 \times 1$ kernel ($K'_1$,$K'_2$,$K'_3$) convolute all three temporary outputs ($J_1$,$J_2$ ,$J_3$) to get the eventual outputs ($O_1$,$O_2$,$O_3$). For example, for the output $O_1$, the process of Separa-Conv can be expressed as follows:

$$O_1 = I_1 \otimes K_1 \otimes K'_1 + I_2 \otimes K_2 \otimes K'_1 + I_3 \otimes K_3 \otimes K'_1 \tag{15}$$

In fact, although Separa-Conv adds P-Conv compared with Trad-Conv, its total parameter quantity does not increase because kernel sizes of $K'_1$,$K'_2$,$K'_3$ in P-Conv are all $1 \times 1$ bringing slight parameter quantity increase. In brief, the parameter quantity of D-Conv gets largely reduced compared with Trad-Conv, meanwhile that of P-Conv gets slightly increased. Thus, the total parameter quantity of Separa-Conv is still largely less than that of Trad-Conv. Assume that image size is $L \times L \times N_{input}$, where $L$ is width and height, and $N_{input}$ is the channel number of input. Moreover, assume that kernel size is $k \times k \times N_{kernel}$, where $k$ is width and height, and $N_{kernel}$ is the number of kernels.

From Fig. 13, the computational costs of Trad-Conv and Separa-Conv are:

$$Computation_{Trad-Conv} = L^2 \cdot k^2 \cdot N_{kernel} \cdot N_{input} \tag{16}$$

$$Computation_{Separa-Conv} = L^2 \cdot k^2 \cdot N_{input} + L^2 \cdot N_{kernel} \cdot N_{input} \tag{17}$$

Therefore, their ratio Eq. (17)/Eq. (16) is:

$$ratio = \frac{1}{N_{kernel}} + \frac{1}{k^2} \tag{18}$$

where $N_{kernel} > > 1$ and $k > 1$ ($N_{kernel} = 32$, $k = 3$ in HyperLi-Net), i.e. $ratio \ll 1$. Therefore, Separa-Conv has less computational cost than Trad-Conv, leading to faster detection speed.

The ablation study of Separa-Conv is discussed experimentally in Section 6.2.4.

### 3.4.5. Mechanism 5: BN-Fusion

BN can accelerate deep network training by reducing internal covariate shift (Sifre, 2014) whose algorithm implementation process can be found in (Sifre, 2014), but it reduces detection speed for extra computation. Thus, after obtaining the initial model, we fuse BN into Separa-Conv (D-Conv and P-Conv) to reduce computational cost, as shown in Fig. 14. In Fig. 14, ReLU is an activation function, defined by:

$$ReLU(x) = \max\{0, x\} \tag{19}$$

For D-Conv layer and P-Conv layer, they can be expressed as:

$$Y = W \cdot X + b \tag{20}$$

where $\boldsymbol{X}$ is the input vector, $\boldsymbol{W}$ is the weight vector, $\boldsymbol{b}$ is the bias vector, and $\boldsymbol{Y}$ is the output vector.

For a BN-layer, it can be expressed as:

$$Y' = \xi \cdot \frac{X' - m(X')}{\sqrt{\sigma^2(X') + \varepsilon}} + \eta \tag{21}$$

where $\boldsymbol{X'}$ denotes input vector, $\boldsymbol{Y'}$ denotes output vector, $\xi$ and $\eta$ are the parameters obtained during training, $m()$ denotes averaging, $\sigma^2()$ denotes variance and $\varepsilon$ is a small constant added to the mini-batch variance for numerical stability (Sifre, 2014).

Therefore, for BN-Fusion, put Eq. (20) into Eq. (21), then obtain:

$$Y'' = W'' \cdot X + b'', \text{ where } \quad W'' = \xi \cdot \frac{W}{\sqrt{\sigma^2(X) + \varepsilon}},$$
$$b'' = \xi \cdot \frac{b - m(X)}{\sqrt{\sigma^2(X) + \varepsilon}} + \eta \tag{22}$$

where $\boldsymbol{W}$" denotes new weight vector and $\boldsymbol{b}$" denotes new bias

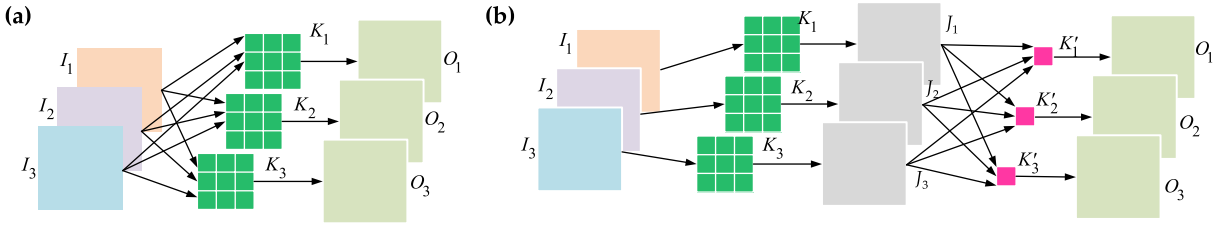



**Fig. 12.** Diagrammatic Sketch of Trad-Conv and Separa-Conv. **(a)** Trad-Conv; **(b)** Separa-Conv.

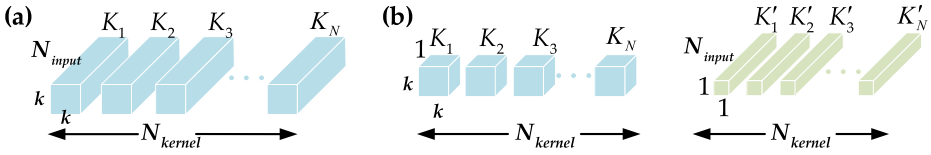



**Fig. 13.** Computational Complexity Analysis between Trad-Conv and Separa-Conv. **(a)** Trad-Conv; **(b)** Separa-Conv.

vector. In Eq. (22), ***W***"and ***b***" are calculated beforehand in detection model, so detection model only contains an equivalent convolution ***Y***" = ***W***" •***X*** + ***b***", avoiding BN operations from Eq. (21), bringing the decrease of computation costs, as a result, the detection speed can obtained further improved.

The ablation study of BN-Fusion is discussed experimentally in Section 6.2.5.

## 4. Experiments

Our experiments are performed on a personal computer with Intel (R) i9-9900 K CPU, NVIDIA RTX2080Ti GPU, and 32G memory. We use CUDA 10.0 and cuDNN 7.6 to call GPU for training acceleration. Our codes are written based on Keras framework (Keras, 2019), a Python-based deep learning library with TensorFlow (Tensorflow, 2019) as backend.

In our experiments, we set IoU = 50% as the detection threshold, which means that if the overlap area of a predictive box and a ground truth box exceeds or equals 50%, then ships in this predictive box are detected successfully. This detection threshold can be determined according to the trade-off between missed-detection and false-alarm. In general, IoU = 50% is a quite reasonable detection threshold in the deep learning community (See Pascal VOC detection evaluation metrics in Reference (Everingham et al., 2014).).

### 4.1. Dataset

Li. et al. (Li et al., 2017; Li et al., 2019; Li et al., 2019; Li et al., 2019) proposed the first open SAR Ship Detection Dataset (SSDD) that has already been used by many scholars (Zhang and Zhang, 2019; Zhang et al., 2019; Yang et al., 2019; Zhang, 2020; Cui et al., Nov. 2019; Chang et al., 2019; An et al., 2019; Liu et al., 2019; Wang et al., 2018; Zhang et al., 2019; Zhang et al., 2019), so we choose SSDD to conduct experiments for more reasonable comparison.

In addition, the number of SAR images in SSDD is far less than some mainstream datasets in the CV community, e.g. Pascal VOC dataset (Everingham et al., 2014), COCO dataset (COCO-Common Objects in Context, 2019), ImageNet dataset (Russakovsky et al., 2014), etc. Limited labeled data may lead to over fitting of training, thus leading to learning bias, so we collect more SAR images from Gaofen-3 and Sentinel-1 to construct a Gaofen-SSDD dataset and a Sentinel-SSDD dataset to further verify effectiveness of HyperLi-Net.

Table 3 shows the detailed descriptions of three datasets. From Table 3, there are 1160 SAR images in SSDD containing 2358 ships, with 2.03 ships per image on average, obtained from different sensors, places, polarization modes, resolutions and ship scenes. In addition, ships in SSDD also have many different sizes, from the smallest size 7 × 7 to the biggest size 211 × 298, which can also evaluate the multi-scale detection performance of detectors. SSDD is a reasonable dataset because of the diversity of its samples, which can evaluate the robustness of ship detectors.

From Table 3, there are 20,000 SAR images with 160 × 160 size in Gaofen-SSDD from Gaofen-3, obtained from References (Wang et al., 2019; Wang et al., 2019). There are 29,250 ships in all these images with 1.46 ships in one image on average. Ships in Gaofen-SSDD also have many different sizes, from the smallest size 4 × 5 to the biggest size 81 × 130. SAR images in Gaofen-SSDD also has multiple scenes, multiple resolutions and multiple polarization modes, so Gaofen-SSDD is also a reasonable dataset. The ground truths in Gaofen-SSDD are annotated by SAR experts using an image annotation tool LabelImg (LabelImg, 2019).

From Table 3, there are 20,000 SAR images with 160 × 160 size in Sentinel-SSDD from Sentinel-1 obtained from References (Wang et al., 2018; Wang et al., 2019), Sentinels Scientific Data Hub (Copernicus Open Access Hub, 2019) and OpenSAR (OpenSAR, 2019). There are 22,925 ships in all these images with 1.17 ships per image on average. Ships in Sentinel-SSDD have many different sizes, from the smallest size 1 × 17 to the biggest size 57 × 49. The ground truths in Sentinel-SSDD are annotated by SAR experts with the help of Association Information Systems (AIS) (Copernicus Open Access Hub, 2019; OpenSAR, 2019; Huang et al., 2018).

### 4.2. Training configuration

The total loss function of training contains the errors of five predictive parameters (*x, y, w, h, score*), i.e.,

$$loss_{(x,y)} = \lambda_1 \sum_{i=0}^{B} \sum_{j=0}^{S^2} P_{cell_i}^{ship} [(x_i - \widetilde{x}_{i,j})^2 + (y_i - \widetilde{y}_{i,j})^2] \tag{23}$$

$$loss_{(w,h)} = \lambda_2 \sum_{i=0}^{B} \sum_{j=0}^{S^2} P_{cell_i}^{ship} [(\sqrt{w_i} - \sqrt{\widetilde{w}_{i,j}})^2 + (\sqrt{h_i} - \sqrt{\widetilde{h}_{i,j}})^2] \tag{24}$$

$$\begin{aligned} loss_{(score)} &= \lambda_3 \sum_{i=0}^{B} \sum_{j=0}^{S^2} P_{cell_i}^{ship} [\widetilde{s}_{i,j} - IOU(B_P, B_G)]^2 + \lambda_p \\ &\sum_{i=0}^{B} \sum_{j=0}^{S^2} \left(1 - P_{cell_i}^{ship}\right) [\widetilde{s}_{i,j} - IOU(B_P, B_G)]^2 \end{aligned} \tag{25}$$

where $(x_i, y_i)$ is the coordinate of the $i$-th ground truth box, $(w_i, h_i)$ is the width and height of the $i$-th ground truth box, $(x_{i,j}, y_{i,j})$ is the coordinate of the $i$-th predictive box of the $j$-th grid cell, $(w_{i,j}, h_{i,j})$ is the width and height of the $i$-th predictive box of the $j$-th grid cell and $s_{i,j}$ is the score of the $i$-th predictive box of the $j$-th grid cell. $\lambda_1 = 5$ is the weight coefficient of coordinate loss, $\lambda_2 = 5$ is the weight coefficient of width and height loss, $\lambda_3 = 1$ is the weight coefficient of score loss and

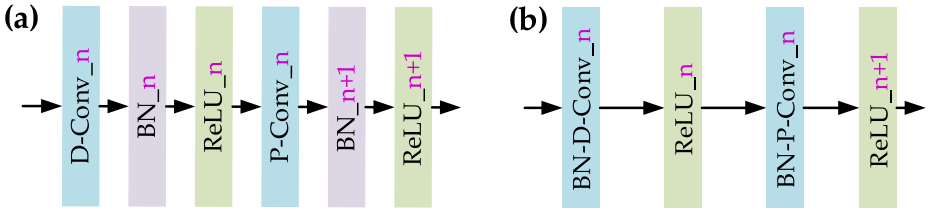



**Fig. 14.** BN-Fusion. **(a)** Training Model; **(b)** Detection Model.

$\lambda_p = 0.5$ is a penalty coefficient when no ships are detected inspired by the experience of YOLOv3 (Redmon and Farhadi, 2018).

We randomly divide the datasets into a training set, a validation set and a test set by 7:2:1 ratio (Zhang and Zhang, 2019; Zhang et al., 2019; Cui et al., Nov. 2019; Chang et al., 2019; Li et al., 2017; Li et al., 2019; Li et al., 2019; Li et al., 2019). We use Adam (Kingma and Ba, 2014) to update parameters and train 2000 epochs with batch size = 32 considering GPU memory limitation. Poly policy (Chen et al., 2017) is used to adjust learning rate with initial learning rate as 0.001 and power as 0.9 (See Fig. 15a).

Fig. 15b shows the training and validation loss curves. From Fig. 15b, for one thing, HyperLi-Net can converge rapidly by using the loss function introduced before. For another thing, the gap between training loss and validation loss is narrow, which shows that there is no over-fitting phenomenon in our network (The verification loss occasionally appears a sudden upward jump, but it can keep basically stable, eventually.).

Fig. 15c shows the mAP (accuracy) on the validation set with the increase of epoch. From Fig. 15c, mAP increases dramatically when epoch < 200 (from 0% to 80%), showing the cost-effective training of HyperLi-Net. Then it increases slowly with slight fluctuation when 200 < epoch < 1000. Finally it keeps basically stable when epoch > 1000, which is in line with common sense because it is impossible for any networks that there is no upper limit of accuracy.

### 4.3. Evaluation index

We use two types of accuracy evaluation indexes respectively from remote sensing community ($P_d$, $P_m$ and $P_f$) and and from deep learning community (recall, precision and mAP).

Detection probability $P_d$, missed-detection probability $P_m$ and false-alarm probability $P_f$ is defined by (Wang et al., 2018; Zhang, 2020; Zhang et al., 2019):

$$P_d = TP/GT \tag{26}$$

$$P_m = FN/GT \tag{27}$$

$$P_f = FP/(TP + FP) \tag{28}$$

where TP is True Positive, GT is Ground Truth, FN is False Negative and FP is False Positive.

Recall, precision and mean Average Precision (mAP) are defined by (Osco et al., 2020; Schilling et al., 2018):

$$Recall = TP/(TP + FN) \tag{29}$$

$$Precision = TP/(TP + FP) \tag{30}$$

$$mAP = \int_0^1 P(R)\cdot dR \tag{31}$$

where P is precision, R is recall and P(R) is the precision-recall (P-R) curve.

The ship detection time of one SAR image denotes *Time* (s) and Frames Per Second (FPS) is defined by:

$$FPS = 1/Time \tag{32}$$

In our work, mAP and FPS are respectively used to represent accuracy and speed presented in Tables 4- 10 and Tables 13-20 of Section 4.4.1-Section 4.4.5, Section 5.1-Section 5.3 and Section 6.1-Section 6.2, because they are the most common-used in the deep learning community. In addition, other indicators are merely responsible for providing an effective observation and a future research baseline.

### 4.4. Model establishment details

#### 4.4.1. Ablation study on L

The input image size of HyperLi-Net is $L$ and the final output size is $L/32$, so the whole network can be regarded as 32 times down-sampling (Input: $L \rightarrow$ Output: $L/32$), i.e., $L$ must be a multiple of 32, then we set $L = 32m$ ($m = 1,2,\ldots,10$) to conduct ten comparative experiments.

Table 4 shows the ablation study on $L$ and the following conclusions can be drawn:

(1) The detection accuracy obtains dramatically improved with the increasement of $L$ when $L < 160$, originating from the increase of images information. Then, the detection accuracy reaches the peak value when $L = 160$. Finally, it does not grow any more, but fluctuates dynamically when $L > 160$.
(2) The detection speed gets decreased with the increasement of image size $L$. Obviously, bigger images are bound to result in much computational cost, bringing about slower speed (from 243 FPS of $L = 32$ to 188 FPS of $L = 320$).

Finally, we set $L = 160$ as our final model, because-1) for one thing, the detection accuracy reaches the best, compared with other values, meanwhile the detection speed does not reduce too much, compared with $L = 32$ (from 243 FPS to 222 FPS); 2) for another thing, the total training time of $L = 160$ is 3.33 h far less than that of $L = 288$ or 320 (8.89 h, 11.67 h), which can improve training efficiency.

#### 4.4.2. Ablation study on Z

From Table 2, we set $Z$ layers in backbones, so we conduct five comparative experiments when Z = 1, 2, 3, 4 and 5 to perform ablation study.

