Recursive Hybrid Fusion Pyramid Network for Real-Time Small Object Detection on Embedded Devices

Ping-Yang Chen[1], Jun-Wei Hsieh[2], Chien-Yao Wang[3], and Hong-Yuan Mark Liao[3]

[1]Department of Computer Science, National Chiao Tung University, Taiwan
[2]College of Artificial Intelligence and Green Energy, National Chiao Tung University, Taiwan
[3]Institute of Information Science, Academia Sinica, Taiwan
pingyang.cs08g@nctu.edu.tw, jwhsieh@nctu.edu.tw, kinyiu@iis.sinica.edu.tw,
and liao@iis.sinica.edu.tw

Abstract

This paper proposes a novel RHF-Net (Recursive Hybrid Fusion pyramid network) to solve the problem of small object detection on real-time embedded devices. Though the object detection accuracy rate is improved by a large margin with SoTA (State-of-The-Art) models, e.g., SSD, YOLO, RetinaNet, and RefineDet, they are still problematic for small object detection and inefficient on embedded systems. One novelty of the RHF-Net is a bidirectional fusion module) that allows to fuse feature maps with both the top-down and bottom-up directions to generate flexible FPs for small object detection. This module can be easily integrated to any feature pyramid based object detection model. Another novelty of this net is a recursive concatenation and reshaping module which can recursively concatenate not only high-level semantic features from deep layers but also reshape spatially richer features from shallower layers to prevent small objects from disappearing. RHF-Net net adopts computationally low-cost and feature preserving operations in the fusion, thus it is efficient and accurate even on embedded devices. The superiority of RHF-Net is investigated on the COCO benchmark and UAVDT dataset in terms of mAP and FPS.

1. Introduction

Since 2012, huge breakthroughs in object detection have been occurred one after another with the technology advancements on deep convolutional neural network (CNN) based models. Very first outstanding CNN model was AlexNet [1] which beat all the former SoTA classical machine learning models on ILSVRC challenge. Currently, there are two-stage (proposal driven) and one-stage (direct) object detection models in the literature; that is, the region-proposal based (two-stage) detection model such as R-CNN [2], faster R-CNN [3], SPP-Net [4], or Mask R-NN [5], and the regression based (one-stage) model such as SSD [8], RetinaNet [17], or YOLOv3 [7]. Generally, the former is known for its higher accuracy and the latter is known for its better efficiency. In the latest few years, accuracies of the one-stage model have been improved with a large margin by various state-of-the-art models like FPN [6], YOLOv3 [7], and SSD [7], which usually consist of deep feature extractors (backbones, *i.e.*, DarkNet-53 [7] and ResNet-101[10]), a feature pyramid (FP), and a classifier. Using a small or medium sized backbone can reduce the computational cost and thus increase the efficiency. However, a shallow backbone cannot generate rich semantic features for object detection. In order to increase the accuracy of object detection, the above SoTA methods usually adopt a deeper backbone (more than a hundred of layers) which will lead to a small object (<32×32 pixels in extent) in an input image become a single pixel at the final feature layer while extracting its features. Obviously, a single pixel is not sufficient for discriminating an object from its background. Generally speaking, deepening the network can enhance the accuracy but it leads to not only a higher computational cost, but also a lower detection rate on small objects. Thus, the above SoTA object detectors are problematic and unsuitable for small object detection.

To improve both the accuracy and efficiency, the detection model should avoid using too many feature-richness dithering operations such as convolutions, and computationally expensive operations, such as pooling and addition, to preserve as much as possible features for prediction. Lately, to improve the accuracy on small object detection, a feature pyramid (FP) structure is commonly adopted in the SoTA detectors due to its multi-scale structure. With this structure, abundant spatial information can be extracted from the last few layers of the network backbone. In general, FP is a multiple-layered pyramid structure that extracts spatial features from the last feature layer so that features can be fused with a top-down direction for detecting various scaled objects. There are few common types of FPs employed in object detection models, *i.e.*, pyramidal feature hierarchy (bottom-up), hourglass (bottom-up and top-down), SPP (spatial pyramid pooling), SPP + multi-scale fusion (which are adopted in SSD [7], FPN [6], SPP [4]), and PFPN [13], respectively. Hourglass FPs are generated by fusing last three layers of a backbone. On the other hand, SPP-based FPs [4][13] are generated from the last layer of a backbone. Thus, hourglass FPs contain richer multi-scaled features than SPP-based FPs,

