# PoolNet+: Exploring the Potential of Pooling for Salient Object Detection

Jiang-Jiang Liu, Qibin Hou, Zhi-Ang Liu, Ming-Ming Cheng

**Abstract**—We explore the potential of pooling techniques on the task of salient object detection by expanding its role in convolutional neural networks. In general, two pooling-based modules are proposed. A global guidance module (GGM) is first built based on the bottom-up pathway of the U-shape architecture, which aims to guide the location information of the potential salient objects into layers at different feature levels. A feature aggregation module (FAM) is further designed to seamlessly fuse the coarse-level semantic information with the fine-level features in the top-down pathway. We can progressively refine the high-level semantic features with these two modules and obtain detail enriched saliency maps. Experimental results show that our proposed approach can locate the salient objects more accurately with sharpened details and substantially improve the performance compared with the existing state-of-the-art methods. Besides, our approach is fast and can run at a speed of 53 FPS when processing a $300 \times 400$ image. To make our approach better applied to mobile applications, we take MobileNetV2 as our backbone and re-tailor the structure of our pooling-based modules. Our mobile version model achieves a running speed of 66 FPS yet still performs better than most existing state-of-the-art methods. To verify the generalization ability of the proposed method, we apply it to the edge detection, RGB-D salient object detection, and camouflaged object detection tasks, and our method achieves better results than the corresponding state-of-the-art methods of these three tasks. Code can be found at http://mmcheng.net/poolnet/.

**Index Terms**—Salient object detection, feature aggregation, global guidance, pooling techniques, mobile application

## 1 Introduction

Salient object detection aims to detect the most visually distinctive objects from a given image and has gained great attention for its importance in many computer vision tasks, such as visual tracking [2], content-aware image cropping and editing [3], [4], image retrieval [5], video segmentation [6], robot navigation [7], and weakly-supervised semantic segmentation [8], [9]. As a basic vision task, salient object detection has gradually become an indispensable part of computer vision and has great meaning in the research of higher-level vision problems. In recent years, the development of salient object detection has been greatly promoted by convolutional neural networks (CNNs) for their capability of extracting both high-level semantics and low-level details in multiple scale-spaces than the traditional methods that relied on hand-crafted features. A typical characteristic of modern CNNs is their pyramidal structure, where the feature maps outputted by shallower layers usually have larger spatial sizes and maintain sophisticated and detailed low-level patterns. In comparison, the ones from deeper layers encode high-level semantics and exact locations of the salient objects. Various new architectures [10]–[12] have been proposed based on the above observation. The U-shape structures [13], [14] draw the most interest among these approaches for their simplicity in building enriched feature maps by augmenting the bottom-up classification networks with top-down pathways.

Though the above type of approaches has achieved good performance, we argue that there is still a large room to improve. One typical shortcoming of the U-shape structure is that the global semantic information collected by the top-most layer may be disturbed and gradually diluted by the massive local patterns in the shallower layers when being progressively transmitted in the top-down pathway, as shown in the top row of Fig. 1. This shortcoming harms the capability of these approaches in accurately discovering and segmenting out every part of the salient objects (see Fig. 3 for more details). Another shortcoming is that the receptive field of a CNN model does not grow proportionally with its layer depth [15]. It will make the output layer(s) lack sufficient high-level semantic information to determine where the salient objects are. To make up for these shortcomings, existing methods propose to introduce attention mechanisms [16], [17] into the U-shape structures, refine feature maps in a recurrent way [16], [18], [19], combine multi-scale feature information [10], [20], [21], or add extra constraints to the saliency maps [20].

Unlike the methods mentioned above, in this paper, we propose to remedy these shortcomings by exploring the potential of the efficient pooling techniques in the U-shape-based architectures. Taking into account the above analysis, we design the network structure mainly basing on two principles. On the one hand, features from deep layers that contain the location information of the salient objects should be delivered to all pyramid levels of the U-shape architecture so that the high-level semantics would not be diluted. On the other hand, since the feature maps at different pyramid levels of the U-shape architecture are often with different resolutions, how to seamlessly merge feature maps from different pyramid levels is also essential for retaining the original shapes of the detected salient objects.