Table 5 shows the ablation study on $Z$, and the following conclusions can be drawn:

(1) The detection accuracy of $Z = 2$ is higher than others'. If $Z = 1$, there will not be no redundant layers for feature extraction because the first layer in backbones is only responsible for changing the size

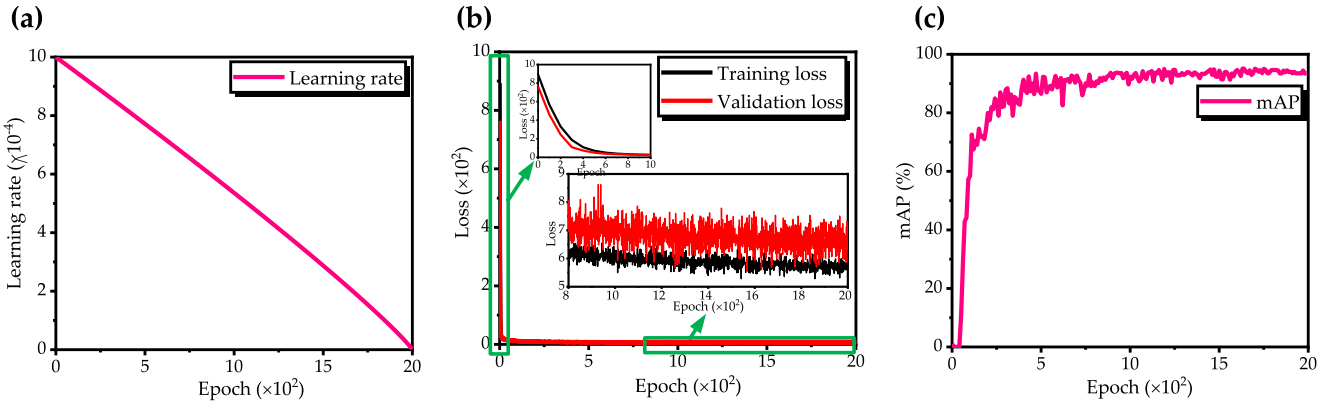



**Fig. 15.** Training Information. **(a)** Poly Policy; **(b)** Training Loss and Validation Loss; **(c)** mAP (accuracy) on the validation set.

**Table 7**
Ablation Study on Pre-Training. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Pre-traning? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | **mAP** | Time | **FPS** | Parameters | FLOPs | **Size** |
| ✓ | 94.02% | 5.98% | 9.90% | 94.02% | 90.10% | 93.34% | 4.50 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| ✗ | **96.74%** | **3.26%** | **9.64%** | **96.74%** | **90.36%** | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

of input feature maps and the remaining layers are responsible for extracting ships' features. For the simplexity of ships' features, one extra layer has already been rather adequate to extract features. Unexpectedly, the detection accuracy declines when $Z > 2$. One possible reason is that the network happens to fall into local minima, hindering further parameter optimization.
(2) The detection speed gets decreased with the increase of $Z$, which is in line with common sense for more parameters, higher FLOPs and bigger model size.

Finally, we set $Z = 2$ in our final model, because-1) for one thing, the detection accuracy reaches the best compared with other values, meanwhile the detection speed does not reduce too much compared with $Z = 1$ (from 234 FPS to 222 FPS); 2) for another thing, compared with $Z = 5$, its model is more light-weight (0.66 MB < 0.91 MB), on the premise of achieving similar detection accuracy (96.08% mAP vs 95.74% mAP). Moreover, the mode size of HyperLi-Net is 0.69 MB.

#### 4.4.3. Ablation study on residual

From Table 2, we adopt residual learning in backbones. Here, we will perform ablation study on residual learning.

Table 6 shows the ablation study on residual, and the following conclusions can be drawn:

(1) The detection accuracy of residual learning is slightly superior to non-residual learning, showing that residual learning is beneficial to improve accuracy, because skip connections (He et al., 2015) can make network training more sufficient.
(2) The detection speed of residual learning is similar to non-residual learning for their same parameter quantity, FLOPs and model size.

Finally, we choose residual learning in HyperLi-Net. For one thing, the detection accuracy can be improved slightly (96.08% mAP > 95.13% mAP). For another thing, the detection speed scarcely declines (222 FPS vs 223 FPS).

#### 4.4.4. Ablation study on Pre-Training

Nowadays, most detectors have to fine-tune networks pre-trained on ImageNet (He et al., 2018) (transfer learning (Wang et al., 2018; Ribani and Marengoni, 2019; Mateo-García et al., 2020; Huang et al., 2020; Huang et al., 2020; Huang et al., 2017, 2020; Huang et al., 2019; Esteva et al., 2017), which incurs learning bias due to the huge domain mismatch between SAR images and ImageNet images (Deng et al., 2019). However, our proposed HyperLi-Net can be trained from scratch (Deng et al., 2019), which can improves training efficiency. We conduct two comparative experiments respectively under pre-learning or not to verify such advantage.

Fig. 16 shows the loss curves of pre-training or not. From Fig. 16, the following conclusions can be drawn:

(1) In the first 1000 epochs, both training loss and validation loss of pre-training are lower than that of non-pre-training and decrease faster than that of non-pre-training. Therefore, this phenomenon shows that the initial parameter values obtained from pre-training are better than that of random initialization of non-pre-training.
(2) When 1000 < epoch < 1800, the loss change rates of pre-training and non-pre-training nearly equal, but the loss of pre-training is still lower than that of non-pre-training.
(3) Finally, when epoch > 1800, the gap of their loss curves becomes rather narrow, then almost similar training results are obtained. One possible reason is that all parameters in HyperLi-Net are fully trained, as long as the network undergoes adequate training iterations, regardless of pre-training or non-pre-training.

Therefore, HyperLi-Net can be trained from scratch, successfully and efficiently. A reasonable explanation for this advantage is that compared with large-scale networks, HyperLi-Net has such fewer parameters that these parameters with random initial values can achieve the similar effect to pre-training, as long as there are adequate training iterations.

Table 7 shows the ablation study on pre-learning, and the following conclusions can be drawn:

(1) The detection accuracy of non-pre-training is, unexpectedly, superior to that of pre-training. A possible cause for this phenomenon is that when huge learning bias appears for the huge domain mismatch between SAR images and ImageNet images (Deng et al., 2019). In fact, pre-training is essentially more suitable for the situation with similar data sources. Obviously, ships in SAR images are greatly different from that in optical ImageNet images.
(2) The detection speed of non-pre-training equals to that of pre-training, due to their same parameter quantity, FLOPs, and model size.

Finally, we do not pre-train our model on ImageNet, because-1) for one thing, the detection accuracy of non-pre-training is superior to pre-training; 2) for another thing, training efficiency can obtained prominently improved because the huge ImageNet dataset is bound to result in much time consumption.

#### 4.4.5. Ablation study on CPU

Nowdays, many mainstream object detectors tend to excessively rely on high-performance GPUs to speed up training coming from their heavy-weight models. Otherwise, 1) the huge CPU training time brings

**Table 8**
Ablation Study on CPU. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Hardware | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Training Time Per Epoch |
|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | |
| Intel i9-9900 K CPU | **96.74%** | **3.26%** | **8.72%** | **96.74%** | **91.28%** | 95.96% | 14.18 ms | 71 | 66 s (total 36.67 h) |
| NVIDIA RTX2080Ti GPU | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 6 s (total 3.33 h) |

**Table 9**
Evaluation Indexes of HyperLi-Net. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Dataset | GT | TP | FN | FP | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS |
| SSDD | 184 | 178 | 6 | 19 | 96.74% | 3.26% | 9.64% | 96.74% | 90.36% | **96.08%** | 4.51 ms | **222** |
| Gaofen-SSDD | 2979 | 2723 | 256 | 446 | 91.41% | 8.59% | 14.07% | 91.41% | 85.93% | **88.20%** | 4.05 ms | **247** |
| Sentinel-SSDD | 2258 | 2046 | 212 | 174 | 90.61% | 9.39% | 7.84% | 90.61% | 92.16% | **89.11%** | 4.03 ms | **248** |

great challenges to the timely training monitoring; 2) CPUs may be forced to stop working due to their limited capacity for massive and highly-triggered parallel computing in the worst cases. Fortunately, the model size of HyperLi-Net is only 0.69 MB, so it can be efficiently trained on CPUs, slashing expenses caused by expensive and high-power-consumption GPUs. More importantly, this advantage can also smoothly reduce obstacles of deep learning in practical application.

Table 8 shows the ablation study on CPU, and the following conclusions can be drawn:

(1) The detection accuracy of CPU undergoing an extra training is nearly close to that of GPU. Certainly, their slight accuracy gap may come from ineluctable random error (perhaps or not). This phenomenon fully shows that the detection performance of HyperLi-Net is rather stable, not affected by different hardwares. Moreover, for the same model undergoing a same training, their detection accuracy remains unchanged.
(2) The detection speed of CPU is largely lower than that of GPU, which is in line with common sense because the computing capability of the former is far inferior to the latter's. Remarkably, the training time per epoch of CPU is 66 s, so the total one is 36.67 h, still relatively acceptable in the deep learning community, in constract to often several days or several weeks for other object detectors if needing to achieve 2000 iterations.

Given the above, HyperLi-Net is a simple and easy-trained network, not limited by hardware configuration, due to its fewer parameters, lower computation costs and lighter model size. Certainly, in order to rapidly obtain the experimental results, our other experiments are run on GPU

### 4.5. Visualization of feature maps

After obtaining the detection model, in order to more vividly and intuitively observe the processing process of SAR image by HyperLi-Net, we visualize partial output feature maps of some representative layers. At present, deep learning is still a "black-box" model, whose underlying mathematical principle is still unknown by human beings, so our such feature visualization work is necessary even interesting (very few reports are involved with this work).

Fig. 17a shows the original SAR image to be detected, which will be inputted into HyperLi-Net. From Fig. 17a, the original SAR image's dimension is 160 × 160 × 3, where 160 is the width and height, and 3 is the channel number of images. There are 2 ships in the image with different sizes under the near-shore background.

Fig. 17b shows the partial output feature maps from Concate-1 after the process of MRF-Module. From Fig. 17b, the dimension of the feature maps is 80 × 80 × 96, where 80 comes from convolution operations with stride = 2, and 96 comes from 3 types of kernels (1 × 1, 2 × 2,

**Table 10**
Evaluation Indexes of Different Methods. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Dataset | Method | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | |
|---|---|---|---|---|---|---|---|---|---|
| | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS |
| SSDD | Faster R-CNN | 85.16% | 14.84% | 18.85% | 85.16% | 81.15% | 82.66% | 327.48 ms | 3 |
| | RetinaNet | 96.70% | 3.30% | 6.88% | 96.70% | 93.12% | 95.68% | 314.43 ms | 3 |
| | R-FCN | 95.65% | 4.35% | 7.37% | 95.65% | 92.63% | 95.15% | 178.16 ms | 6 |
| | SSD | 94.51% | 5.49% | 14.85% | 94.51% | 85.15% | 92.67% | 48.86 ms | 20 |
| | YOLOv3 | 96.70% | 3.30% | **6.38%** | 96.70% | **93.62%** | 95.34% | 22.30 ms | 45 |
| | YOLOv1 | 84.07% | 15.93% | 15.47% | 84.07% | 84.53% | 81.24% | 21.95 ms | 46 |
| | YOLOv2 | 92.86% | 7.14% | 15.08% | 92.86% | 84.92% | 90.09% | 19.01 ms | 53 |
| | YOLOv3-tiny | 70.33% | 29.12% | 22.29% | 70.33% | 77.58% | 64.64% | 10.25 ms | 98 |
| | YOLOv2-tiny | 47.80% | 52.20% | 26.27% | 47.80% | 73.73% | 44.40% | 9.43 ms | 107 |
| | **HyperLi-Net (Ours)** | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | **4.51 ms** | **222** |
| Gaofen-SSDD | Faster R-CNN | 87.95% | 12.05% | 10.61% | 87.95% | 89.39% | 85.17% | 322.15 ms | 3 |
| | RetinaNet | 89.93% | 10.07% | 13.47% | 89.93% | 86.53% | 87.16% | 318.75 ms | 3 |
| | R-FCN | 90.57% | 9.43% | 15.85% | 90.57% | 84.15% | 86.70% | 167.84 ms | 6 |
| | SSD | 88.15% | 11.85% | 12.93% | 88.15% | 87.07% | 85.41% | 45.51 ms | 22 |
| | YOLOv3 | 90.23% | 9.77% | 12.33% | 90.23% | 87.67% | 87.55% | 19.32 ms | 52 |
| | YOLOv1 | 74.72% | 25.28% | 14.61% | 74.72% | 85.39% | 70.77% | 18.23 ms | 55 |
| | YOLOv2 | 84.26% | 15.74% | **10.39%** | 84.26% | **89.61%** | 81.66% | 16.94 ms | 59 |
| | YOLOv3-tiny | 64.15% | 35.85% | 15.78% | 64.15% | 84.22% | 60.10% | 9.10 ms | 110 |
| | YOLOv2-tiny | 42.20% | 57.80% | 16.31% | 42.20% | 83.69% | 39.11% | 8.25 ms | 121 |
| | **HyperLi-Net (Ours)** | **91.41%** | **8.59%** | 14.07% | **91.41%** | 85.93% | **88.20%** | **4.05 ms** | **247** |
| Sentinel-SSDD | Faster R-CNN | 87.38% | 12.62% | 6.71% | 87.38% | 93.29% | 86.08% | 322.10 ms | 3 |
| | RetinaNet | 90.08% | 9.92% | 7.04% | 90.08% | 92.96% | 88.89% | 319.45 ms | 3 |
| | R-FCN | 88.93% | 11.07% | 7.12% | 88.93% | 92.88% | 87.49% | 166.94 ms | 6 |
| | SSD | 86.85% | 13.15% | 5.04% | 86.85% | 94.96% | 85.65% | 45.57 ms | 22 |
| | YOLOv3 | 89.59% | 10.41% | 6.82% | 89.59% | 93.18% | 88.24% | 19.49 ms | 51 |
| | YOLOv1 | 69.71% | 30.29% | 5.47% | 69.71% | 94.53% | 72.41% | 18.36 ms | 54 |
| | YOLOv2 | 85.03% | 14.97% | 5.42% | 85.03% | 94.58% | 83.83% | 17.02 ms | 59 |
| | YOLOv3-tiny | 63.37% | 36.63% | **4.66%** | 63.37% | **95.34%** | 62.78% | 9.15 ms | 110 |
| | YOLOv2-tiny | 42.03% | 57.97% | 7.23% | 42.03% | 92.77% | 41.07% | 8.35 ms | 120 |
| | **HyperLi-Net (Ours)** | **90.61%** | **9.39%** | 7.84% | **90.61%** | 92.16% | **89.11%** | **4.03 ms** | **248** |

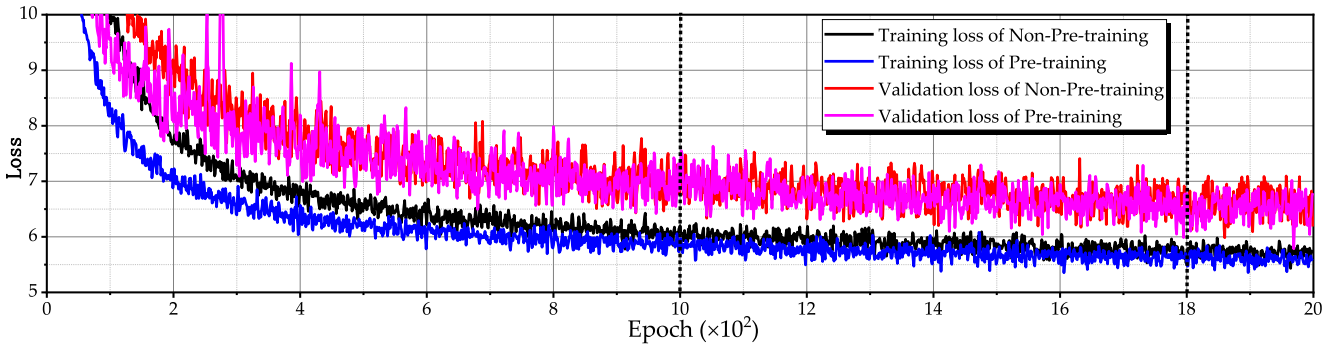



**Fig. 16.** Loss Curves of Non-Pre-Training and Pre-Training.

3 × 3) with 32 channels in each kernel. In Fig. 17b, we choose 8 representative feature maps among them to display, numbered as b1 ~ b8. Firstly, we can notice that the features of b1 ~ b8 are obviously different. For these features extracted by MRF-Module, our tentative explanation is that b1 seems to be a sharpening result, b2 seems to be a denoising result, b3 seems to be a result of skeleton extraction, b4 seems to be a corrosion results, and b5 seem to be a result of land-ship separation. However, b6 ~ b8 are too abstract to explain, which in fact is a common sense in the deep learning community (Lecun et al., 1998; LeCun et al., 2015; Zeiler and Fergus, 2013). Secondly, we can clearly notice that b1 ~ b5 may be valuable (attention channels marked in black), while b6 ~ b8 may be useless (non-attention channels marked in red).