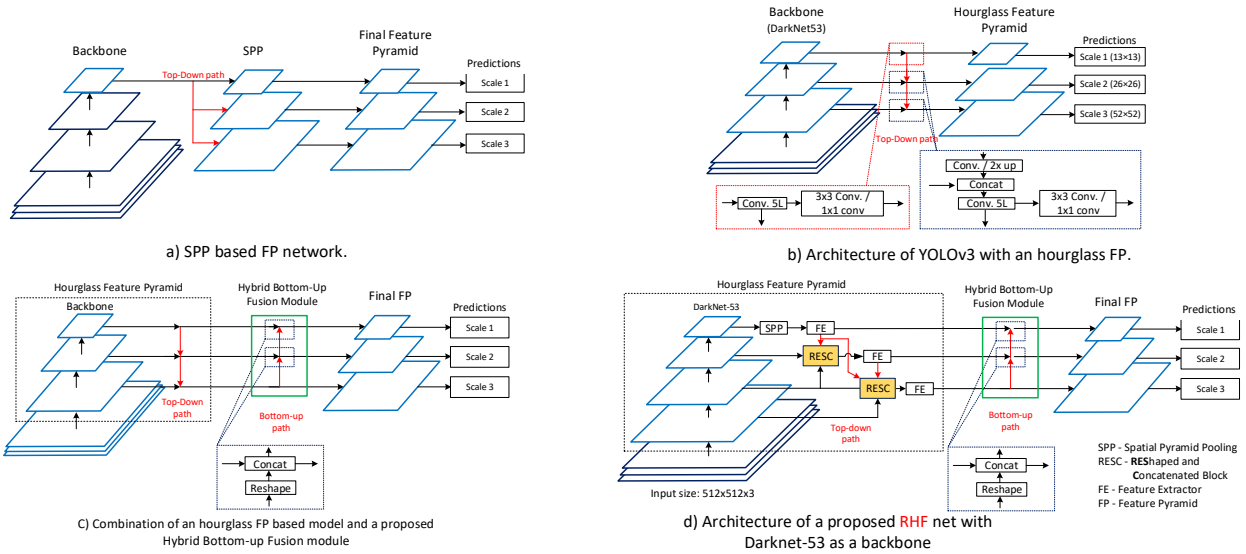

Figure 1. Architectures of a) SPP based FP network, b) YOLOv3 [7] with an hourglass feature pyramid, c) a model updated with a proposed hybrid bottom-up fusion module, and d) a proposed **RHF** model consists of a backbone (Darknet53 [7]), a three-scaled hourglass feature pyramid, and a final FP generated by the hybrid fusion module. Scale 1 adopts SPP (spatial pyramid pooling) and a feature extractor (FE), and Scale 2 and 3 comprises of **RESC** (**RES**haped and **C**oncatenated) block, and FE.

and this will lead to a higher accuracy in small object detection. However, the hourglass-based method adopts a top-down path to generate a three-scale FP for object prediction by summing features from the deeper layers to the shallower layers of the backbone. This one-directional path will prohibit the networks from detecting small objects. Therefore, its accuracy on small object detection will be problematic if small object features have already disappeared at the last layer of a backbone. This problem can be solved if contextual features of small objects from shallow layers can be recursively brought to other deeper layers for object prediction. To solve all above mentioned problems, we propose a novel deep detection model named **RHF** net (**R**ecursive **H**ybrid **F**usion pyramid network) to recursively fuse feature maps with both the top-down and bottom-up mechanism to generate flexible FPs for small object detection. First direction is a *top-down path* which forms an hourglass FP. The second direction is a *hybrid bottom-up path* that generates final FP layers by concatenating not only the reshaped features from a shallower layer but also features of the current layer of the hourglass FP. With this hybrid fusion mechanism, an object with very small size (even if it is down-sampled to 1 pixel) can re-appear at shallower layers. Another novelty of the proposal RHF net is to create a **RESC** (**RES**haped and **C**oncatenation) module which can be recursively executed to not only concatenate high-level semantic features from deep layers to shallower layers (top-down direction) but also re-shape spatially richer features of small objects at a shallower layer to a deeper layer (bottom-up direction). The RHF-net can perform better than SoTA FP-based methods due to its special design on concatenation and reshaping operations during the fusion process. Figure 1 shows the architecture of RHF net which is composed of DarkNet-53 as a backbone, an hourglass FP, and a final FP. DarkNet-53 is adopted due to its outstanding feature extraction ability and a low computational cost compared to very deep backbones, e.g., ResNet-101. The hourglass FP is generated by recursively performing the RESC block, SPP, and convolutional feature extractors (FEs) for the tradeoff between accuracy and efficiency. The hybrid architecture is superior to the SoTA FP-based methods, *e.g.*, YOLOv3[7], PFPNet, SSD[8], RefineDet, RetinaNet[17], MobileNet [42], PeLee [20], due to its special computationally low-cost and feature preserving operations. Superiority of the RHF net over the existing SoTA methods is proven for general object detection and small object detections on the *MS COCO* [45] and *UAVDT* [44] benchmark datasets. Moreover, in case of a light backbone is adopted, the proposed model will also perform the best among SoTA light-weight nets for real-life small object detection on embedded devices. Main contributions of this paper are summarized as follows:

- Superior performances in terms of accuracy and efficiency are achieved by the proposed RHF net;
- Hourglass FP based object detection models are improved by adding our hybrid fusion module;
- Small object features are enriched in deeper layers by fusing higher-semantic features from a shallower layer with RESC block and the hybrid fusion module.

Therefore, a higher accuracy is achieved for small object detection;

- Computation cost is decreased with a large margin by adopting concatenation operations instead of using addition and convolution;
- Feature richness is preserved by fusing features not only from a shallower layer but also a current layer with a reshaping operation instead of pooling and addition operations;

2. Related Works

2.1. One-stage object detectors

One-stage object detector consists of a backbone network (referred to backbone) and a predictor. The backbone is a stacked feature map that is pre-trained as a single image classifier on a very large dataset, *i.e*., ImageNet. In 2013, the first CNN-based one-stage object detector OverFeat [15] was developed using sliding-window paradigm. Then, two years later, YOLO [16] achieved the SoTA performance by integrating bounding box proposal and subsequent feature resampling as one stage. Moreover, YOLO divides an input image into 7×7 grids and simultaneously predicts bounding boxes and class confidences on each grid. Next, SSD [8] employed in-network multiple feature maps for detecting objects with varying shapes and sizes, and this feature makes SSD more robust than YOLO. YOLOv2 [9] achieved outstanding results in terms of accuracy and efficiency by proposing DarkNet-19 [9] as a backbone, and it includes several new aspects such as batch normalization, higher resolution classifier, anchor box prediction. For better detection of small objects, FPN is developed using a feature pyramid (FP) structure and it achieves a higher detection accuracy on small objects. Later, the SoTA YOLOv3 [7] was developed by adopting the concept of FPN. By changing the backbone from DarkNet-19 [9] to DarkNet-53 [7], YOLO v3 achieves the best performance. Similarly, RetinaNet [17], a combination of FPN and ResNet as a backbone, proposes the use of focal loss to significantly reduce false positives in one-stage detectors by dynamically adjusting the weights of each anchor box.