Regarding the above design criteria, our model consists of two primary modules based on the feature pyramid networks[1] (FPNs) [14]: a global guidance module (GGM) and a feature aggregation

- *The authors are with the College of Computer Science, Nankai University (j04.liu@gmail.com). M.M. Cheng is the corresponding author (cmm@nankai.edu.cn).*
- *A preliminary version of this work has appeared in CVPR 2019 [1].*

[1] In what follows, the U-shape architecture we use in this paper by default refers to the feature pyramid networks (FPNs) [14].

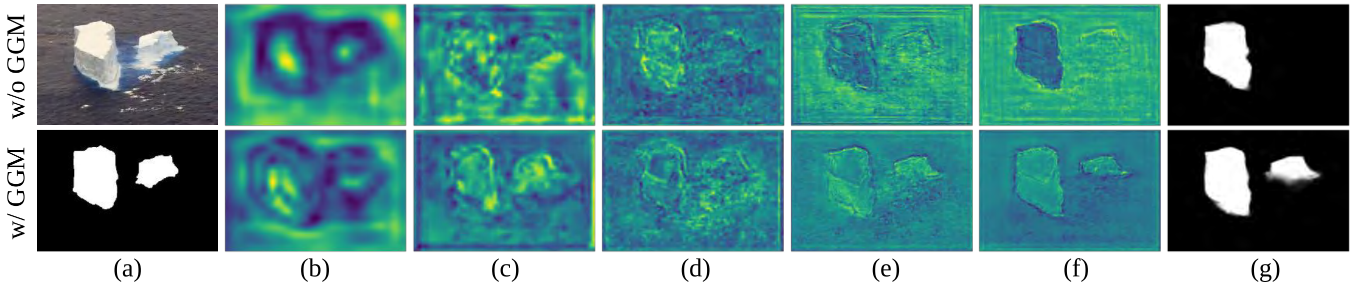


Fig. 1: Feature maps captured at different pyramid levels of FPNs. (a) Source image and its annotation; (b-f) Feature maps captured from high to low levels of FPNs; (g) Prediction results. The location information captured by deep layers is gradually diluted when building the pyramid in the original FPNs (top row). However, when adding global guidance to each level of the pyramid (bottom row), the locations of salient objects can be better rendered. This phenomenon is especially clear when the salient object is less salient (*e.g.*, the right iceberg in the example).

module (FAM). As shown in Fig. 2, the GGM consists of a modified pyramid pooling module (PPM) and a series of global guiding flows (GGFs). The GGFs transmit high-level semantic information collected by the PPM to feature maps at all pyramid levels, remedying the drawback of FPNs that top-down signals are gradually diluted. Considering the fusion problem of the coarse-level feature maps from the GGFs with the feature maps at different scales of the pyramid, we further propose the FAM, which takes the feature maps after fusion as input. FAM first converts the fused feature maps into multiple feature spaces to capture local context information at different scales. It then combines the information to weigh the compositions of the fused input feature maps better.

This paper is an extended version of our previous work [1]. In particular, (a) we remove the edge-related parts and now depend on no extra training data. (b) We propose an advanced version of FAM, called FAM+, which produces richer feature representations and achieves better performance than our previous method and other state-of-the-art algorithms. (c) We dissect the computational complexity components of our approach and cut off the redundancy for efficiency while maintaining its performance. (d) We propose a light-weighted version of our approach to satisfy the demands on mobile devices. (e) We include additional detailed analysis, more quantitative and qualitative ablation results to help comprehensively analyze the design criteria of the proposed approach and better understand why it can achieve good performances and a fast speed. (f) We apply our approach to the edge detection, RGB-D salient object detection, and camouflaged object detection tasks to demonstrate its generalization ability.

It has been shown in our conference version that FAMs help discover rich local details. In this paper, we show that this ability can be further advanced. Unlike the original FAM that the feature transformations in different scale-spaces are conducted individually, inspired by [22], FAM+ explicitly builds internal communications between the parallel branches so that the output feature representations can be further enriched. Compared to FAM, FAM+ does not introduce any learnable parameters but largely improves the performance. We will give more analysis and numerical results in our experiment section.