Fig. 17c shows the partial output feature maps from Concate-2 after the process of DC-Module. From Fig. 17c, equally, the dimension of the feature maps is also 80 × 80 × 96, where 80 comes from convolution operations with stride = 2 and 96 comes from 3 types of kernels (dilated 1 × 1, 2 × 2, 3 × 3) with 32 channels in each kernel. In Fig. 17c, we also choose 8 representative feature maps among them to display, numbered as c1 ~ c8. Firstly, some conclusions obtained from Fig. 17b also hold on in Fig. 17c. Secondly, the features in DC-Module are different from that in MRF-Module, showing the exsisting necessity and value of DC-Module. Thirdly, c5 seems to be ships' outlines. Fourthly, more interestingly, the two ships are successfully separated from lands in c6. Finally, c7 ~ c8 are hard-explained. Thus, the powerful feature extraction ability of deep learning is confirmed, avoiding heavy artificial participation and complex theoretical design.

Fig. 17d shows the partial output feature maps from CSA-Module-1 after the process of CA-module. From Fig. 17d, CA-Module can successfully distinguish valuable channels and useless ones. For example, the valuable channels b1 ~ b5 and c1 ~ c6 are retained while the useless channels b6 ~ b8 and c7 ~ c8 are suppressed (marked in white zero).

Fig. 17e shows the partial output feature maps from CSA-Module-1 after the process of SA-module. From Fig. 17e, SA-Module can successfully seek out valuable places in space. For example, the two ships are mapped into two small regions in the top of feature maps, i.e., attention space (marked in pink ellipses in Fig. 17e). However, unexpectedly, the positions of ships (two small regions) are shifted. In fact, the reason for this phenomenon is that $f^{7\times7}$ Separa-Conv in Fig. 6c contains D-Conv2D operation and P-Conv2D operation (introduced in Fig. 12b).

The former (D-Conv2D) has already been able to successfully focus on these two ships. However, in order to reduce calculation costs in the network's backend, the latter (P-Conv2D) can compress original ships' space information into small regions (feature concentration similar to compressed sensing and sparse representation), and then it can also sort them on the feature map in importance order from top to bottom. Therefore, our CSA-Module is different from the original attention modules from Reference (Woo et al., 2018). For example, the

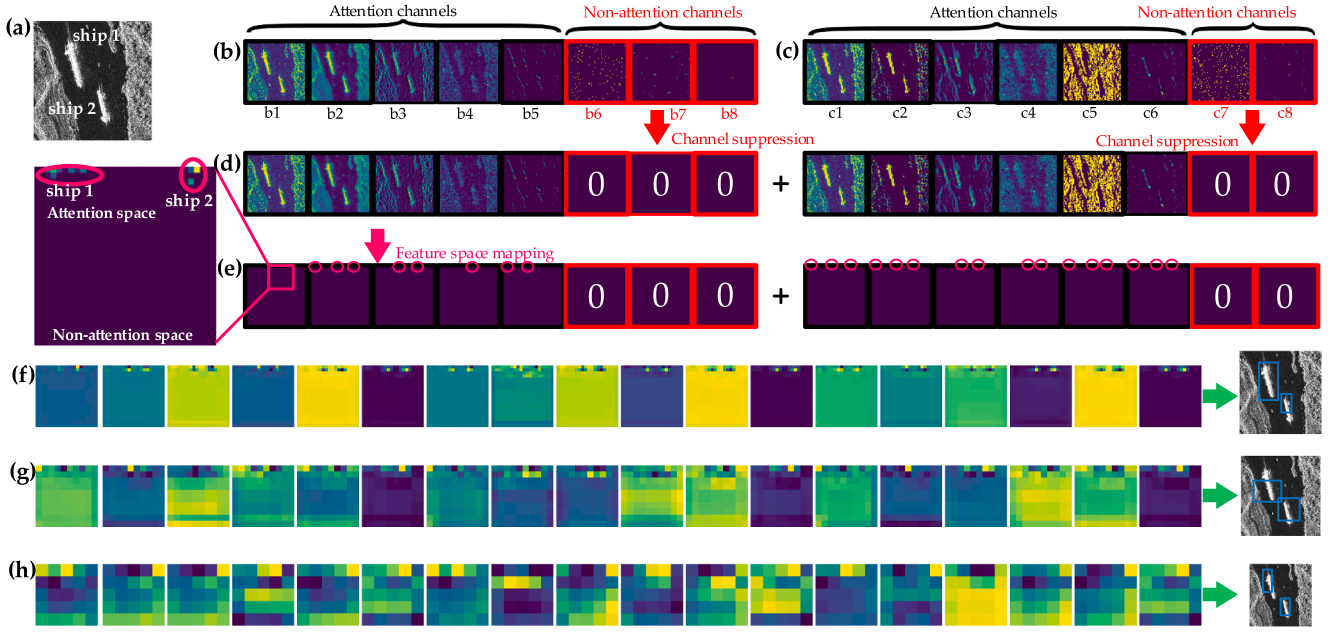



**Fig. 17.** Visualization of Feature Maps. **(a)** Inputs; **(b)** Outputs of MRF-Module (Concate-1); **(c)** Outputs of DC-Module (Concate-2); **(d)** Outputs of CA-Module-1; **(e)** Outputs of SA-Module-1; **(f)** Outputs of Small-Scale $L/8$; **(g)** Outputs of Medium-Scale $L/16$; **(h)** Outputs of Big-Scale $L/32$.

importance of ship 1 and ship 2 is similar, so they are all located at the top of the feature map, after undergoing feature space mapping (feature concentration). Of course, in our work, all ships after feature concentration are all located at the top of the feature map due to their similar importance.

In Fig. 17e, another point to be noticed is that some feature maps generate only one ship region (missed-detection only during intermediate process) meanwhile some feature maps generate three ship regions (false-alarm only during intermediate process), but these missed-detection and false-alarm cases take up a small percentage among total channels, so finally satisfactory detection results can still be obtained.

Fig. 17f shows the all output feature maps in small-scale detection $L/8$ from Separa-Conv-17 layer ($20 \times 20 \times 18$). Fig. 17g shows the all output feature maps in medium-scale detection $L/16$ from Separa-Conv-16 layer ($10 \times 10 \times 18$). Fig. 17h shows the all output feature maps in big-scale detection $L/32$ from Separa-Conv-15 layer ($5 \times 5 \times 18$). From Fig. 17f-h, with the deepening of the network, the features extracted by HyperLi-Net have exceeded our cognition, but amazingly they can be well-understood by computers, probably coming from their exsiting certain logical-relationships just not discovered by humans. Finally, HyperLi-Net will generate ship detection bounding boxes based on $20 \times 20$, $10 \times 10$ and $5 \times 5$ grid cells.

## 5. Results

### 5.1. Ship detection results

Fig. 18 shows the detection results of HyperLi-Net on three datasets. To be clear, we only show the detection results of typical complex scenes and some samples with too simple scenes are not shown. From Fig. 18a-c, most ships in SSDD, Gaofen-SSDD and Sentinel-SSDD with differnet sizes under various backgrounds can be correctly detected.

However, there are some failed detection cases, for example:

(1) Some background are mistakenly detected as ships (false alarm), coming from the high-similarity between ships and inshore facilities and repeated detection of the same ship. One possible reason is that HyperLi-Net may be sensitive to inshore port facilities, but the deep-seated reason is unknown for deep learning's "black-box" model.
(2) Some ships are not detected (missed detection), coming from side by side parking of inshore ships and small-dense distribution ships, bound to create much difficulties for detection. One possible reason is that HyperLi-Net's feature pyramid is not robust enough. Equally, the deep-seated reason is unknown for deep learning's "black-box" model.

Appreciably, ships in those SAR images with severe speckle noise can also be successfully detected, so the powerful anti-noise capacity of HyperLi-Net can be fully reflected, which can smoothly overcome the defects of bad anti-noise capacity from traditional concrete-feature-based methods. Moreover, the detection performance of small-size ships is still satisfactory, originating from multi-scale detection means ($L/8$, $L/16$ and $L/32$), although some missed-detections still modestly exist. Finally, we also find that the detecion performance of offshore ships is superior to that of inshore ships, which is in line with common sense, because more ship-like facilities in harbour make detection more challenging.

Table 9 shows the evaluation indexes on three datasets, and the following conclusions can be drawn:

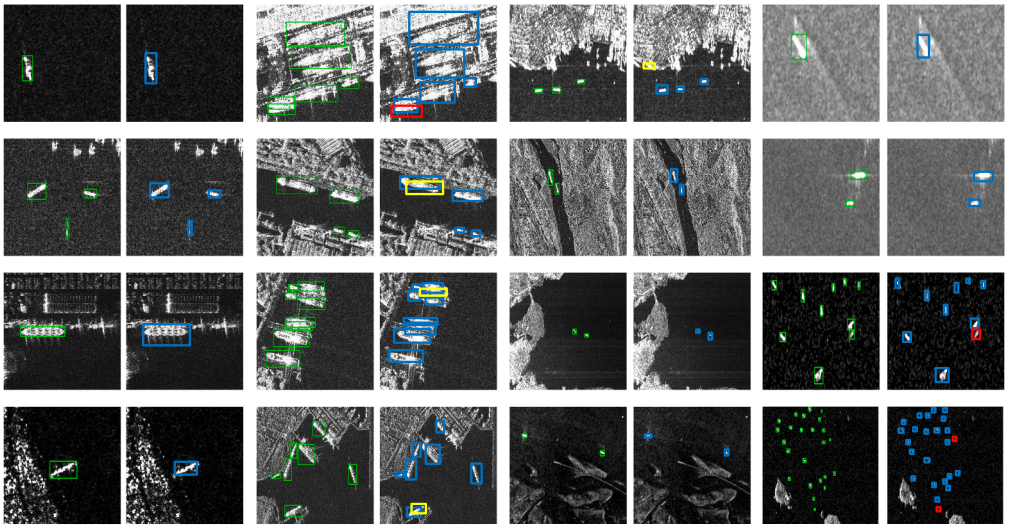
(1) HyperLi-Net's detection accuracy on SSDD is 96.08% mAP. There are 184 real ships in the test set of SSDD. Thereinto, 178 ships are successfully detected with high detection probability ($P_d$) of 96.74%. There are 6 ships being missed with low missed-detection probability ($P_m$) of 3.26%. For another thing, there are 19 false-alarm cases emerging with false-alarm probability ($P_f$) of 9.64%. One possible reason is that many ship-like harbor facilities are mistakenly detected as ships due to the simple and crude grid division mechanism coming from one-stage object detectors' basic concept. HyperLi-Net's detection speed on SSDD is 222 FPS. It takes only 4.51 ms to accomplish ship detection task in a SAR image with $160 \times 160$ size, i.e., only 523.16 ms is needed for detecting all 116 SAR images in the test set.
(2) HyperLi-Net's detection accuracy on Gaofen-SSDD is 88.20% mAP. There are 2979 real ships in the test set of Gaofen-SSDD. Thereinto, 2723 ships are successfully detected with detection probability ($P_d$) of 91.41%. There are 256 ships being missed with missed-detection probability ($P_m$) of 8.59%. For another thing, there are 446 false-alarm cases emerging with false-alarm probability ($P_f$) of 14.07%. The accuracy on Gaofen-SSDD is inferior to that on SSDD, because-1) images in Gaofen-SSDD have more complex scenarios and these images possessing many inshore ships account for a large proportion of the total test set, inevitably bringing great obstacles to detect; 2) images from low-resolution Gaofen-3 are frequently accompanied with severe speckle noise, declining the detection accuracy. HyperLi-Net's detection speed on Gaofen-SSDD is 247 FPS. It takes only 4.05 ms to accomplish ship detection tasks in a $160 \times 160$ SAR image. Thus, only 8.1 s is needed for detecting all 2,000 SAR images in the test set, showing the amazing detection speed of HyperLi-Net. The detecion speed on Gaofen-SSDD is superior to that on SSDD, given that SAR images in SSDD possess different sizes, so the image resampling into $160 \times 160$ takes extra time, while that in Gaofen-SSDD are all $160 \times 160$ sizes, so image resampling is avoided.
(3) HyperLi-Net's detection accuracy on Sentinel-SSDD is 89.11% mAP. There are 2258 real ships in the test set of Sentinel-SSDD. Thereinto, 2046 ships are successfully detected with detection probability ($P_d$) of 90.61%. There are 212 ships being missed with missed-detection probability ($P_m$) of 9.39%. For another thing, there are 174 false-alarm cases emerging with false-alarm probability ($P_f$) of 7.84%. The detection accuracy on Sentinel-SSDD is slightly superior to that on Gaofen-SSDD, because, for one thing, SAR images from high-resolution Sentinel-1 satellite have better quality than that from low-resolution Gaofen-3 satellite, scarcely accompanied with severe speckle noise. For another thing, small ships with dense distribution account for smaller proportion among the total test set of Sentinel-SSDD than Gaofen-SSDD's. The detection speed on Sentinel-SSDD of HyperLi-Net is 248 FPS that is similar to Gaofen-SSDD. It takes only 4.03 ms to accomplish ship detection tasks in a SAR image with $160 \times 160$ size. Therefore, only 8.06 s is needed for detecting all 2,000 SAR images in the test set.

### 5.2. Compared to State-Of-The-Art

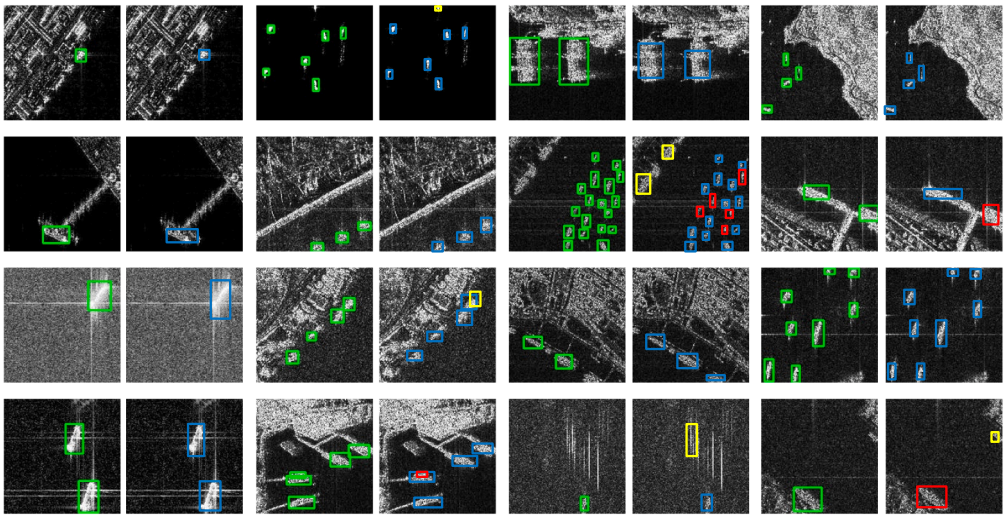
To verify HyperLi-Net's good performance, nine state-of-the-art detectors are used to compare. These detectors are all trained again on our three SAR datasets that is similar to the other scholars' practice of SAR ship detection filed (Zhang et al., 2019; Wang et al., 2018; Cui et al., Nov. 2019; Deng et al., 2018; Wei et al., 2020; Mao et al., 2020; Li et al., 2017; Li et al., 2019; Li et al., 2019; Li et al., 2019). Moreover, traditional methods' accuracy (e.g. Global $G^0$ CFAR, Two-stage CFAR, etc. (Cui et al., Nov. 2019), is far inferior to deep learing, so we ignore their comparison. Fig. 19 shows the results of different methods on three datasets.

Table 10 shows the evaluation indexes of different methods. From Table 10, the following conclusions can be drawn:

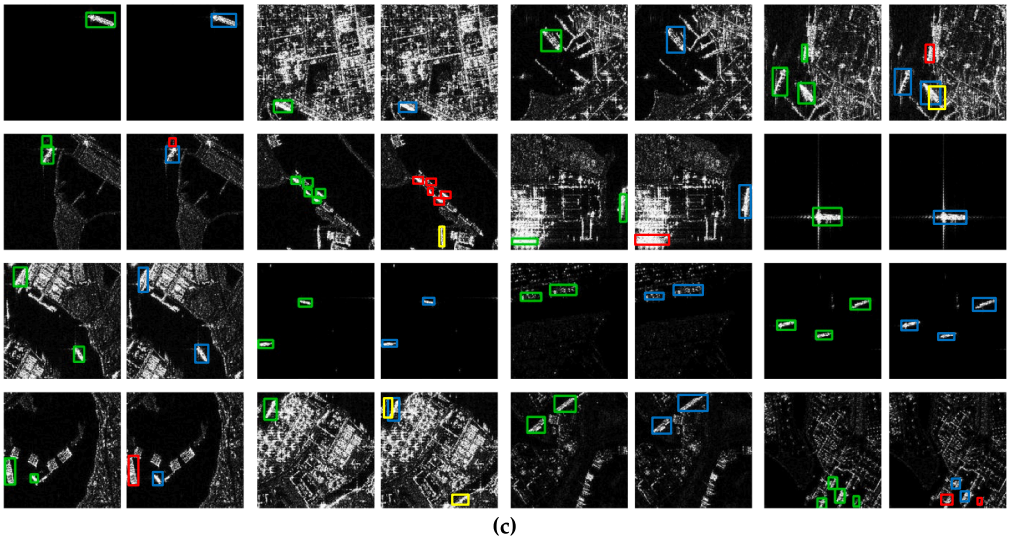
(1) HyperLi-Net's detection accuracy on SSDD is superior to all other state-of-the-art detectors. The second highest detection accuracy is 95.68% mAP of RetinaNet, still lower 96.08% mAP of HyperLi-Net.