2.2. Latest one-stage object detectors

An hourglass feature pyramid (HFP) (see Figure 1(b, c, and d)), one of common types of FP, is adopted in many models, *e.g*., FPN, YOLOv3, RetinaNet, and RefineDet, to improve the detection rate of small objects. HFP is generated by fusing from a high-feature resolution in the deepest layer of a backbone to a low-feature resolution in the shallower scales with a top-down path and lateral connections, in which the shallower layers are expected to richen strong features of small objects. Consequently, HFP provides a higher accuracy on small object detection. However, most of these models have a high computational cost due to their very deep backbones and high-cost operations such as convolutions. Thus, the models are not applicable for real-time embedded applications. Moreover, fusing features of small objects along the top-down pathways is not possible if small objects are already disappeared or become 1 pixel.

2.3. Latest one-stage object detectors

Since object detection is one of the most popular fields in computer vision, several SoTA models have been developed in the past one year. RefineDet [19] employed an Encode-Decode structure for deepening the network and up-sampling deeper scale features to the shallower scales to enrich the contextual information for the final FP. Based on the VGG16 [1] backbone, it achieved AP_S of 16.3 and AP_{50} of 54.5 on input size 512, MS COCO [45] test-dev. A newly proposed PeLee [20] model, a variant of DenseNet [21], outperformed SSD+MobileNet by 6.53% on Stanford Dogs [22] dataset with its much shallower network. However, PeLee has a lower performance on MS COCO [45] dataset and lower accuracy on small object detection. PFPN [13] adopts VGGNet-16 as a backbone and SPP for generating a final FP from the last layer of a backbone which concatenates multi-scale features. The above mentioned model outperformed other methods on small object detection. CornerNet [24], with Hourglass-104 as a backbone, detects an object bounding box as a pair of key points via a corner pooling technique, and it outperformed the existing methods on multi-scale general object detection and small object detection categories. The latest SoTA one-stage object detector M2Det [29] outperformed all the existing methods on all multi-scale categories on MS COCO. However, all these models are not suitable for a real-time object detection task due to their high computational cost.

3. Method

Most of the SoTA methods, *i.e*., FPN, YOLOv3, RetinaNet, and RefineDet use an HFP to improve the detection rate of small objects. The HFP is often generated from a very deep backbone by fusing features along a top-down path. Consequently, HFP provides a higher accuracy on small object detection but makes these models time-consuming and not applicable for real-time embedded applications due to high-cost operations such as convolutions. Moreover, fusing features of small objects along the top-down pathways is not possible if small objects are already disappeared or become 1 pixel. To reduce the computation cost, this paper proposes a RESC (REShaped and Concatenation) block to recursively generate an hourglass

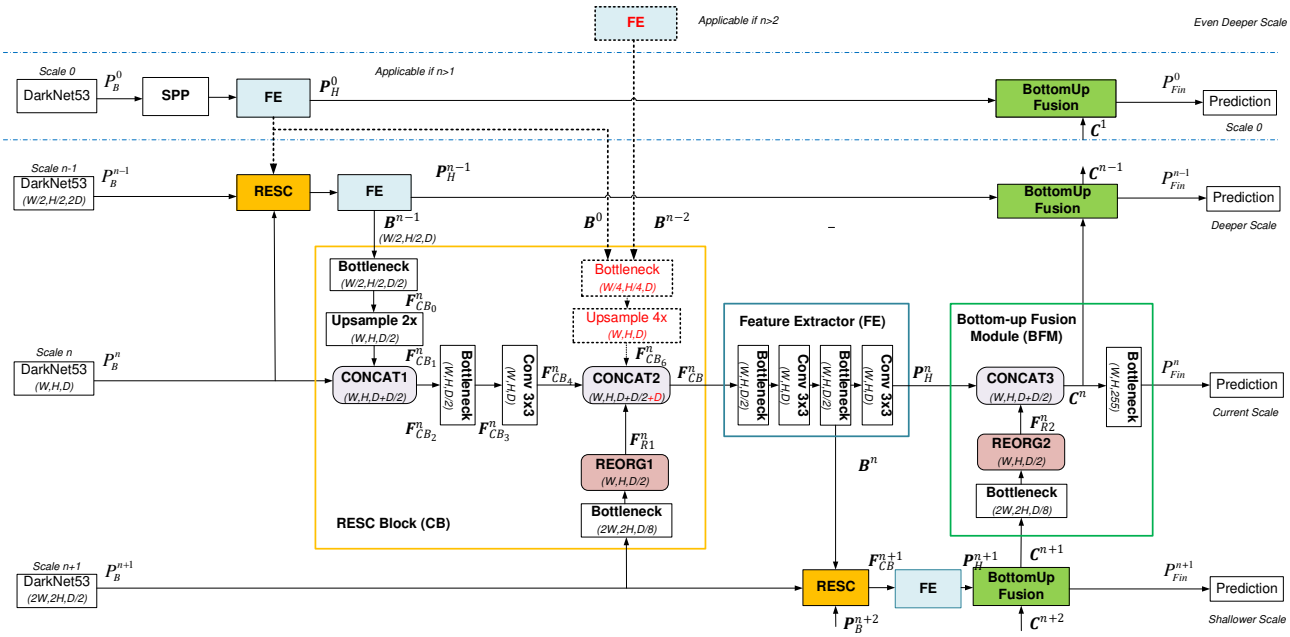

Figure 2. Final feature pyramid generating flowchart for Scale n consisting of RESC block and FE. However, for the Scale0, feature pyramid scale is generated with SPP and FE.

FP by merging features not only from the deeper layers but also from the shallower layer. In addition to the RESC block, a hybrid bottom-up fusion module is proposed to fuse the hourglass FP from a bottom-up path so that more discriminant features to identify small and medium objects can be fed into the final feature map for object prediction.