(a)

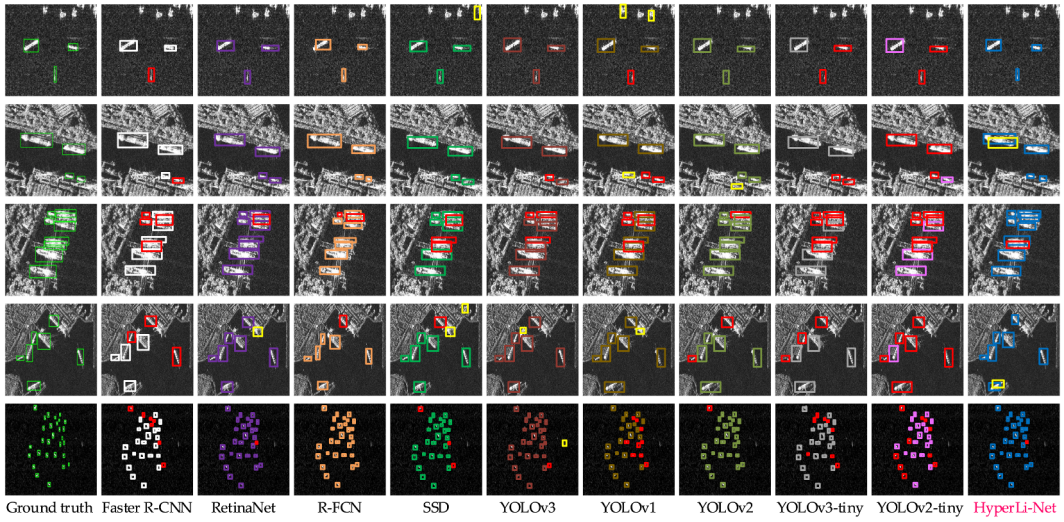


(b)

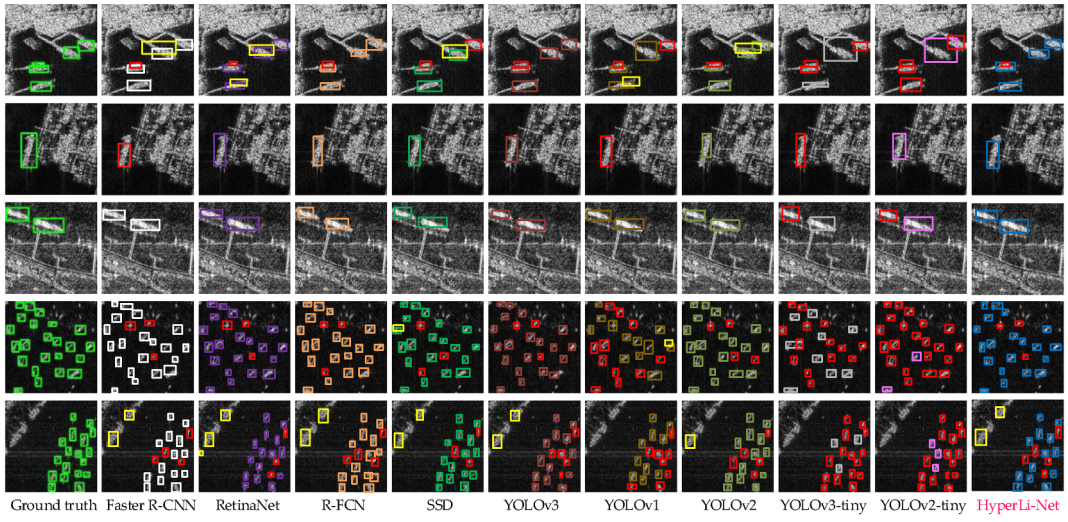


(c)

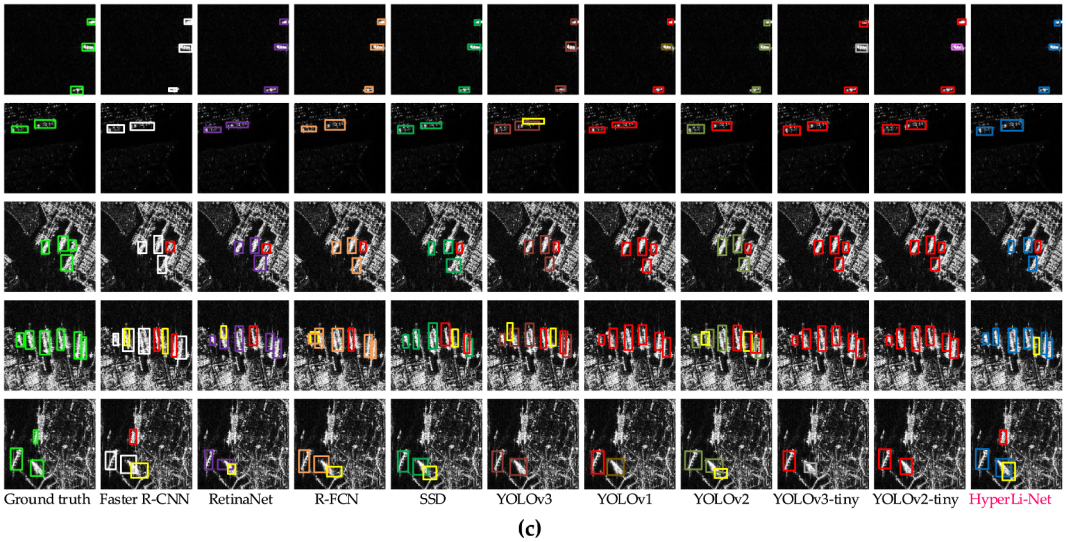
**Fig. 18.** SAR Ship Detection Results of HyperLi-Net on Three Datasets. **(a)** SSDD; **(b)** Gaofen-SSDD; **(c)** Sentinel-SSDD. Ground truths are marked in green, correct detections are marked in blue, missed-detections are marked in red, false-alarms are marked in yellow. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)




**(a)**




**(b)**




**(c)**

**Fig. 19.** Ship Detection Results of Different Methods. **(a)** SSDD; **(b)** Gaofen-SSDD; **(c)** Sentinel-SSDD. Missed-detections are marked in red, false-alarms are marked in yellow. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Thus, HyperLi-Net indeed achieves high-accurate SAR ship detection successfully. HyperLi-Net's detection speed on SSDD is also superior to others. More prominently, HyperLi-Net's detection speed is faster than others' by several or even tens of times. The second fastest speed is 107 FPS of YOLOv2-tiny, still lower 222 FPS of HyperLi-Net. Thus, HyperLi-Net indeed achieves high-speed SAR ship detection. All other object detectors on SSDD cannot achieve a good balance between accuracy and speed. For example, RetinaNet's accuracy is 95.68% mAP while its speed is 3 FPS. YOLOv2-tiny's speed is 107 FPS while its accuracy is 44.40% mAP.

(2) HyperLi-Net's detection accuracy on Gaofen-SSDD is superior to all other state-of-the-art detectors. The second highest accuracy is 87.55% mAP of YOLOv3, still lower 88.20% mAP of HyperLi-Net. HyperLi-Net's detection speed on Gaofen-SSDD is also superior to others. The second fastest speed is 121 FPS of YOLOv2-tiny, still lower 247 FPS of HyperLi-Net. All other object detectors on Gaofen-SSDD cannot achieve a good balance between accuracy and speed. Notably, HyperLi-Net can achieve both high-speed and high-accurate ship detection.

(3) HyperLi-Net's detection accuracy on Sentinel-SSDD is superior to all other state-of-the-art detectors. The second highest accuracy is 88.89% mAP of RetinaNet, still lower 89.11% mAP of HyperLi-Net. HyperLi-Net's detection speed on Sentinel-SSDD is also superior to others. The second fastest speed is 120 FPS of YOLOv2-tiny, still lower 248 FPS of HyperLi-Net. All other object detectors on Sentinel-SSDD cannot achieve a good balance between accuracy and speed. Notably, HyperLi-Net can achieve both high-speed and high-accurate ship.

Fig. 20a-d, Fig. 21a-d and Fig. 22a-d respectively show $P_d$-$P_f$ curves, P-R curves, mAP-IoU curves and mAP-FPS bar graph of different methods on SSDD, Gaofen-SSDD and Sentinel-SSDD.

Finally, in order to verify HyperLi-Net' hyper-light attribute, we compared different methods' parameter quantity, computational cost and model size. From Table 11, HyperLi-Net's model is smaller than others', by tens even hundreds of times, meanwhile its parameter quantity and computational cost are both the fewest. Thus, HyperLi-Net indeed is a hyper-light deep learning network bringing faster speed than all other state-of-the-art object detectors.

### 5.3. Migration ability verification

In order to confirm the powerful migration capability and practical application value of HyperLi-Net, we perform actual ship detection in another two large-size and wide-region Sentinel-1 SAR images that are obtained from Sentinels Scientific Data Hub (Copernicus Open Access Hub, 2019). From Table 12, the size of image 1 is 1375 × 907 and the size of image 2 is 1313 × 908, so we respectively divide these two SAR images into 45 sub-images via direct width and height average blocking. As a result, the average size of these sub-images is about 150 × 150. Afterwards, these 45 sub-images with 150 × 150 size are inputted into HyperLi-Net. As mentioned before, HyperLi-Net will resize them into 160 × 160 size for detecting ships in them.

Fig. 23 shows their ship detection results. From Fig. 23, most ships in these two images can be successfully detected. In image 1, there are 2 missed detection cases meanwhile only 1 false alarm cases. In image 2, there are 3 missed detection cases meanwhile 5 false alarm cases. Obviously, the scene of image 2 is more complex than that of image 1 (lots of ship-like facilities onshore and many parallel parking ships), so its detecion results are slightly inferior.

Table 13 shows the quantitative evaluation results of HyperLi-Net's detection performance on these two SAR images.

From Table 13, the following conclusions can be drawn:

(1) The detection accuracy of HyperLi-Net on image 1 is 92.49% mAP, and that on image 2 is 90.91% mAP. For image 1, there are 26 ships being successfully detected among the 28 real ships with detection probability ($P_d$) of 92.86%, and there are 2 missed detection cases with missed-detection probability ($P_m$) of 7.14% meanwhile only 1 false alarm cases with false-alarm probability ($P_f$) of 3.70%. For image 2, there are 42 ships being successfully detected among the 45 real ships with detection probability ($P_d$) of 93.33%, and there are 3 missed detection cases with missed-detection probability ($P_m$) of 6.67% meanwhile 5 false alarm cases with false-alarm probability ($P_f$) of 10.64%. Image 1 has better detection accuracy than Image 2, because the background of Image 2 is more complex.

(2) The detection speed of HyperLi-Net on image 1 is basically similar to that on image 2. The slight gap may come from inevitable random error. In addition, in Table 13, the time of preprocessing refers the time to divide large original images into small sub-images, by using OpenCV (OpenCV, 2019), an image processing tool. Therefore, detecting the whole image 1 needs 227.22 ms, and detecting the whole image 2 needs 226.66 ms. For such SAR images with about 1300 × 900 size, the total detection time is far < 1 s, showing the amazing detection speed of HyperLi-Net.

Given the above, HyperLi-Net has powerful migration and generalization capabilities. It has high detection accuracy in actual application meanwhile it also has notable detection speed, contributing to real-time SAR application occasions.

## 6. Ablation study

We make the follwing ablation studies on SSDD to verify the effectiveness of five external modules and five internal mechanisms. In addition, in most cases, the conclusions from SSDD equally hold on Gaofen-SSDD and Sentinel-SSDD. Considering limited pages, we will not redundantly show their experimental results any more.

### 6.1. Ablation study on five external modules

#### 6.1.1. Ablation study on Module 1: MRF-Module

We conduct two comparative experiments to discuss the influence of MRF-Module on the detection performance.

Table 14 shows the ablation study on MRF-Module, and the following conclusions can be drawn:

(1) The detection accuracy of MRF-Module is obviously superior to that of non-MRF-Module (96.08% mAP > 94.83% mAP). This shows that MRF-Module can indeed improve detection accuracy.

(2) The detection speed of MRF-Module is slightly inferior to that of non-MRF-Module. MRF-Module increases more parameters, more FLOPs, and model size, compared with non-MRF-Module. However, it does not decline the detection speed too much, because, as shown in Fig. 5a, the convolution operations from 1 × 1, 3 × 3 and 5 × 5 kernels are simultaneously executed in parallel on GPU, only slightly affect the total detection time.

#### 6.1.2. Ablation study on Module 2: DC-Module

We conduct two comparative experiments to discuss the influence of DC-Module on the detection performance.

Table 15 shows the ablation study on DC-Module, and the following conclusions can be drawn:

(1) The detection accuracy of DC-Module is obviously superior to non-DC-Module (96.08% mAP > 94.17% mAP). This shows that DC-Module can indeed improve detection accuracy.

(2) The detection speed of DC-Module is slightly inferior to non-DC-Module. DC-Module increases more parameters, more FLOPs and model size, but it does not decline speed too much, because in Fig. 5b, the convolution operations from dilated 1 × 1, 3 × 3, 5 × 5 kernels are executed in parallel on GPU, only slightly

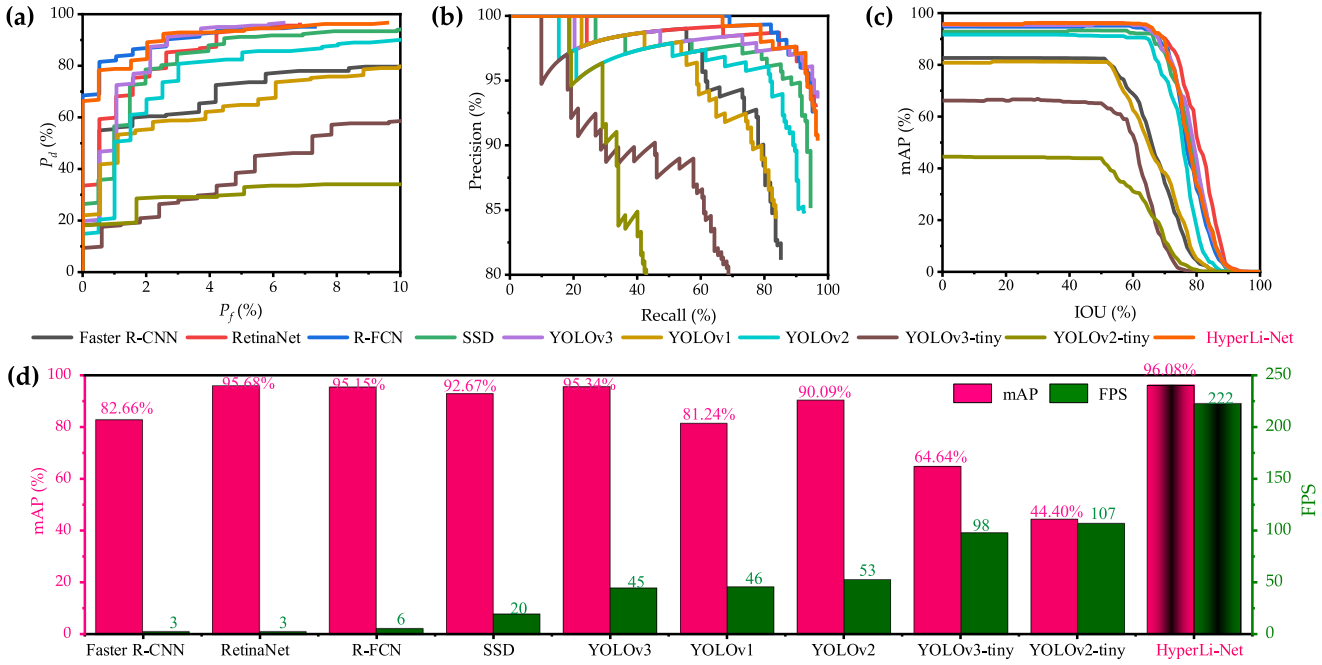



**Fig. 20.** Evaluation Curves of Different Methods on SSDD. **(a)** $P_d$-$P_f$ curves; **(b)** P-R curves; **(c)** mAP-IoU curves. **(d)** mAP-FPS.

affecting the total time.

Moreover, we also conduct five comparative experiments to discuss the dilated rate $d$ on the detection performance. In these five experiments, $d$ must be even numbers in order to make receptive field symmetrical when zeropadding, so we respectively set $d = 2$, 4, 6, 8 and 10. Table 16 shows the ablation study on dilated rates, and the following conclusions can be drawn:

(1) The detection accuracy of $d = 2$ is the best. One possible reason is that larger $d$ will make the extracted features appear serious discontinuities (too many holes), leading to missing features (Yu and Koltun, 2015).

(2) The detection speed decreases with the increase of $d$ (from 222 FPS of $d = 2$ to 203 FPS of $d = 10$). Although their model sizes are equal for their same parameter quantity, the dilation process requires extra program execution time.

Finally, we set $d = 2$ in DC-Module, because-1) for one thing, the detection accuracy reaches the best; 2) for one thing, the detection speed is also the fastest.

### 6.1.3. Ablation study on Module 3: CSA-Module

We conduct four comparative experiments to discuss the influence

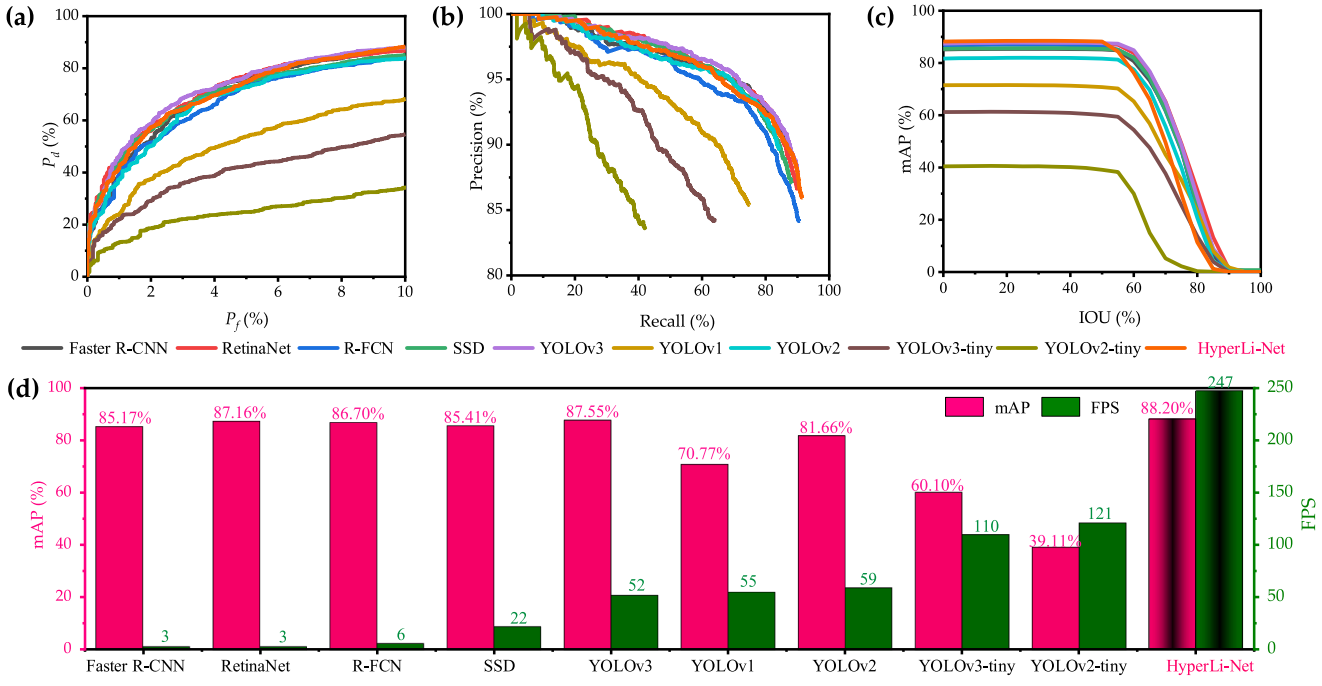



**Fig. 21.** Evaluation Curves of Different Methods on Gaofen-SSDD. **(a)** $P_d$-$P_f$ curves; **(b)** P-R curves; **(c)** mAP-IoU curves. **(d)** mAP-FPS.

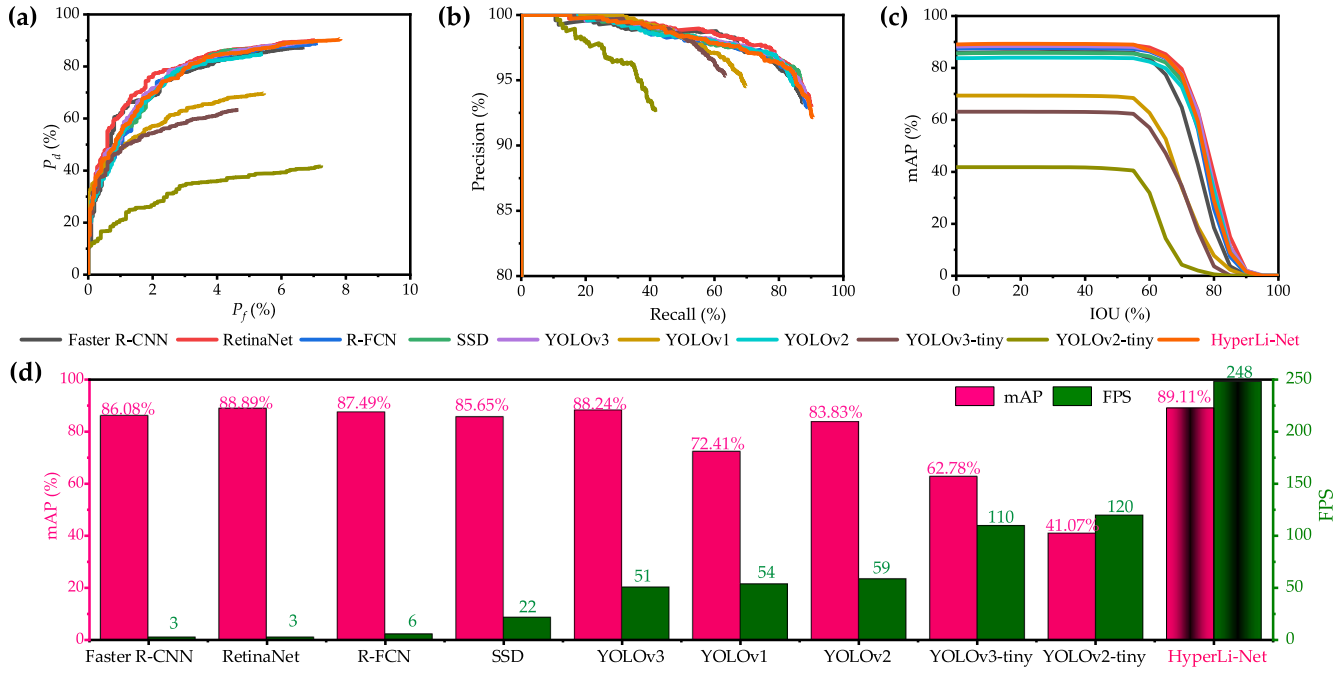



**Fig. 22.** Evaluation Curves of Different Methods on Sentinel-SSDD. **(a)** $P_d$-$P_f$ curves; **(b)** P-R curves; **(c)** mAP-IoU curves. **(d)** mAP -FPS.

**Table 11**
Parameter Quantity, Computational Cost and Model Size of Different Object Detectors.

| Object Detectors | Parameter Quantity | Computational Cost (FLOPs) | Model Size |
|---|---|---|---|
| YOLOv1 | 272,746,867 | 545,429,460 | 752.75 MB |
| YOLOv3 | 61,576,342 | 307,592,895 | 235.44 MB |
| YOLOv2 | 50,578,686 | 101,385,166 | 193.04 MB |
| R-FCN | 47,663,806 | 95,040,404 | 181.24 MB |
| RetinaNet | 36,382,957 | 72,545,184 | 139.25 MB |
| Faster R-CNN | 28,342,195 | 46,981,897,900 | 108.54 MB |
| SSD | 23,745,908 | 118,685,133 | 90.73 MB |
| YOLOv2-tiny | 15,770,510 | 31,608,360 | 60.22 MB |
| YOLOv3-tiny | 8,676,244 | 86,692,284 | 33.20 MB |
| **HyperLi-Net (Ours)** | **103,754** | **203,570** | **0.69 MB** |

of CSA-Module on the detection performance.

Table 17 shows the ablation study on CSA-Module, and the following conclusions can be drawn:

(1) The detection accuracy of CA-SA is obviously superior to others'. Spatial attention can improve detection accuracy meanwhile channel attention does, compared with non-attention of both channel and spatial. Combined with CA and SA, the accuracy reaches the best.
(2) The detection speed of CA-SA is the lower than others', because more parameters and more FLOPs are produced, leading to bigger model size.

Finally, we add CSA-Module into HyperLi-Net, because-1) for one thing, the detection accuracy obtains obvious improvement; 2) for another thing, the detection speed does not sacrifice too much, which is still superior to exsiting other state-of-the-art object detectors.

#### 6.1.4. Ablation study on Module 4: FF-Module

We conduct two comparative experiments to discuss the influence of FF-Module on the detection performance.

Table 18 shows the ablation study on FF-Module, and the following conclusions can be drawn:

(1) The detection accuracy of FF-Module is obviously superior to that of non-FF-Module. This shows that FF-Module can indeed improve detection accuracy for its more robust and contextual features.
(2) The detection speed of FF-Module is slightly inferior to that of non-FF-Module (222 FPS < 225 FPS), due to only slight increase in model size (from 0.68 MB to 0.69 MB).

#### 6.1.5. Ablation study on Module 5: FP-Module

We conduct seven comparative experiments to discuss the influence of FP-Module on the detection performance. FP-Module consists of three detection scales namely $L/8$, $L/16$, and $L/32$. Here, we respectively set one scale or two scale or all three scale in HyperLi-Net to make detailed discussion.

Table 19 shows the ablation study on FP-Module, and the following conclusions can be drawn:

(1) The detection accuracy increases with the increase of the number of detection scales. If all three detection scales are employed in HyperLi-Net, it reaches the best. Therefore, FP-Module can indeed improve detection accuracy.
(2) The detection speed decreases with the increase of the number of detection scales. This is in line with common sense, because more parameters and more FLOPs are produced, leading to bigger model size (from 0.52 MB to 0.69 MB).

**Table 12**
Detailed Descriptions of Two Wide-Region Sentinel-1 SAR images.

| Name | Cover Area | Time | Resolution | Polarization | Image Size (Width × Height) |
|---|---|---|---|---|---|
| Image 1 | State of Malacca, Malaysia | 3 Sep., 2019 | 10 *m* | VH | 1375 × 907 |
| Image 2 | Kuala Lumpur, Malaysia | 8 Jul., 2019 | 10 *m* | VV | 1313 × 908 |

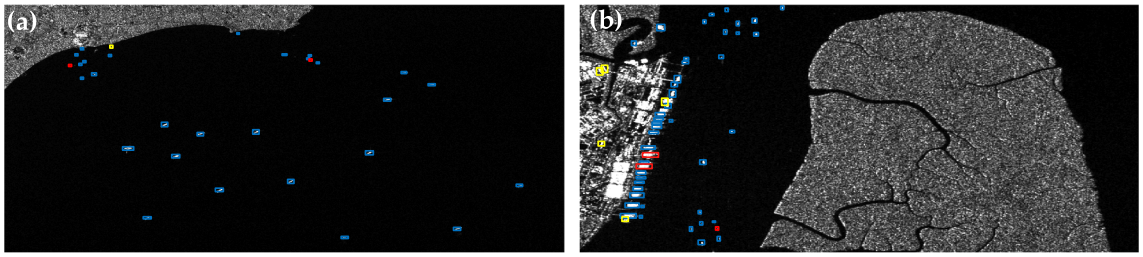



**Fig. 23.** Results on Wide-Region Sentinel-1 Images. **(a)** Image 1; **(b)** Image 2.

**Table 13**
Evaluation Indexes on Sentinel-1 Images. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Name | GT | TP | FN | FP | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Time of Preprocessing |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | |
| Image 1 | 28 | 26 | 2 | 1 | 92.86% | 7.14% | 3.70% | 92.86% | 96.30% | 92.49% | 4.15 ms | 241 | 3.12 ms |
| Image 2 | 45 | 42 | 3 | 5 | 93.33% | 6.67% | 10.64% | 93.33% | 89.36% | 90.91% | 4.14 ms | 242 | 3.10 ms |

In addition, we also conduct two comparative experiments to discuss scale fusion on the detection performance.

Table 20 shows the ablation study on scale fusion, and the following conclusions can be drawn:

(1) The detection accuracy can obtain improvement, if the scale fusion is adopted in FP-Module, regardless of down sampling or upsamplin. In addition, the upsampling fusion has better detection accuracy than the downsampling, because the features from $L/32$ scale (pyramid tip) are more representative, and if they are transmitted to the bottom of pyramid, features in $L/16$ scale and $L/8$ scale will become more robust.
(2) The detection speed of scale fusion is similar to that of non-scale fusion, because no more parameters are generated and their model sizes are all 0.69 MB.

Finally, we use FP-Module in HyperLi-Net by upsampling scale fusion. In this way, the detection accuracy are improved, and the detection speed is also not negatively affected.

### 6.2. Ablation study on five internal mechanisms

#### 6.2.1. Ablation study on mechanism 1: RF-Model

In Table 10, Faster R-CNN and R-FCN are two-stage object detectors based on R-model. RetinaNet, SSD, YOLOv1, YOLOv2, YOLOv3, YOLOv2-tiny, YOLOv3-tiny and HyperLi-Net are one-stage object detectors based on RF-Model. We can find that almost all one-stage detectors have faster detection speed than two-stage ones. Thus, it is correct that we adopt the basic concept of RF-Model to design HyperLi-Net. RetinaNet is an exception because it produces many useless anchors causing large increase in computation cost. Moreover, HyperLi-Net does not have RPNs, i.e., it inherently cannot be converted as a R-Model, so here, we cannot give the comparative experimental results of R-Model and RF-Model.

#### 6.2.2. Ablation study on mechanism 2: S-Kernel

We conduct three comparative experiments ($k$ = 1,3,5) to discuss the influence of different size kernels on the detection performance.

Table 21 shows the ablation study on S-Kernel, and the following conclusions can be drawn:

(1) The detection accuracy of $k = 3$ is the best, and that of $k = 1$ is the worst due to its limited receptive field. In addition, although $5 \times 5$ kernels have larger receptive field, its detection accuracy is lower than that of $3 \times 3$ kernels. One possible reason for this is that many small ships are missed detected if using $5 \times 5$ kernels.
(2) The detection speed decreases with the increase of $k$, because more parameters and more FLOPs are produced, leading to bigger model size. Therefore, smaller kernel can indeed improve detection speed.

Given the above, we set $k = 3$ in the backbones of HyperLi-Net. For one thing, smaller size kernels brings lighter model and faster speed. For another thing, its detection performance reaches the best.

#### 6.2.3. Ablation study on mechanism 3: N-Channel

We conduct ten comparative experiments to discuss the influence of different channel widths on the detection performance. Generally, $N_{kernel}$ is a multiple of 2, so we respectively set $N_{kernel} = 2^i$, $i$ = 1,2, …,10.

Table 22 shows the ablation study on N-Channel, and the following conclusions can be drawn:

(1) The detection accuracy shows an upward trend with the increase of $N_{kernel}$. The accuracy of $N_{kernel} = 32$ has already reached 96.08% mAP, only slightly lower than that of $N = 256$, 512,1024.
(2) The detection speed shows a downward trend with the increase of $N_{kernel}$, because more parameters and more FLOPs are produced, leading to bigger model size. Therefore, narrower channel can indeed improve detection speed.