Figure 1(d) illustrates the basic architecture of our RHF net, a one-stage object detection model for real-time object detection, which consists of an hourglass FP and the hybrid fusion module. We denote the RHF-Net with a quadruple $\mathbb{M} = (\mathbb{B}, \mathbb{H}, \mathbb{F}, \mathbb{P})$, where $\mathbb{B}$ is a *backbone*, $\mathbb{H}$ is an *hourglass FP generating function*, $\mathbb{F}$ is a *final FP generating function*, and $\mathbb{P}$ is a *prediction function*. $\mathbb{B}$ could be any feature extracting backbone, *i.e.*, DarkNet-53 [7] or ResNet-101[10]. From $\mathbb{B}$, base features $\boldsymbol{P}_B = \{\boldsymbol{P}_B^1, \boldsymbol{P}_B^2, ..., \boldsymbol{P}_B^{n-1}\}$ can be formed for generating an hourglass FP, *i.e.*, $\boldsymbol{P}_H = \{\boldsymbol{P}_H^1, \boldsymbol{P}_H^2, ..., \boldsymbol{P}_H^{n-1}\}$ and the final FP, *i.e.*, $\boldsymbol{P}_{Fin} = \{\boldsymbol{P}_{Fin}^1, \boldsymbol{P}_{Fin}^2, ..., \boldsymbol{P}_{Fin}^{n-1}\}$, where n is the number of scales for FP. The function $\mathbb{H}$ takes $\boldsymbol{P}_B$ as input to generate $\boldsymbol{P}_H$ by recursively performing the **RESC** module (see Figure 2). Finally, $\mathbb{P}$ predicts the object class and bounding box from $\boldsymbol{P}_{Fin}$ by a logistic regression.

Figure 2 demonstrates a comprehensive flowchart to generate the nth scale of a final FP, where the RESC block and FE (feature extractor) together form $\boldsymbol{P}_H^n$, the nth scale of an hourglass FP, and then the hybrid fusion module forms $\boldsymbol{P}_{Fin}^n$, the nth scale of the final FP. The RESC block generates the final FP, $\boldsymbol{P}_{Fin}$, by recursively concatenating and reorganizing four different scales from $\boldsymbol{P}_B$. RESC block adopts concatenation for fusing features of a deeper layer to a current layer and reorganization operation for fusing features of a shallower layer to a current layer. Both operations have a very low computational cost and can preserve all the contextual information. Under those circumstances, the accuracy and efficiency are both improved. On the other hand, the zero-th scale is generated by SPP (spatial pyramid pooling) [4], FE, and the hybrid fusion module.

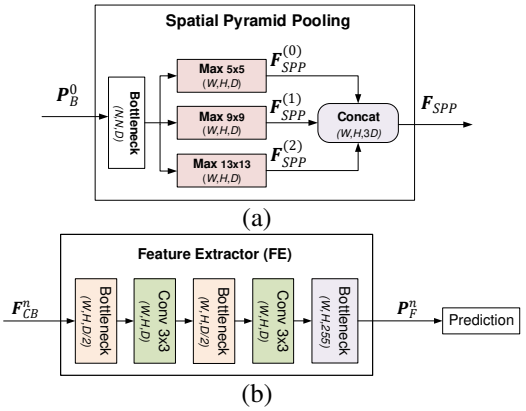

Figure 3. (a) Spatial pyramid pooling block. (b) FE block.

Figure 3(a) shows SPP that consists of a bottleneck layer, 3 max pooling layers with kernel sizes of (5×5), (9×9), and (13×13), and a concatenation. Number of feature channels

are reduced to half with the bottleneck layer, then three groups of max pooled feature maps with the same dimension, $\boldsymbol{F}_{SPP}^{(0)}$, $\boldsymbol{F}_{SPP}^{(1)}$, and $\boldsymbol{F}_{SPP}^{(2)}$, are generated. For concatenation purpose, all three max pooling operations employ a zero padding and a stride of 1 to create the same output sized feature maps. The concatenated max pooled features will become the output of SPP.

3.1. RESC Module for Feature Fusion

RESC block has two concatenation and one reshaping operations, *i.e*., CONCAT1, CONCAT2, and REORG1 [9]. Unlike concatenation methods in SoTA methods, the proposed RESC block recursively concatenates contextual features of not only adjacent layers but also even deeper (*n*-2)th layer. In other words, RESC block fuses features from 4 adjacent scales (shallow, current, deep, and deeper) of a backbone to richen the features for better detection. CONCAT1 concatenates current scale features of a backbone $\boldsymbol{P}_B^n$ and features $\boldsymbol{F}_{CB_1}^n$ that come from the output of bottleneck layer B^{n-1} after FE in the deeper scale. CONCAT2 is applied to concatenate $\boldsymbol{F}_{CB_4}^n$ of current scale, reshaped features $\boldsymbol{F}_{R_1}^n$ of a shallow scale $\boldsymbol{P}_B^{n+1}$, and 4× up-sampled features from even deeper scale. The output of RESC block $\boldsymbol{F}_{CB}^n$ can be formulated as:

$$\boldsymbol{F}_{CB}^n = \begin{cases} [\boldsymbol{F}_{CB_4}^{(n)}, \boldsymbol{F}_{R1}^{(n)}, \boldsymbol{F}_{CB_6}^{(n)}], n > 2 \in \mathbb{N} \\ [\boldsymbol{F}_{CB_4}^{(n)}, \boldsymbol{F}_{R1}^{(n)}], \quad \text{otherwise.} \end{cases} \tag{2}$$

To increase feature richness for deeper scales, REORG1 brings contextual features from a shallow layer to its deep layer without subsequent operations. First, the bottleneck layer uses 1×1 convolution to reduce the number of channels from D/2 to D/8. Then, REORG1 takes every single pixel of a feature map from the bottleneck block to generate 4 channels by separating adjacent pixels into different channels. If the size of feature maps were (2*W*, 2*H*, *D*/8), then the size of reshaped features is (*W*, *H*, *D*/2).