Finally, we set $N_{kernel} = 32$ in HyperLi-Net. For one thing, it is

**Table 14**
Ablation Study on MRF-Module. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| MRF-Module? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 95.65% | 4.35% | 11.11% | 95.65% | 88.89% | 94.83% | 4.44 ms | 225 | 43,640 | 84,314 | 0.43 MB |
| ✓ | **96.74%** | **3.26%** | **9.64%** | **96.74%** | **90.36%** | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 15**
Ablation Study on DC-Module. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| DC-Module? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 95.11% | 4.89% | **6.91%** | 95.11% | **93.09%** | 94.17% | 4.45 ms | 225 | 43,640 | 84,314 | 0.43 MB |
| ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 16**
Ablation Study on Dilated Rates. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Dilated Rate ($d$) | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| 2 | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| 4 | 94.57% | 5.43% | 7.94% | 94.57% | 92.06% | 94.03% | 4.55 ms | 220 | 103,754 | 203,570 | 0.69 MB |
| 6 | 96.20% | 3.80% | 8.76% | 96.20% | 91.24% | 95.11% | 4.59 ms | 218 | 103,754 | 203,570 | 0.69 MB |
| 8 | 95.11% | 4.89% | 11.62% | 95.11% | 88.38% | 94.34% | 4.65 ms | 215 | 103,754 | 203,570 | 0.69 MB |
| 10 | 95.11% | 4.89% | **7.89%** | 95.11% | **92.11%** | 94.40% | 5.09 ms | 203 | 103,754 | 203,570 | 0.69 MB |

**Table 17**
Ablation Study on CSA-Module. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| CA? | SA? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | ✗ | 92.93% | 7.07% | 9.52% | 92.93% | 90.48% | 91.44% | 3.89 ms | 260 | 22,618 | 43,018 | 0.26 MB |
| ✗ | ✓ | 94.02% | 5.98% | **8.47%** | 94.02% | 91.53% | 93.45% | 4.12 ms | 244 | 23,018 | 43,826 | 0.30 MB |
| ✓ | ✗ | 94.02% | 5.98% | 7.49% | 94.02% | **92.51%** | 93.21% | 4.33 ms | 237 | 103,354 | 202,762 | 0.65 MB |
| ✓ | ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

already universally narrower than other object detectors, and we cannot set smaller width for poor accuracy. For another thing, although detection accuracy of $N_{kernel} = 256$, 512, 1024 is better than $N_{kernel} = 32$, but-1) their gap is small; 2) 96.08% mAP of $N_{kernel} = 32$ is already superior to other object detectors; 3) more parameters, more FLOPs, and bigger model will be produced when $N_{kernel} > 32$, leading to lower speed, otherwise, HyperLi-Net's advantages will not be fully reflected compared with the other state-of-the-art object detectors.

#### 6.2.4. Ablation study on mechanism 4: Separa-Conv

We conduct two comparative experiments to discuss the influence of Separa-Conv on the detection performance. Separa-Conv is employed in HyperLi-Net in one experiment, and Trad-Conv is employed in another experiment.

Table 23 shows the ablation study on Separa-Conv, and the following conclusions can be drawn:

(1) The detection accuracy of Trad-Conv is slightly superior to Separa-Conv, but their gap is small. This fully shows that Separa-Conv does not sacrifice accuracy too much, due to Trad-Conv exists certain parameter redundancy.
(2) The detection speed of Separa-Conv is obviously superior to that of Trad-Conv. This is in line with common sense, because more parameters and more FLOPs are produced, leading to bigger model size. Therefore, in HyperLi-Net, the mechanism that we replace Trad-Conv with Separa-Conv is effective for improving the detection speed.

#### 6.2.5. Ablation study on mechanism 5: BN-Fusion

We conduct two comparative experiments to discuss the influence of BN-Fusion on the detection performance.

Table 24 shows the ablation study on BN-Fusion, and the following conclusions can be drawn:

(1) The detection accuracy of BN-Fusion is fully equal to non-BN-Fusion, because the process of BN-Fusion is lossless, just combining two steps into one step via manual participation.
(2) The detection speed of BN-Fusion is obviously superior to non-BN-Fusion, because more parameters and more FLOPs are produced, leading to bigger model size. Therefore, BN-Fusion can indeed improve detection speed.

In addition, we also conduct two comparative experiments to discuss the existing necessity of BN during training.

Table 25 shows the ablation study on BN-training, and the following conclusions can be drawn:

(1) The detection accuracy of BN-Training is obviously superior to that of non-BN-Training because BN can accelerate deep network training (not detection) via reducing internal covariate shift (Sifre, 2014). In other words, HyperLi-Net can be fully trained with the help of BN, so adequate training loss decrease can be achieved.

**Table 18**
Ablation Study on FF-Module. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| FF-Module? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 95.11% | 4.89% | **4.89%** | 95.11% | **95.11%** | 94.25% | 4.45 ms | 225 | 101,834 | 199,922 | 0.68 MB |
| ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 19**
Ablation Study on FP-Module. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| $L/8$ | $L/16$ | $L/32$ | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✓ | ✗ | ✗ | 84.78% | 15.22% | 12.85% | 84.78% | 88.64% | 82.71% | 4.35 ms | 230 | 91,166 | 178,872 | 0.52 MB |
| ✗ | ✓ | ✗ | 84.78% | 15.22% | 12.85% | 84.78% | 87.15% | 83.54% | 4.38 ms | 228 | 96,162 | 188,674 | 0.59 MB |
| ✗ | ✗ | ✓ | 89.67% | 10.22% | 10.81% | 89.67% | 89.19% | 88.45% | 4.44 ms | 225 | 101,158 | 198,477 | 0.66 MB |
| ✓ | ✓ | ✗ | 93.48% | 6.52% | 8.99% | 93.48% | 91.01% | 92.09% | 4.41 ms | 227 | 97,460 | 191,221 | 0.61 MB |
| ✓ | ✗ | ✓ | 95.65% | 4.35% | **6.88%** | 95.65% | **93.12%** | 95.05% | 4.49 ms | 223 | 102,456 | 201,023 | 0.68 MB |
| ✗ | ✓ | ✓ | 95.65% | 4.35% | 16.19% | 95.65% | 83.81% | 94.21% | 4.49 ms | 223 | 102,456 | 201,023 | 0.68 MB |
| ✓ | ✓ | ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 20**
Ablation Study on Scale Fusion. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Scale Fusion? | DownSampling? | UpSampling? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS |
| ✗ | – | – | 93.48% | 6.52% | 10.88% | 93.48% | 89.12% | 92.59% | 4.50 ms | 222 |
| ✓ | ✓ | – | 95.11% | 4.89% | **9.33%** | 95.11% | **90.67%** | 94.46% | 4.51 ms | 222 |
| ✓ | – | ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 |

**Table 21**
Ablation Study on S-Kernel. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Kernel Size ($k$) | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| 1 | 86.41% | 13.59% | 22.44% | 86.41% | 77.56% | 82.09% | 4.44 ms | 225 | 100,426 | 196,914 | 0.68 MB |
| 3 | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| 5 | 95.65% | 4.35% | **7.85%** | 95.65% | **92.15%** | 94.70% | 5.57 ms | 219 | 110,410 | 216,882 | 0.72 MB |

**Table 22**
Ablation Study on N-Channel. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| $N_{kernel}$ | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| 2 | 63.59% | 36.41% | 17.61% | 63.59% | 82.39% | 60.75% | 4.21 ms | 237 | 1,664 | 2,990 | 0.30 MB |
| 4 | 73.91% | 26.09% | 19.53% | 73.91% | 80.47% | 69.87% | 4.28 ms | 234 | 3,374 | 6,170 | 0.31 MB |
| 8 | 85.33% | 14.67% | 15.59% | 85.33% | 84.41% | 83.30% | 4.37 ms | 229 | 8,978 | 16,898 | 0.33 MB |
| 16 | 92.39% | 7.61% | 12.82% | 92.39% | 87.18% | 91.32% | 4.44 ms | 225 | 28,922 | 55,826 | 0.41 MB |
| 32 | 96.74% | 3.26% | 9.64% | 96.74% | 90.36% | 96.08% | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| 64 | 95.65% | 4.35% | 5.88% | 95.65% | 94.12% | 95.00% | 4.78 ms | 209 | 393,194 | 778,610 | 1.80 MB |
| 128 | 96.74% | 3.26% | 5.82% | 96.74% | 94.18% | 95.38% | 5.24 ms | 191 | 1,531,178 | 3,046,898 | 6.14 MB |
| 256 | 97.28% | 2.72% | 5.79% | 97.28% | 94.21% | 96.25% | 6.38 ms | 157 | 6,043,562 | 12,056,306 | 23.35 MB |
| 512 | 97.83% | 2.17% | 4.76% | 97.83% | 95.24% | 96.76% | 8.56 ms | 117 | 24,013,994 | 47,966,450 | 91.91 MB |
| 1024 | **97.83%** | **2.17%** | **4.76%** | **97.83%** | **95.24%** | **97.06%** | 15.10 ms | 66 | 95,737,514 | 191,352,050 | 365.51 MB |

**Table 23**
Ablation Study on Separa-Conv. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| Type | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| Trad-Conv | **96.74%** | **3.26%** | **5.82%** | **96.74%** | **94.14%** | 96.10% | 5.13 ms | 195 | 247,550 | 492,204 | 1.17 MB |
| Separa-Conv | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 24**
Ablation Study on BN-Fusion. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| BN-Fusion? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 96.74% | 3.26% | 9.64% | 96.74% | 90.36% | 96.08% | 5.64 ms | 178 | 107,282 | 211,788 | 0.82 MB |
| ✓ | 96.74% | 3.26% | 9.64% | 96.74% | 90.36% | 96.08% | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

**Table 25**
Ablation Study on BN-Training. Note: mAP is detection accuracy defined by Eq. (31); FPS is detection speed defined by Eq. (32).

| BN Training? | Detection Accuracy (Remote Sensing Community) | | | Detection Accuracy (Deep Learning Community) | | | Detection Speed | | Model | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_d$ | $P_m$ | $P_f$ | Recall | Precision | mAP | Time | FPS | Parameters | FLOPs | Size |
| ✗ | 95.11% | 4.89% | **8.85%** | 95.11% | **91.15%** | 93.64% | 4.50 ms | 222 | 103,754 | 203,570 | 0.69 MB |
| ✓ | **96.74%** | **3.26%** | 9.64% | **96.74%** | 90.36% | **96.08%** | 4.51 ms | 222 | 103,754 | 203,570 | 0.69 MB |

(2) The detection speed of BN-Training is equal to that of non-BN-Training, because they have the same parameter quantity, FLOPs, and model size. The former processed by BN-Fusion, has the same network structure as the latter's.

Finally, during training, we retain BN in HyperLi-Net. After obtaining the initial model, we fuse BN into Separa-Conv to get the final detection model, not containing BN layers. In this way, HyperLi-Net can-1) not only achieve higher detection accuracy (96.08% mAP > 93.64% mAP); 2) but also achieve faster detection speed (222 FPS > 178 FPS).

Another needed to be clarified is that HyperLi-Net's training model can be converted into detection model, but its detection model cannot be converted into training model, because the parameters' information in BN has already been lost during the BN-Fusion process.

## 7. Conclusions

In this paper, in view of the situation that previous most studies are frequently improving detection accuracy at the expense of detection speed, we propose HyperLi-Net to achieve both high-speed and high-accurate SAR ship detection. Five internal mechanisms are adopted in HyperLi-Net for high-speed ship detection meanwhile five external modules are proposed for high-accurate ship detection. We have emphatically verified the correctness and effectiveness of the above ten means, through lots of experiments and detailed discussions. Experimental results on SSDD, Gaofen-SSDD and Sentinel-SSD show that HyperLi-Net's detection accuracy and speed are both superior to the other nine state-of-the-art object detectors. Moreover, the satisfactory detection results on two wide-region Sentinel-1 images show HyperLi-Net's strong migration capability. HyperLi-Net is built from scratch with fewer parameters, lower computation costs and lighter model. It can also avoid the trouble of pre-training on ImageNet enhancing training efficiency. For its light model, it can be efficiently trained on CPUs, reducing hardware cost caused by GPUs. Finally, it is helpful for future hardware transplantation and for ship detection of in-orbit missile-borne/satellite-borne SAR.

The reasons, why such a hyper-light model can achieve such high detection accuracy, originate from:

(1) Five external modules added in HyperLi-Net indeed play an important role that can greatly improve accuracy.
(2) Compared to the multi-classes object detection tasks, the task of SAR ship detection is relatively easy, which only contains binary classification of ship and background.
(3) Compared to optical images, SAR images have relatively simple backgrounds, so there is no need to use large-scale network to achieve such simple detection task. Otherwise, there must be huge redundancy in networks (Zhang et al., 2019).
(4) If object detectors with huge model size in the CV community are directly applied to SAR ship detection, they may overfit for smaller SAR datasets. However, one light model may avoid such problem (Zhang et al., 2019).
(5) During training, lighter models can be fully trained, because, all network parameters can be updated iteratively at a faster speed, make the network get a full and thorough fitting (Zhang et al., 2019).

Our future work is as follow:

(1) We will make efforts to suppress the false-alarm probability ($P_f$) of HyperLi-Net that is slightly higher than some state-of-the-art object detectors.
(2) We will continue to implement the hardware transplantation of some embedded devices, e.g., FPGAs, DSPs, etc., given the only 0.69 MB model size of HyperLi-Net.
(3) We will combine deep learning abstract features and traditional concrete ones to further improve accuracy (e.g. Ai et al. (Ai et al., 2019). So far, most scholars in this SAR ship detection community still scarcely focus on much information of SAR images and ship identification, when they applied those object detectors in the deep learning filed to this SAR ship detection field.
(4) We will make a detailed study on the domain adaptation ideas from transfer learning community, given the limited SAR data, to further enrich our work in the future (e.g. Huang et al. (Huang et al., 2020; Huang et al., 2020; Huang et al., 2017, 2020; Huang et al., 2019), Ye et al. (Ye et al., 2019).

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

The authors would like to thank the editors who handled this manuscript and the anonymous reviewers for their comments towards improving this manuscript. This work was supported in part by the National Key R&D Program of China under Grant 2017YFB0502700 and in part by the National Natural Science Foundation of China under Grants 61571099, 61501098 and 61671113.

## References

Agrawal, Anupam; Mangalraj, P.; Bisherwal, Mukul Anand. Target detection in SAR images using SIFT. Proceedings of IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), pp. 90-94, Jan., 2016. https://doi.org/10.1109/ISSPIT.2015.7394426.

Ai, Jiaqiu, Tian, Ruitian, Luo, Qiwu, Jin, Jing, Tang, Bo, 2019. Multi-Scale Rotation-Invariant Haar-Like Feature Integrated CNN-Based Ship Detection Algorithm of Multiple-Target Environment in SAR Imagery. IEEE Transactions on Geoscience and Remote Sensing 57 (12), 10070–10087. https://doi.org/10.1109/TGRS.2019.2931308.

Akagündüz, Erdem. Scale invariant sillhouette features. Proceedings of Signal Processing and Communications Applications Conference (SIU), pp. 1-4., Haspolat, 2013. https://doi.org/10.1109/SIU.2013.6531586.

An, Quanzhi, Pan, Zongxu, Liu, Lei, You, Hongjian, 2019. DRBox-v2: An Improved Detector With Rotatable Boxes for Target Detection in SAR Images. IEEE Transactions on Geoscience and Remote Sensing 57 (11), 8333–8349. https://doi.org/10.1109/TGRS.2019.2920534.

An, Wentao, Xie, Chunhua, Yuan, Xinzhe, 2014. An improved iterative censoring scheme for CFAR ship detection with SAR imagery. IEEE Transactions on Geoscience and Remote Sensing 52 (8), 4585–4595. https://doi.org/10.1109/TGRS.2013.2282820.

Anastassopoulos, Vassilis, Lampropoulos, George A., 1995. Optimal CFAR detection in Weibull clutter. IEEE Transactions on Aerospace and Electronic Systems 31 (1), 52–64. https://doi.org/10.1109/7.366292.

Atteia, G.E., Collins, Michael J., 2013. On the use of compact polarimetry SAR for ship detection. ISPRS Journal of Photogrammetry and Remote Sensing 80, 1–9. https://doi.org/10.1016/j.isprsjprs.2013.01.009.
Benachenhou, Kamel, Taleb-Ahmed, Abdelmalik, Hamadouche, Mhamed, 2013. Performances evaluation of GNSS ALTBOC acquisition with CFAR detection in Rayleigh fading channel. In: Proceedings of Saudi International Electronics, Communications and Photonics Conference (**SIECPC**), pp. 1–7. https://doi.org/10.1109/SIECPC.2013.6550786.
Biao, Hou, Chen, Xingzhong, Jiao, Licheng, 2015. Multilayer CFAR detection of ship targets in very high resolution SAR images. IEEE Geoscience and Remote Sensing Letters 12 (4), 811–815. https://doi.org/10.1109/LGRS.2014.2362955.
Bodla, Navaneeth; Singh, Bharat; Chellappa, Rama; Davis, Larry S. Soft-NMS-Improving Object Detection With One Line of Code. arXiv preprint, arXiv:1704.04503. https://arxiv.org/abs/1704.04503.
Born, G.H., Dunne, J.A., Lame, D.B., 1979. Seasat mission overview. Science 204 (4400), 1405–1406. https://doi.org/10.1126/science.204.4400.1405.
Bundy, Alan, Wallen, Lincoln, 1984. Difference of Gaussians in Catalogue of Artificial Intelligence Tools. Springer. https://doi.org/10.1007/978-3-642-96868-6_57.
Cai, Zhaowei; Vasconcelos, Nuno. Cascade R-CNN: Delving into High Quality Object Detection. arXiv preprint, arXiv:1712.00726. https://arxiv.org/abs/1712.00726.
Cai, Zhaowei, Fan, Quanfu, Feris, Rogerio S., Vasconcelos, Nuno, 2016. A unified multi-scale deep convolutional neural network for fast object detection. Proceedings of European Conference on Computer Vision (**ECCV**) 9908, 354–370. https://doi.org/10.1007/978-3-319-46493-0_22.
Chang, Yang-Lang, Anagaw, Amare, Chang, Lena, Wang, Yi Chun, Hsiao, Chih-Yu, Lee, Wei-Hong, 2019. Ship detection based on YOLOv2 for SAR imagery. Remote Sensing 11 (7), 786. https://doi.org/10.3390/rs11070786.
Chen, Liang-Chieh; Papandreou, George; Schroff, Florian; Adam, Hartwig. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv preprint, arXiv: 1706.05587. https://arxiv.org/abs/1706.05587.
Chen, Chen, He, Chuan, Hu, Changhua, Pei, Hong, Jiao, Licheng, 2019. A Deep Neural Network Based on an Attention Mechanism for SAR Ship Detection in Multiscale and Complex Scenarios. IEEE Access 7, 104848–104863. https://doi.org/10.1109/ACCESS.2019.2930939.
Chen, Peng, Li, Ying, Zhou, Hui, Liu, Bingxin, Liu, Peng, 2020. Detection of Small Ship Objects Using Anchor Boxes Cluster and Feature Pyramid Network Model for SAR Imagery. Remote Sensing 8 (2), 112. https://doi.org/10.3390/jmse8020112.
Chollet, François. Xception: Deep learning with depthwise separable convolutions. arXiv preprint, arXiv:1610.02357. https://arxiv.org/abs/1610.02357.
COCO-Common Objects in Context. Available online: http://cocodataset.org/ (accessed on 10 Nov., 2019).
Copernicus Open Access Hub. Available online: . (accessed on 6 Sep., 2019).
Cui, Zongyong, Li, Qi, Cao, Zongjie, Liu, Nengyuan, Nov. 2019. Dense Attention Pyramid Networks for Multi-Scale Ship Detection in SAR Images. IEEE Transactions on Geoscience and Remote Sensing 57 (11), 8983–8997. https://doi.org/10.1109/TGRS.2019.2923988.
Dai, Jifeng; Li, Yi; He, Kaiming; Sun, Jian. R-FCN: Object detection via region-based fully convolutional networks. arXiv preprint, arXiv:1605.06409. https://arxiv.org/abs/1605.06409.
Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (**CVPR**) 1, 886–893. https://doi.org/10.1109/CVPR.2005.177.
Deng, Zhipeng, Sun, Hao, Zhou, Shilin, Zhao, Juanping, Lei, Lin, Zou, Huanxin, 2018. Multi-scale object detection in remote sensing imagery with convolutional neural networks. ISPRS Journal of Photogrammetry and Remote Sensing 145, 3–22. https://doi.org/10.1016/j.isprsjprs.2018.04.003.
Deng, Zhipeng, Sun, Hao, Zhou, Shilin, Zhao, Juanping, 2019. Learning deep ship detector in SAR images from scratch. IEEE Transactions on Geoscience and Remote Sensing 57 (6), 4021–4039. https://doi.org/10.1109/TGRS.2018.2889353.
Dong, Chao, Liu, Jinghong, Xu, Fang, Liu, Chenglong, 2019. Ship detection from optical remote sensing images using multi-scale analysis and fourier HOG descriptor. Remote Sensing 11 (13), 1529. https://doi.org/10.3390/rs11131529.
Erfanian, Saeed, 2009. Tabataba Vakili, Vahid. Introducing excision switching-CFAR in K distributed sea clutter. Signal Processing 89 (6), 1023–1031. https://doi.org/10.1016/j.sigpro.2008.12.001.
Esteva, Andre, Kuprel, Brett, Novoa, Roberto A., Ko, Justin, Swetter, Susan M., Blau, Helen M., Thrun, Sebastian, 2017. Dermatologist-level classification of skin cancer with deep neural networks. Nature 542, 115–118. https://doi.org/10.1038/nature21056.
Everingham, Mark, Eslami, S.M.Ali, Van Gool, Luc, Williams, Christopher K.I., Winn, John, Zisserman, Andrew, 2014. The Pascal Visual Object Classes Challenge: A Retrospective. International Journal of Computer Vision 111 (1), 98–136. https://doi.org/10.1007/s11263-014-0733-5.
Gan, Lu., Liu, Peng, Wang, Lizhe, 2016. Rotation Sliding Window of the HOG Feature in Remote Sensing Images for Ship Detection. Proceedings of International Symposium on Computational Intelligence and Design (**ISCID**) 1, 401–404. https://doi.org/10.1109/ISCID.2015.248.
Gao, Gui, Gao, Sheng, He, Juan, Li, Gaosheng, 2018. Ship detection using compact polarimetric SAR based on the notch filter. IEEE Transactions on Geoscience and Remote Sensing 56 (9), 5380–5393. https://doi.org/10.1109/TGRS.2018.2815582.
Gao, Fei, Shi, Wei, Wang, Jun, Yang, Erfu, Zhou, Huiyu, 2019. Enhanced Feature Extraction for Ship Detection from Multi-Resolution and Multi-Scene Synthetic Aperture Radar (SAR) Images. Remote Sensing 11 (22), 2694. https://doi.org/10.3390/rs11222694.
Gidaris, Spyros, Komodakis, Nikos, 2015. Object detection via a multi-region and semantic segmentation-aware CNN model. In: Proceedings of IEEE International Conference on Computer Vision (**ICCV**), pp. 1134–1142. https://doi.org/10.1109/ICCV.2015.135.
Girshick, Ross. Fast R-CNN. arXiv preprint, arXiv:1504.08083. https://arxiv.org/abs/1504.08083.
Girshick, Ross; Donahue, Jeff; Darrell, Trevor; Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 580-587, Sep., 2014. https://doi.org/10.1109/CVPR.2014.81.
Goodfellow, Ian J.; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Yoshua. Generative Adversarial Networks. arXiv preprint, arXiv:1406.2661. https://arxiv.org/abs/1406.2661.
Gui, Gao, 2011. A parzen-window-kernel-based CFAR algorithm for ship detection in SAR images. IEEE Geoscience and Remote Sensing Letters 8 (3), 557–561. https://doi.org/10.1109/LGRS.2010.2090492.
Gui, Yunchuan, Li, Xiuhe, Xue, Lei, 2019. A multilayer fusion light-head detector for SAR ship detection. Sensors 19 (5), 1124. https://doi.org/10.3390/s19051124.
Guida, Maurizio, Longo, Maurizio, Lops, Marco, 1993. Biparametric CFAR procedures for lognormal clutter. IEEE Transactions on Aerospace and Electronic Systems 29 (3), 798–808. https://doi.org/10.1109/10.1109/7.220931.
He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian. Deep residual learning for image recognition. arXiv preprint, arXiv:1512.03385. https://arxiv.org/abs/1512.03385.
He, Kaiming; Girshick, Ross; Dollar, Piotr. Rethinking ImageNet Pre-training. arXiv preprint, arXiv: 1811.08883. https://arxiv.org/abs/1811.08883.
He, Kaiming, Gkioxari, Georgia, Dollár, Piotr, Girshick, Ross, Mask, R.-C.N.N., 2020. IEEE Transactions on Pattern Analysis and Machine Intelligence 42 (2), 386–397. https://doi.org/10.1109/TPAMI.2018.2844175.
Hoang, Van-Dung, Le, My-Ha, Jo, Kang-Hyun, 2014. Hybrid cascade boosting machine using variant scale blocks based HOG features for pedestrian detection. Neurocomputing 135, 357–366. https://doi.org/10.1016/j.neucom.2013.12.017.
Hong, Sanghoon; Roh, Byungseok; Kim, Kye-Hyeon; Cheon, Yeongjae; Park, Minje. PVANet: Lightweight Deep Neural Networks for Real-time Object Detection. arXiv preprint, arXiv:1611.08588. https://arxiv.org/abs/1611.08588.
Hosang, Jan; Benenson, Rodrigo; Schiele, Bernt. Learning non-maximum suppression. arXiv preprint, arXiv:1705.02950. https://arxiv.org/abs/1705.02950.
Howard, Andrew G.; Zhu, Menglong; Chen, Bo; Kalenichenko, Dmitry; Wang, Weijun; Weyand, Tobias; Andreetto, Marco; Adam, Hartwig. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint, arXiv: 1704.04861. https://arxiv.org/abs/1704.04861.
Hu, Jie; Shen, Li; Sun, Gang. Squeeze-and-Excitation Networks. arXiv preprint, arXiv:1709.01507. https://arxiv.org/abs/1709.01507.
Huang, Gao; Liu, Zhuang; Maaten, Laurens van der; Weinberger, Kilian Q. Densely connected convolutional networks. arXiv preprint, arXiv:1608.06993. https://arxiv.org/abs/1608.06993.
Huang, Zhongling; Dumitru, Corneliu Octavian; Pan, Zongxu; Lei, Bin; Datcu, Mihai. Can a Deep Network Understand the Land Cover Across Sensors? Proceedings of IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 9847-9850, Yokohama, Japan, 2019, https://doi.org/10.1109/IGARSS.2019.8899080.
Huang, Zhongling; Dumitru, Corneliu Octavian; Pan, Zongxu; Lei, Bin; Datcu, Mihai, 2020. Classification of Large-Scale High-Resolution SAR Images With Deep Transfer Learning. IEEE Geoscience and Remote Sensing Letters [Online]. https://doi.org/10.1109/LGRS.2020.2965558.
Huang, Lanqing, Liu, Bin, Li, Boying, Guo, Weiwei, Yu, Wenhao, Zhang, Zenghui, Yu, Wenxian, 2018. OpenSARShip: A Dataset Dedicated to Sentinel-1 Ship Interpretation. IEEE J. Sel. Top. Appl. Earth Observations Remote Sensing 11 (1), 195–208. https://doi.org/10.1109/JSTARS.2017.2755672.
Huang, Zhongling, Pan, Zongxu, Lei, Bin, 2017. Transfer Learning with Deep Convolutional Neural Network for SAR Target Classification with Limited Labeled Data. Remote Sensing 9 (9), 907. https://doi.org/10.3390/rs9090907.
Huang, Zhongling, Datcu, Mihai, Pan, Zongxu, Lei, Bin, 2020. Deep SAR-Net: Learning objects from signals. ISPRS Journal of Photogrammetry and Remote Sensing 161, 179–193. https://doi.org/10.1016/j.isprsjprs.2020.01.016.
Huang, Zhongling, Pan, Zongxu, Lei, Bin, 2020. Where, What, and How to Transfer in SAR Target Recognition Based on Deep CNNs. IEEE Transactions on Geoscience and Remote Sensing 58 (4), 2324–2336. https://doi.org/10.1109/TGRS.2019.2947634.
Hubel, D.H., Wiesel, T.N., 1959. Receptive fields of single neurones in the cat's striate cortex. J. Physiol. 148 (3), 574–591. https://doi.org/10.1113/jphysiol.1959.sp006308.
Iandola, Forrest N.; Han, Song; Moskewicz, Matthew W.; Ashraf, Khalid; Dally, William J.; Keutzer, Kurt. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv preprint, arXiv:1602.07360. https://arxiv.org/abs/1602.07360.
Ioffe, Sergey; Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint, arXiv:1502.03167. https://arxiv.org/abs/1502.03167.
Jiang, Shaofeng, Wang, Chao, Zhang, Bo, Zhang, Hong, 2012. Ship detection based on feature confidence for high resolution SAR images. In: Proceedings of International Geoscience and Remote Sensing Symposium (**IGARSS**), pp. 6844–6847. https://doi.org/10.1109/IGARSS.2012.6352591.
Jiao, Jiao, Zhang, Yue, Sun, Hao, Yang, Xue, Gao, Xun, Hong, Wen, Fu, Kun, Sun, Xian, 2018. A Densely Connected End-to-End Neural Network for Multiscale and Multiscene SAR Ship Detection. IEEE Access 6, 20881–20892. https://doi.org/10.1109/ACCESS.2018.2825376.
Kang, Miao, Ji, Kefeng, Leng, Xiangguang, Lin, Zhao, 2017. Contextual Region-Based Convolutional Neural Network with Multilayer Fusion for SAR Ship Detection. Remote Sensing 9 (8), 860. https://doi.org/10.3390/rs9080860.
Kanjir, U., Greidanus, H., 2018. Oštir, Krištof. Vessel detection and classification from