3.2. Feature Extractor

Figure 3(b) illustrates the flowchart of the FE block. It is put after the RESC block to extract more contextual and semantic features from fused features of 4 adjacent scales. The FE block consists of 2 consecutive parts of feature extraction where each part includes one bottleneck layer and a 3×3 convolutional layer. The former is employed to reduce the number of channels from *D* to *D*/2. The latter is employed to extract contextual features. Output of the second bottleneck layer is fed to RESC at the shallower scale for fusion.

3.3. Hybrid Bottom-up Fusion

FE can generate more contextual and semantic features. If the output of FE can be added into the final FP with a bottom-up direction, higher accuracy in object detection should be obtained. Thus, different from other SoTA methods which create prediction maps only from a top-down direction (see the red path in Figure 1(a)), this paper proposes a hybrid bottom-up fusion module to fuse the hourglass FP, i.e., $\boldsymbol{P}_H$ not only from a top-down direction but also from a bottom-up direction (see the green block in Figure 1(c) or (d)). The bottom-up fusion can bring contextual features from shallower scales to enrich the features in the final FP to predict and detect small and medium objects more accurately. The hybrid fusion module consists of CONCAT3, REORG2, and two bottleneck layers. CONCAT3 merges the outputs of FE and REORG2, i.e., $\boldsymbol{P}_H^n$ and $\boldsymbol{F}_{R2}^n$ (which is the reorganized feature map of $\boldsymbol{C}^{n-1}$ in the shallow scale) to generate $\boldsymbol{C}^n$. In other words, the output of CONCAT3 can be calculated as follows:

$$\boldsymbol{C}^n = [\boldsymbol{F}_{R2}^n, \boldsymbol{P}_H^n]. \tag{3}$$

The last bottleneck layer is used for fixing the number of channels to *L*, *L*=255 for our RHF net. Finally, the last layer outputs the final FP, i.e., $\boldsymbol{P}_{Fin}^n$ at the scale *n*.

3.4. Backbone

For the trade-off between accuracy and efficiency, most SoTA object detectors adopt VGGNet-16 or ResNet-101 which have 16 and 101 layers, respectively. The former's accuracy is limited for its number of convolution layers and the latter is limited for inference speed but with high accuracy. However, we chose DarkNet-53 [7] as the backbone of RHF net due to its medium-sized architecture and excellent feature extraction capability. Input image size is 512×512 and sizes of the backbone feature pyramid scales are 16×16×512, 32×32×512, and 64×64×256, respectively. For edge computing, the RHF net can adopt a shallower backbone similar to Tiny15 from YOLOv3-tiny, or PeLee net [20].

4. Experimental Results

We have employed three datasets (CarFlow, UAVDT[44], and COCO[45]) to evaluate our model using a machine with NVIDIA Jetson TX2 and NVIDIA Titan X. CarFlow is an in-house dataset consisting of 6000 (train: 2500, val: 500, and test: 3000) images with a resolution 1920×1920 that were extracted from day/night videos captured by a fisheye camera installed at different intersections and tested for edge computing (NVIDIA Jetson TX2 which works on mode 0). Metric adopted for performance evaluation is Average Precision (AP). Inference time is represented as FPS (Frames per Second).

4.1. Accuracy Improvements by Hybird Fusion Module

Since the RHF net is developed for improving the accuracy of small object detection, we evaluate the effects of our RHF model with/without hybrid fusion module on object detection based on the CarFlow and UAVDT dataset[44]. Table 1 tabulates the ablation studies to show the advantage of this module. A light backbone, *i.e.*, Pelee is adopted here to test the computation load of this hybrid fusion module. Clearly, from Table 1, the accuracy of small object detection is most significantly improved if the hybrid fusion module is adopted. Table 2 shows the ablation study of our method with/without the hybrid fusion module based on the UAVDT dataset. Three backbones, *i.e.*, Darknet53[7], CSPdarknet53 [23], and VGG-16 [1] were compared in this table. Actually, when Darknet53 [7] was adopted as the backbone without the fusion module, the detector will be YOLOv3 [19] which is used as a baseline for performance comparisons. Table 1 and Table 2 tell us the computation load of this hybrid module is light and can be ignored even though different backbones are adopted. Both tables prove the generalization of our method to improve the accuracy of object detection across different backbones. An important phenomenon found from Table 1 and Table 2 is: the accuracy of a light backbone is improved with a larger margin than a deep one.

Table 1. Ablation study of Hybrid Fusion Module on CarFlow dataset.

Backbone	Hybrid fusion	FPS	APS	APM	APL
Pelee-SPP 416x416		15.9	34.49	16.67	12.01
	✓	15.6	43.93	21.19	12.15

Table 2. Ablation study of Hybrid Fusion Module on UAVDT dataset [44].

backbone	Hybrid fusion	AP	fps
Darknet53 [7]		59.39	28.9
	✓	62.18	28.5
CSPdarknet53 [23]		65.89	36.8
	✓	68.75	36.4
VGG-16 [1]		53.98	31.8
	✓	55.18	31.3

4.2. Improvements by RHF net (RESC Module + Hybrid bottom-up fusion module)

Another novelty of this work is the design of RESC module. With the hybrid fusion module and the RESC module, we proposed the RHF net for object detection especially in small object. Table 3 shows the ablation study of RESC and hybrid fusion module on the CarFlow dataset. Clearly, the fusion module improves the accuracy of object detection more than the RESC module. Table 4 shows the ablation study of RESC and the hybrid fusion module on the UAVDT dataset[44]. The fusion module spends less time and is more efficient that the RESC module. CSPdarknet53 [23] is also a light backbone. It is also true for the RESC module that the accuracy of a light backbone is improved with a larger margin than a deep one. From Table 3 and Table 4, the hybrid fusion model works better than the RESC module in terms of both accuracy and efficiency under different backbones. Important finding is that the proposed RHF net reduces the computational cost but also improves the accuracy by reducing convolutional layers and replacing addition by concatenation operation. The tables also prove the generalization of our RHF net to improve the accuracy of object detection under different backbones.

Table 3. Ablation study of RESC and Hybrid Fusion Module on the CarFlow dataset.

Backbone	RESC	Hybrid fusion	SPP	Bflops	FPS	Test-AP$_{50}$	Test-Night-AP$_{50}$
Pelee 416x416	✓			**4.732**	16.7	33.99	17.73
	✓	✓		5.219	16.1	36.67	19.85
		✓	✓	5.889	15.6	37.69	**20.93**
	✓	✓	✓	6.525	15.3	**38.59**	20.23

Table 4. Ablation study of RESC and Hybrid Fusion Module on the UAVDT dataset [44].

backbone	RESC	Hybrid fusion	AP	fps
Darknet53		✓	62.18	28.5
	✓		61.25	28.1
	✓	✓	66.18	27.8
CSPdarknet53		✓	68.75	36.4
	✓		67.61	35.2
	✓	✓	70.42	34.8
VGG-16		✓	55.18	31.3
	✓		56.14	30.8
	✓	✓	60.13	29.8

(a) YOLOv3_512x512

(b) RHF 512x512

Figure 4. Small Object detection results on MS COCO dataset.

Figure 4 shows the comparisons of object detection between YOLO v3 and RHF on one image (with input size 512×512) selected from MS COCO test-dev set. They models are a) YOLOv3, b) RHF net with the hybrid bottom-up fusion module. It is obvious that the last one receives the best detection results.

4.3. Learning rich information with light weight backbone

Table 5 tabulates the performances of RHF and other SoTA methods, which can run on Nvidia Jetson TX2 for real-time. The existing methods (YOLO3, SSD, RefineDet) cannot perform real-time on TX2 embedded device since their FPS < 3. We select the Pelee backbone to prove that RHF can use a lightweight backbone to learn rich information from RHF. From Table 5, our proposed RHF model outperforms the SoTA models to prove its capability for real-time applications on TX2 embedded device.

Table 5. Ablation Study with SoTA backbone on Jetson TX2.

Method	Backbone	Size	FPS*	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv3	darknet53	416	2		55.3				
v3-tiny	Tiny15	416	17	-	33.1	-	-	-	-
Pelee	Pelee	304	14	22.4	38.3	22.9	-	-	-
RHF	Pelee	416	23	**26.7**	**49.5**	**26.3**	**10.3**	**28**	**37.4**
PRN[49]	Pelee	416	27	-	45	-	8.1	24.4	34.7

*FPS including preprocessing, model inference, and postprocessing time.

4.4. Comparisons with SoTA Models

To compare the efficiency and accuracy of the RHF net with SoTA models, the inference time is calculated for a single image by taking the sum of the CNN time and NMS (non-maximum suppression) time of 999 random images, and divide by 999. Table 6 shows the comparisons with other SoTA methods. All the compared methods were evaluated on NVIDIA Titan X. The corner-net [24] is not compared due to its inefficiency with a huge backbone.

Inherited from the nature of YOLO V3, our RHF model is not suitable for smaller input size such as 320×320. Thus, for the input size 300×300 and 320×320, our RHF net is lower than M2Det-320 and LRFNet-320. However, our RHF-416 model outperforms other SoTA models under 320×320 input size for the metrics AP, AP50, AP75, AP_S, and AP_L. Although the input size of RHF-416 model is larger than M2Det-320, its efficiency is up to 50 *fps* which is nearly 2.5 times faster than M2Det-320 with the backbone ResNet-101. For the input size 512×512, RHF net-512 achieved the highest AP_{50} than the second best performance by M2Det-512-ResNet101, and the second best AP_S of 19.0 after M2Det 512-ResNet101. Although M2Det-512 outperformed other methods on MS COCO [14], its model is complicated and time-consuming for a real-time object detection task. If efficiency is considered, for the input size 608×608, the accuracy of our RHF model outperforms M2Det-512 with better accuracy (21.3 in APs) and double efficiency. For applications such as traffic flow estimation, counting from drone, or smart farming to detect small objects, our method will be more suitable than M2Det. Inference time vs. APs curve is shown in Figure 5. RHF has the advantage of hybrid fusion of multi-scale contextual features and computationally low yet feature preserving operations; therefore, it achieves outstanding speed-accuracy compared with SoTA methods.

5. Discussion and Conclusions

The effectiveness of the hybrid fusion module to improve the accuracy of small object detection is proven and can be generalized to different backbones. Because pooling is a cheaper operation than convolution, it is often adopted to design a light backbone for efficiency improvement. However, if an object is small, the pooling operation will also let it disappear in the last layer of feature pyramid. Shift-invariance should be carefully tackled to achieve high detection rates based on edge computing. The hybrid bottom-up fusion module will be a good solution to improve the accuracy of small object detection. Moreover, the recursive RESC module improves the contextual features of all scaled images (small, medium, and large). Experimental results prove that our RHF net is superior for real-time applications especially for small object detection.

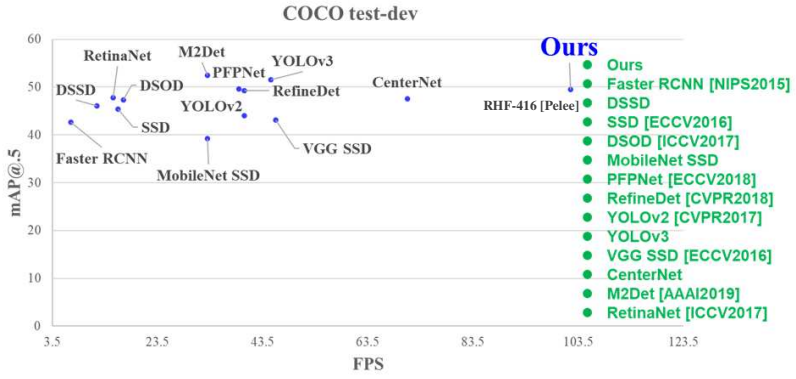

Figure 5. AP_{50} vs. Inference Time Curve. RHF nets are tested on NVIDIA Titan X.