spaceborne optical images: A literature survey. Remote Sensing of Environment 207, 1–26. https://doi.org/10.1016/j.rse.2017.12.033.
Keras. Available online: https://keras.io/ (accessed on 10 Nov., 2019).
Kingma, Diederik P.; Ba, Jimmy Lei. Adam: A method for stochastic optimization. arXiv preprint, arXiv: 1412.6980. https://arxiv.org/abs/1412.6980v8.
Koushik, Jayanth. Understanding Convolutional Neural Networks. arXiv preprint, arXiv:1605.09081. https://arxiv.org/abs/1605.09081.
Koyama, C.N., Gokon, H., Jimbo, M., Koshimura, S., Sato, M., 2016. Disaster debris estimation using high-resolution polarimetric stereo-SAR. ISPRS Journal of Photogrammetry and Remote Sensing 120, 84–98. https://doi.org/10.1016/j.isprsjprs.2016.08.003.
Krizhevsky, Alex, Sutskever, Ilya, Hinton, Geoffrey E., 2017. ImageNet classification with deep convolutional neural networks. Communications of the ACM 60 (6), 84–90. https://doi.org/10.1145/3065386.
LabelImg. Available online: https://github.com/tzutalin/labelImg. (accessed on 10 Nov., 2019).
LeCun, Yann, Bengio, Yoshua, Hinton, Geoffrey, 2015. Deep learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539.
Lecun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86 (11), 2278–2324. https://doi.org/10.1109/5.726791.
Lee, Hyungtae, Kwon, Heesung, 2017. Going Deeper With Contextual CNN for Hyperspectral Image Classification. IEEE Trans. on Image Process. 26 (10), 4843–4855. https://doi.org/10.1109/TIP.2017.2725580.
Li, Jianwei; Qu, Changwen; Shao, Jiaqi. Ship detection in SAR images based on an improved faster R-CNN. Proceedings of SAR in Big Data Era: Models, Methods and Applications (BIGSARDATA), pp. 1-6, Beijing, 2017. https://doi.org/10.1109/BIGSARDATA.2017.8124934.
Li, Jianwei; Qu, Changwen; Peng, Shujuan. A ship detection method based on Cascade CNN in SAR images. Control and Decision, vol. 34, no. 10, pp. 2191-2197, Oct., 2019. https://doi.org/10.13195/j.kzyjc.2018.0168.
Li, J., Qu, C., Peng, S., Jiang, Y., 2019. Ship Detection in SAR images Based on Generative Adversarial Network and Online Hard Examples Mining. Journal of Electronics and Information Technology 41 (1), 143–149. https://doi.org/10.11999/JEIT180050.
Li, J., Qu, C., Peng, S., 2019. A Joint SAR Ship Detection and Azimuth Estimation Method. Geomatics and Information Science of Wuhan University 44 (6), 901–907. https://doi.org/10.13203/j.whugis20170328.
Lin, Tsung-Yi, Dollár, Piotr, Girshick, Ross, He, Kaiming, Hariharan, Bharath, Belongie, Serge, 2017. Feature pyramid networks for object detection. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (**CVPR**), pp. 936–944. https://doi.org/10.1109/CVPR.2017.106.
Lin, Tsung-Yi; Goyal, Priya; Girshick, Ross; He, Kaiming; Dollar, Piotr. Focal loss for dense object detection. arXiv preprint, arXiv:1708.02002. https://arxiv.org/abs/1708.02002.
Lin, Min; Chen, Qiang; Yan, Shuicheng. Network In Network. arXiv preprint, arXiv:1312.4400. https://arxiv.org/abs/1312.4400.
Lin, Zhao, Ji, Kefeng, Leng, Xiangguang, Kuang, Gangyao, 2019. Squeeze and Excitation Rank Faster R-CNN for Ship Detection in SAR Images. IEEE Geoscience and Remote Sensing Letters 16 (5), 751–755. https://doi.org/10.1109/LGRS.2018.2882551.
Lin, Huiping, Song, Shengli, Yang, Jian, 2018. Ship classification based on MSHOG feature and task-driven dictionary learning with structured incoherent constraints in SAR images. Remote Sensing 10 (2), 190. https://doi.org/10.3390/rs10020190.
Liu, Nengyuan, Cao, Zongjie, Cui, Zongyong, Pi, Yiming, Dang, Sihang, 2019. Multi-scale proposal generation for ship detection in SAR images. Remote Sensing 11 (5), 526. https://doi.org/10.3390/rs11050526.
Liu, Wei; Anguelov, Dragomir; Erhan, Dumitru; Szegedy, Christian; Reed, Scott; Fu, Cheng-Yang; Berg, Alexander C. SSD: Single shot multibox detector. arXiv preprint, arXiv:1512.02325. https://arxiv.org/abs/1512.02325.
Liu, Jieyu, Zhao, Tong, Liu, Min, 2020. Ship Target Detection in SAR Image Based on RetinaNet. Journal of Hunan University Natural Sciences 47 (2), 85–91. https://doi.org/10.16339/j.cnki.hdxbzkb.2020.02.012.
Long, Jonathan; Shelhamer, Evan; Darrell, Trevor. Fully Convolutional Networks for Semantic Segmentation. arXiv preprint, arXiv:1411.4038. https://arxiv.org/abs/1411.4038.
Lowe, David G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60 (2), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94.
Mao, Yuxing, Yang, Yuqin, Ma, Ziyuan, Li, Mingzhe, Su, Hao, Zhang, Jun, 2020. Efficient Low-Cost Ship Detection for SAR Imagery Based on Simplified U-Net. IEEE Access 8, 69742–69753. https://doi.org/10.1109/ACCESS.2020.2985637.
Mateo-García, Gonzalo, Laparra, Valero, López-Puigdollers, Dan, Gómez-Chova, Luis, 2020. Transferring deep learning models for cloud detection between Landsat-8 and Proba-V. ISPRS Journal of Photogrammetry and Remote Sensing 160, 1–17. https://doi.org/10.1016/j.isprsjprs.2019.11.024.
Meyer, F., Hinz, S., Laika, A., Weihing, D., Bamler, R., 2006. Performance analysis of the TerraSAR-X Traffic monitoring concept. ISPRS Journal of Photogrammetry and Remote Sensing 61 (3–4), 225–242. https://doi.org/10.1016/j.isprsjprs.2006.08.002.
Mita, T., Kaneko, T., Hori, O., 2005. Joint Haar-like features for face detection. Proceedings of IEEE International Conference on Computer Vision (**ICCV**) 2, 1619–1626. https://doi.org/10.1109/ICCV.2005.129.
Nunziata, F., Gambardella, A., Migliaccio, M., 2013. On the degree of polarization for SAR sea oil slick observation. ISPRS Journal of Photogrammetry and Remote Sensing 78, 41–49. https://doi.org/10.1016/j.isprsjprs.2012.12.007.
OpenCV. Available online: https://opencv.org/. (accessed on 6 Sep., 2019).
OpenSAR. Available online: http://opensar.sjtu.edu.cn. (accessed on 6 Sep., 2019).
Osco, Lucas Prado, Arruda, Mauro dos Santos de, Marcato Junior, José, da Silva, Neemias Buceli, Ramos, Ana Paula Marques, Moryia, Érika Akemi Saito, Imai, Nilton Nobuhiro, Pereira, Danillo Roberto, Creste, José Eduardo, Matsubara, Edson Takashi, Li, Jonathan, Gonçalves, Wesley Nunes, 2020. A convolutional neural network approach for counting and geolocating citrus-trees in UAV multispectral imagery. ISPRS Journal of Photogrammetry and Remote Sensing 160, 97–106. https://doi.org/10.1016/j.isprsjprs.2019.12.010.
Petit, M., Stretta, J.-M., Farrugio, H., Wadsworth, A., 1992. Synthetic aperture radar imaging of sea surface life and fishing activities. IEEE Transactions on Geoscience and Remote Sensing 30 (5), 1085–1089. https://doi.org/10.1109/36.175346.
Redmon, Joseph; Farhadi, Ali. YOLO9000: Better, faster, stronger. arXiv preprint, arXiv:1612.08242. https://arxiv.org/abs/1612.08242.
Redmon, Joseph; Farhadi, Ali. YOLOv3: an incremental improvement. arXiv preprint, arXiv: 1804.02767. https://arxiv.org/abs/1804.02767.
Redmon, Joseph; Divvala, Santosh; Girshick, Ross; Farhadi, Ali. You only look once: Unified, real-time object detection. arXiv preprint, arXiv:1506.02640. https://arxiv.org/abs/1506.02640.
Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. 39 (6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.
Rezatofighi, Hamid; Tsoi, Nathan; Gwak, Junyoung; Sadeghian, Amir; Reid, Ian; Savarese, Silvio. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. arXiv preprint, arXiv:1902.09630. https://arxiv.org/abs/1902.09630.
Ribani, Ricardo; Marengoni, Mauricio. A Survey of Transfer Learning for Convolutional Neural Networks. Proceedings of SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T), pp. 47-57, Brazil, 2019. https://doi.org/10.1109/SIBGRAPI-T.2019.00010.
Ronneberger, Olaf; Fischer, Philipp; Brox, Thomas. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv preprint, arXiv:1505.04597. https://arxiv.org/abs/1505.04597.
Russakovsky, Olga; Deng, Jia; Su, Hao; Krause, Jonathan; Satheesh, Sanjeev; Ma, Sean; Huang, Zhiheng; Karpathy, Andrej; Khosla, Aditya; Bernstein, Michael; Berg, Alexander C.; Li, Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. arXiv preprint, arXiv:1409.0575. https://arxiv.org/abs/1409.0575.
Schilling, Hendrik, Bulatov, Dimitri, Middelmann, Wolfgang, 2018. Object-based detection of vehicles using combined optical and elevation data. ISPRS Journal of Photogrammetry and Remote Sensing 136, 85–105. https://doi.org/10.1016/j.isprsjprs.2017.11.023.
Schwegmann, C.P., Kleynhans, W., Salmon, B.P., 2017. Synthetic Aperture Radar Ship Detection Using Haar-Like Features. IEEE Geoscience and Remote Sensing Letters 14 (2), 154–158. https://doi.org/10.1109/LGRS.2016.2631638.
Shen, Yi-Kang; Chiu, Ching-Te. Local binary pattern orientation based face recognition. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1091-1095, Aug., 2015. https://doi.org/10.1109/ICASSP.2015.7178138.
Shrivastava, Abhinav; Gupta, Abhinav; Girshick, Ross. Training Region-based Object Detectors with Online Hard Example Mining. arXiv preprint, arXiv:1604.03540. https://arxiv.org/abs/1604.03540.
Sifre, Laurent. Rigid-motion scattering for image classification. Ecole Polytechnique, CMAP, Ph. D. thesis, 2014. Also available online: https://www.di.ens.fr/data/publications/papers/phd_sifre.pdf. (accessed on 8 May, 2020).
Simonyan, Karen; Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. arXiv preprint, arXiv:1409.1556. https://arxiv.org/abs/1409.1556.
Song, Zhina, Sui, Haigang, Wang, Yujie, 2014. Automatic ship detection for optical satellite images based on visual attention model and LBP. Proceedings of IEEE Workshop on Electronics, Computer and Applications IWECA, 722–725. https://doi.org/10.1109/IWECA.2014.6845723.
Song, Shengli, Xu, Bin, Yang, Jian, 2016. SAR target recognition via supervised discriminative dictionary learning and sparse representation of the SAR-HOG feature. Remote Sensing 8 (8), 683. https://doi.org/10.3390/rs8080683.
Sun, Ke; Zhao, Yang; Jiang Borui; Cheng, Tianheng; Xiao, Bin; Liu, Dong; Mu, Yadong; Wang, Xinggang; Liu, Wenyu; Wang, Jingdong. High-Resolution Representations for Labeling Pixels and Regions. arXiv preprint, arXiv:1904.04514. https://arxiv.org/abs/1904.04514.
Szegedy, Christian; Liu, Wei; Jia, Yangqing; Sermanet, Pierre; Reed, Scott; Anguelov, Dragomir; Erhan, Dumitru; Vanhoucke, Vincent; Rabinovich, Andrew. Going Deeper with Convolutions. arXiv preprint, arXiv:1409.4842. https://arxiv.org/abs/1409.4842.
Szegedy, Christian; Vanhoucke, Vincent; Ioffe, Sergey; Shlens, Jon; Wojna, Zbigniew. Rethinking the Inception Architecture for Computer Vision. arXiv preprint, arXiv:1512.00567. https://arxiv.org/abs/1512.00567.
Tan, Mingxing; Le, Quoc V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv preprint, arXiv:1905.11946. https://arxiv.org/abs/1905.11946?context=stat.ML.
Tello, Mariví, López-Martínez, Carlos, Mallorqui, Jordi J., 2006. Automatic vessel monitoring with single and multidimensional SAR images in the wavelet domain. ISPRS Journal of Photogrammetry and Remote Sensing 61 (3–4), 260–278. https://doi.org/10.1016/j.isprsjprs.2006.09.012.
Tensorflow. Available online: https://www.tensorflow.org/ (accessed on 10 Nov., 2019).
Uijlings, J.R.R., Van De Sande, K.E.A., Gevers, T., Smeulders, A.W.M., 2013. Selective search for object recognition. International Journal of Computer Vision 104 (2), 154–171. https://doi.org/10.1007/s11263-013-0620-5.
Wang, Chonglei, Bi, Funkun, Chen, Liang, Chen, Jing, 2016. A novel threshold template algorithm for ship detection in high-resolution SAR images. In: Proceedings of International Geoscience and Remote Sensing Symposium (**IGARSS**), pp. 100–103.

https://doi.org/10.1109/IGARSS.2016.7729016.
Wang, Jizhou, Lu, Changhua, Jiang, Weiwei, 2018. Simultaneous ship detection and orientation estimation in SAR images based on attention module and angle regression. Sensors 18 (9), 2851. https://doi.org/10.3390/s18092851.
Wang, Shigang, Wang, Min, Yang, Shuyuan, Jiao, Licheng, 2017. New Hierarchical Saliency Filtering for Fast Ship Detection in High-Resolution SAR Images. IEEE Trans. Geosci. Remote Sensing 55 (1), 351–362. https://doi.org/10.1109/TGRS.2016.2606481.
Wang, Yuanyuan, Wang, Chao, Zhang, Hong, 2018. Combining a single shot multibox detector with transfer learning for ship detection using Sentinel-1 SAR images. Remote Sensing Letters 9 (8), 780–788. https://doi.org/10.1080/2150704X.2018.1475770.
Wang, Yuanyuan, Wang, Chao, Zhang, Hong, Dong, Yingbo, Wei, Sisi, 2019. Automatic Ship Detection Based on RetinaNet Using Multi-Resolution Gaofen-3 Imagery. Remote Sensing 11 (5), 531. https://doi.org/10.3390/rs11050531.
Wang, Yuanyuan, Wang, Chao, Zhang, Hong, Dong, Yingbo, Wei, Sisi, 2019. A SAR Dataset of Ship Detection for Deep Learning under Complex Backgrounds. Remote Sensing 11 (7), 765. https://doi.org/10.3390/rs11070765.
Wei, Shunjun, Su, Hao, Ming, Jing, Wang, Chen, Yan, Min, Kumar, Durga, Shi, Jun, Zhang, Xiaoling, 2020. Precise and Robust Ship Detection for High-Resolution SAR Imagery Based on HR-SDNet. Remote Sensing 12 (1), 167. https://doi.org/10.3390/rs12010167.
Woo, Sanghyun; Park, Jongchan; Lee, Joon-Young; Kweon, In So. CBAM: Convolutional block attention module. arXiv preprint, arXiv:1807.06521. https://arxiv.org/abs/1807.06521.
Yang, Long, Su, Juan, Li, Xiang, 2019. Ship detection in SAR images based on deep convolutional neural network. Systems Engineering and Electronics 41 (9), 1990–1997. https://doi.org/10.3969/j.issn.1001-506X.2019.09.11.
Yang, Long, Su, Juan, Huang, Hu.a., Li, Xiang, 2020. SAR Ship Detectin Based on Convolutional Neural Network with Deep Multiscale Feature Fusion. Acta Optica Sinica 40 (2), 0215002. https://doi.org/10.3788/AOS202040.0215002.
Yang, Feng, Xu, Qizhi, Li, Bo, 2017. Ship Detection From Optical Satellite Images Based on Saliency Segmentation and Structure-LBP Feature. IEEE Geoscience and Remote Sensing Letters 14 (5), 602–606. https://doi.org/10.1109/LGRS.2017.2664118.
Ye, Famao, Luo, Wei, Dong, Meng, He, Hailin, Min, Weidong, 2019. SAR Image Retrieval Based on Unsupervised Domain Adaptation and Clustering. **IEEE Geoscience and Remote Sensing Letters** 16 (9), 1482–1486. https://doi.org/10.1109/LGRS.2019.2896948.
Yin, Kuiying ; Jin, Lin ; Zhang ,Changchun ; Jiang, Jin. SAR Automatic Target Recognition Based on Shadow Contour. Proceedings of International Conference on Digital Manufacturing & Automation (ICDMA), pp. 1179-1183, Qingdao, 2013. https://doi.org/10.1109/ICDMA.2013.279.
Yin, Kui-Ying, Jin, Lin, Liu, Hong-Wei, Wang, Ying-Hua, 2012. SAR variant target automatic recognition algorithm based on local texture characteristic. Journal of Jilin University 42 (3), 743–748. https://doi.org/10.1109/CGC.2012.42.
Yu, Fisher; Koltun, Vladlen. Multi-Scale Context Aggregation by Dilated Convolutions. arXiv preprint, arXiv: 1511.07122. https://arxiv.org/abs/1511.07122.
Zeiler, Matthew D.; Fergus, Rob. Visualizing and understanding convolutional networks. arXiv preprint, https://arxiv.org/abs/1311.2901.
Zhang, Tao, Jiang, Linfeng, Xiang, Deliang, Ban, Yifang, Pei, Ling, Xiong, Huilin, 2019. Ship detection from PolSAR imagery using the ambiguity removal polarimetric notch filter. ISPRS Journal of Photogrammetry and Remote Sensing 157, 41–58. https://doi.org/10.1016/j.isprsjprs.2019.08.009.
Zhang, Tao, Ji, Jinsheng, Li, Xiaofeng, Yu, Wenxian, Xiong, Huilin, 2019. Ship Detection From PolSAR Imagery Using the Complete Polarimetric Covariance Difference Matrix. IEEE Transactions on Geoscience and Remote Sensing 57 (5), 2824–2839. https://doi.org/10.1109/TGRS.2018.2877821.
Zhang, Xiaohan, Wang, Haipeng, Xu, Congan, Lv, Yafei, Fu, Chunlong, Xiao, Huachao, He, You, 2019. A Lightweight Feature Optimizing Network for Ship Detection in SAR Image. IEEE Access 7, 141662–141678. https://doi.org/10.1109/ACCESS.2019.2943241.
Zhang, T., Zhang, X., 2019. High-speed ship detection in SAR images based on a grid convolutional neural network. Remote Sensing 11 (10), 1206. https://doi.org/10.3390/rs11101206.
Zhang, Tianwen, Zhang, Xiaoling, Shi, Jun, Wei, Shunjun, 2019. Depthwise Separable Convolution Neural Network for High-Speed SAR Ship Detection. Remote Sensing 11 (21), 2483. https://doi.org/10.3390/rs11212483.
Zhang, Xiaoling, Zhang, Tianwen, Shi, Jun, Wei, Shunjun, 2019. High-speed and high-accurate SAR ship detection based on a depthwise separable convolution neural network. Journal of Radars 8 (6), 841–851. https://doi.org/10.12000/JR19111.
Zhang, Tianwen; Zhang, Xiaoling. ShipDeNet-20: An Only 20 Convolution Layers and <1 MB Light-Weight SAR Ship Detector. IEEE Geoscience and Remote Sensing Letters [Online], Early Access, 2020. https://doi.org/10.1109/LGRS.2020.2993899.
Zhao, Juanping, Zhang, Zenghui, Yu, Wenxian, Truong, Trieu-Kien, 2018. A Cascade Coupled Convolutional Neural Network Guided Visual Attention Method for Ship Detection from SAR Images. IEEE Access 6, 50693–50708. https://doi.org/10.1109/ACCESS.2018.2869289.
Zhao, Juanping, Guo, Weiwei, Zhang, Zenghui, Yu, Wenxian, 2019. A coupled convolutional neural network for small and densely clustered ship detection in SAR images. Science China Information Sciences 62 (4), 4. https://doi.org/10.1007/s11432-017-9405-6.
Zhao, Zhong-Qiu, Zheng, Peng, Xu, Shou-Tao, Wu, Xindong, Nov. 2019. Object Detection With Deep Learning: A Review. IEEE Transactions on Neural Networks and Learning Systems 30 (1), 3212–3232. https://doi.org/10.1109/TNNLS.2018.2876865.
Zhou, Deyun; Zeng Lina; Zhang Kun. A novel SAR target detection algorithm via multi-scale SIFT features. Journal of Northwestern Polytechnical University, vol. 33, no. 5, pp. 867-873, Oct., 2015. https://doi.org/10.3969/j.issn.1001-2400.2016.02.016.
Zhu, Jiwei, Qiu, Xiaolan, Pan, Zongxu, Zhang, Yueting, Lei, Bin, 2017. Projection Shape Template-Based Ship Target Recognition in TerraSAR-X Images. IEEE Geoscience and Remote Sensing Letters 14 (2), 222–226. https://doi.org/10.1109/LGRS.2016.2635699.