Table 6. Comparisons on MS COCO test-dev set

Method	Backbone	Input size	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN [3]	VGGNet-16	~1000x600	7	21.9	42.7	-	-	-	-
R-FCN [3]	ResNet-101	~1000x600	9	29.9	51.9	-	10.8	32.8	45
Faster R-CNN w/ FPN [3]	ResNet-101-FPN	~1000x600	6	36.2	59.1	39	18.2	39	48.2
Cascade R-CNN	ResNet-101-FPN	~1280x800	7	42.8	62.1	46.3	23.7	45.5	55.2
Mask-RCNN	ResNet-101-FPN	~1280x800	5	39.8	62.3	43.4	22.1	43.2	51.2
SNIP	DPN-98	-	-	45.7	67.3	51.1	29.3	48.8	57.1
Deformable R-FCN	ResNet-101	~1000x600	8	34.5	55	-	14	37.7	50.3
SSD-300 [8]	VGGNet-16	300x300	43	25.1	43.1	25.8	6.6	25.9	41.4
SSD [8]	ResNet-101	321x321	50	28	45.4	29.3	6.2	28.3	49.3
YOLOv3-320 [7]	DarkNet-53	320x320	45	-	51.5				
CFENet-300 [30]	VGGNet-16	-	-	30.2	50.5	31.3	12.7	32.7	46.6
RefineDet-320 [29]	VGGNet-16	320x320	38.7	29.4	49.2	31.3	10	32	44.4
RefineDet-320 [29]	ResNet-101	320x320	-	32	51.4	34.2	10.5	34.7	50.4
RFBNet [48]	VGGNet-16	300x300	67	30.3	49.3	31.8	11.8	31.9	45.9
EFIP	VGGNet-16	300x300	71	30	48.8	31.7	10.9	32.8	46.3
PFPNet-R320 [13]	VGGNet-16	320x320	33	31.8	52.9	33.6	12	35.5	46.1
M2Det-320 [29]	VGGNet-16	320x320	33.4	33.5	52.4	35.6	14.4	37.6	47.6
M2Det-320 [29]	ResNet-101	320x320	21.7	34.3	53.5	36.5	**14.8**	38.8	47.9
LRFNet [12]	VGGNet-16	300x300	**77**	32	51.5	33.8	12.6	34.9	47
LRFNet [12]	ResNet101	300x300	53	34.3	54.1	36.6	13.2	38.2	**50.7**
RHF [Ours]	DarkNet-53	320x320	61	32.3	53.4	34.8	13.8	36.5	48.9
YOLOv3-416 [7]	DarkNet-53	416x416	35	-	55.3	-	-	-	-
RHF-416 [Ours]	Pelee	416x416	**103**	26.7	49.5	26.3	10.3	28	37.4
RHF-416 [Ours]	DarkNet-53	416x416	50	**35.2**	**57.5**	**37.6**	**15.9**	37.5	48.9
YOLOv2 [9]	DarkNet-19	544x544	40	21.6	44	19.2	5	22.4	35.5
YOLOv3-608 [7]	DarkNet-53	608x608	19.8	33	57.9	34.4	18.3	35.4	41.9
SSD-512 [8]	VGGNet-16	512x512	22	28.8	48.5	30.3	10.9	31.8	43.5
SSD-512 [8]	ResNet101	513x513	31.3	31.2	50.4	33.3	10.2	34.5	49.8
DSSD	ResNet101	513x513	6.4	33.2	53.3	35.2	13	35.4	51.1
RefineDet-512 [29]	VGGNet-16	512x512	22.3	33	54.5	35.5	16.3	36.3	44.3
RefineDet-512 [29]	ResNet101	512x512	-	36.4	57.5	39.5	16.6	39.9	51.4
Rev-Dense	VGGNet-16	512x512	-	31.2	52.9	32.4	15.5	32.9	43.9
CFENet-512 [30]	VGGNet-16	-	-	34.8	56.3	36.7	18.5	38.4	47.4
RFBNet [48]	VGGNet-16	512x512	33	33.8	54.2	35.9	16.2	37.1	47.4
RFBNet-E [48]	VGGNet-16	512x512	30	34.4	55.7	36.4	17.6	37	47.6
RetinaNet [17]	ResNet-101-FPN	~832x500	11	34.4	55.7	36.8	14.7	37.1	47.4
RetinaNet+AP-Loss[17]	ResNet-101-FPN	512x512	11	37.4	58.6	40.5	17.3	40.8	51.9
EFIP	VGGNet-16	512x512	34	34.6	55.8	36.8	18.3	38.2	47.1
PFPNet-S512 [13]	VGGNet-16	512x512	24	33.4	54.8	35.8	16.3	36.7	46.7
PFPNet-R512 [13]	VGGNet-16	512x512	24	35.2	57.6	37.9	18.7	38.6	45.9
M2Det-512 [29]	VGGNet-16	512x512	18	37.6	56.6	40.5	18.4	43.4	51.2
M2Det-512 [29]	ResNet-101	512x512	15.8	**38.8**	59.4	**41.7**	**20.5**	**43.9**	**53.4**
LRFNet [12]	VGGNet-16	512x512	38	36.2	56.6	38.7	19	39.9	48.8
LRFNet [12]	ResNet-101	512x512	31	37.3	58.5	39.7	19.7	42.8	50.1
RHF-512 [Ours]	ResNet-101	512x512	29.1	37.7	59.8	40.1	19.9	42.9	51.5
RHF-608 [Ours]	DarkNet-53	608x608	32	37.1	**60.3**	40.0	**21.3**	39.9	45.5

References

[1] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv:1409.1556, 2014.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," In *CVPR*, 2014.

[3] S. Ren, et al., “Faster r-cnn: Towards real-time object detection with region proposal networks,” In NIPS, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” In ECCV, 2014.

[5] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask r-cnn,” In *ICCV*, 2017.

[6] T.Y. Lin, et al., “Feature pyra-mid networks for object detection,” In *CVPR*, 2017.

[7] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” arXiv:1804.02767, 2018.

[8] W. Liu, et al., “SSD: Single shot multibox detector,” In ECCV, 2016.

[9] J. Redmon and A. Farhadi. “Yolo9000: better, faster, stronger,” arXiv:1612.08242, 2016.

[10] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” In *CVPR* 2016.

[11] D. Scherer, A. Muller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” In ICANN, 2010.

[12] T. Wang, et al. “Learning Rich Features at High-Speed for Single-Shot Object Detection,” In ICCV 2019.

[13] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, “Parallel Feature Pyramid Network for Object Detection,” In ECCV, 2018.

[14] S. Y. Woo, et al., “Gated bidirectional feature pyramid network for accurate one-shot detection,” vol. 30, pp.543-555, machine vision and applications, 2019.

[15] P. Sermanet, et al., “Overfeat: Integrated recognition, localization and detection using convolutional networks,” In ICLR, 2014.

[16] J. Redmon, *et al*., “You only look once: Unified, real-time object detection,” In CVPR, 2015.

[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal Loss for Dense Object Detection,” In ICCV, 2017.

[18] R. Zhang, “Making Convolutional Networks Shift-Invariant Again,” In ICLR 2019.

[19] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, “Single-Shot Refinement Neural Network for Object Detection,” In CVPR, 2018.

[20] R. J. Wang, X. Li, C. X. Ling, “Pelee: A Real-Time Object Detection System on Mobile Devices,” In NIPS, 2018.

[21] G. Huang, Z. Liu, L. van der Maaten, L., K.Q. Weinberger, “Densely connected convolutional networks,” In CVPR, 2017.

[22] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, “Novel dataset for fine-grained image categorization: Stanford dogs,” In Proc. CVPR Workshop on Fine-Grained Visual Categorization, volume 2, page 1, 2011.

[23] C.Y. Wang, *et al*., “CSPNet: A New Backbone that can Enhance Learning Capability of CNN,” arXiv e-prints arXiv:1911.11929 (Nov 2019).

[24] H. Law and J. Deng, “CornerNet: Detecting Objects as Paired Keypoints,” In ECCV, 2018.

[25] X. Wu, et al., “Single-Shot Bidirectional Pyramid Networks for High Quality Object Detection,” AAAI 2018.

[26] Z. Tian, C. Shen, H. Chen, and T. He, “Fcos: Fully convolutional one-stage object detection,” In CVPR 2019.

[27] C. Zhu, Y. He, and M. Savvides, “Feature selective anchor-free module for single-shot object detection,” In CVPR 2019.

[28] J. Cao, Y. Pang, J. Han, and X. Li, “Hierarchical Shot Detector,” In ICCV 2019.

[29] Q. Zhao, *et al*., “M2Det: A Single-Shot Object Detector based on Multi-Level Feature Pyramid Network,” AAAI 2019.

[30] Krizhevsky, A., Sutskever, I., & Hinton, G. E., “ImageNet classification with deep convolutional neural networks,” In *Communications of the ACM*, 60(6), 84–90. 2012.

[31] https://rebootingcomputing.ieee.org/lpirc/2019

[32] X. Wu, D. Zhang, J. Zhu, and C.H. Hoi, “Single Shot Bidirectional Pyramid Networks for High Quality Object Detection,” In AAAI 2018.

[33] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, Y. Wei, “Deformable convolutional networks,” In ICCV, 2017.

[34] T. Kong, A. Yao, Y. Chen, F. Sun, “HyperNet: Towards Accurate Region Proposal Generation and Joint Object Detection,” In CVPR, 2016.

[35] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, “Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks,” In CVPR, 2016.

[36] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, “CoupleNet: Coupling Global Structure with Local Parts for Object Detection,” In ICCV, 2017.

[37] B. Bosquet, et al., “STDnet: A ConvNet for Small Target Detection,” In BMVC, 2018.

[38] S. Bell, C. L. Zitnick, K. Bala, R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” In CVPR, 2016.

[39] J. Jeong, H. Park, N. Kwak, “Enhancement of ssd by concatenating feature maps for object detection,” In BMVC, 2017.

[40] C. Eggert, S. Brehm, A. Winschel, D. Zecha and R. Lienhart, "A closer look: Small object detection in faster R-CNN," In Proc. *IEEE Int. Conf. on Multimedia and Expo*, Hong Kong, pp. 421-426, 2017.

[41] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, K. Cho, “Augmentation for small object detection,” arXiv:1902.07296v1, 2019.

[42] A. G. Howard, et al., “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” In ECCV, 2018.

[43] X. Zhang, et al., “ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices,” In *ICCV*, 2017.

[44] https://sites.google.com/site/daviddo0323/

[45] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, C.L. Zitnick, “Microsoft coco: Common objects in context,” In *ECCV*, 2014.

[46] Z. Cai, Z. and N. Vasconcelos, “Cascade R-CNN: Delving into high quality object detection,” arXiv preprint arXiv:1712.00726, 2017.
[47] Y. Bai, Y. Zhang, M. Ding, B. Chanem, “SOD-MTGAN: Small Object Detection via Multi-Task Generative Adversarial Network,” In ECCV, 2018.
[48] Songtao Liu, Di Huang, and Yunhong Wang, “Receptive field block net for accurate and fast object detection,” In Proceedings of European Conference on Computer Vision, 2018.
[49] C. Wang, M. Liao, P. Chen, and J. Hsieh,” Enriching Variety of Layer-wise Learning Information by Gradient Combination”, In LPIRC, 2019.