Our original network is called PoolNet in that the new modules we design are mainly based on the pooling techniques. To distinguish with our conference version [1], we call our new network with FAM+ as PoolNet+. Here, '+' means an improved version. To the best of our knowledge, we are the first to study how to design effective pooling-based modules to improve the performance of salient object detection models. To evaluate the performance of PoolNet+, we report results on five popular salient object detection benchmarks. Without bells and whistles, PoolNet+ outperforms all previous state-of-the-art methods by a large margin. Also, we conduct a series of ablation experiments to let readers better understand the impact of each component in PoolNet+ on the performance. Other than attractive model performance, PoolNet+ can also run fast. We achieve a speed of 53 FPS on a single NVIDIA RTX-2080Ti GPU for an input image of size $300 \times 400$. Training PoolNet+ takes less than 7 hours on a training set of 10,533 images, which is much faster than most of the previous methods [10], [17], [20], [21], [23], [24].

Regarding applications in mobile devices, we also present a light-weighted version of PoolNet+, named PoolNet-M+, which can run at 66 FPS without sacrificing too much on performance ($\sim$ $1\%$ drop in F-measure) but with only a small number of parameters ($\sim$ 3 M) and MAdds ($\sim$ 1.2 G). It is mainly due to the effective utilization of pooling techniques that require a small number of computational resources. PoolNet+ also shows great generalization ability, which achieves state-of-the-art results when applied to edge detection, RGB-D salient object detection, and camouflaged object detection. PoolNet+, therefore, can be viewed as a baseline to help ease future research in salient object detection.

The rest of our paper is organized as follows. Sec. 2 briefly reviews previous researches that are strongly related to this work. Sec. 3 describes the proposed PoolNet and PoolNet+ and analyzes the functions of the proposed pooling techniques. Sec. 4 extends PoolNet+ to a light-weighted version, named PoolNet-M+. Sec. 5 gives the experimental results of our approach in both salient object detection and edge detection. Sec. 6 discusses the redundancy, efficiency, and failure cases. Sec. 7 applies our approach to the task of edge detection. Finally, Sec. 8 concludes the whole paper.

## 2 Related Work

### 2.1 Salient Object Detection

Traditional salient object methods are mostly based on hand-crafted features [25]–[28]. Early deep learning-based methods usually predicted the saliency scores regionally by using features extracted from image patches or super-pixels [29]–[32] or object proposals [33]–[35]. These methods were time-consuming for the extracted regional features needed to be processed sequentially. Inspired by the success of fully convolutional networks [36], recent research mostly focuses on designing saliency models that make pixel-wise predictions. In this section, we will only review representative approaches that are based on CNNs. Readers may refer to recent review work [37]–[39] for more details about traditional methods. The CNN-based approaches can be briefly categorized into five categories [39] according to their network architectures.

**Single-stream** network is the simplest structure. It consists of a sequence of convolutional layers, non-linear layers, and several pooling layers for spatially down-sampling. The prediction is made directly at the end of the network. Because there are few places where a single-stream network can be modified, most methods focus on other aspects. For example, [24] proposed to enhance its model's robustness by utilizing a re-formulated dropout mechanism and designed a hybrid up-sampling function for more accurate prediction. In [19], the idea of recurrent learning was utilized by taking the saliency prior maps generated from low-level cues as guidance. [40] proposed to stage-wisely refine the coarse prediction maps generated in the early stages with more details.

**Multi-stream** network usually takes inputs with different resolutions and utilizes multiple network streams to generate multi-scale saliency features. For example, [41] designed a two-stream network consisting of a pixel-level fully convolutional stream and a segment-wise spatial pooling stream to directly produce a pixel-level saliency map and efficiently extract segment-wise features, respectively. The complementary features were then fused for final prediction. [42] aimed to make high-resolution saliency prediction by building a global semantic network and a local refinement network to extract global and local information, respectively.

**Side-fusion** network takes advantage of the spatial pyramid structure of CNNs and utilizes the extracted multi-scale features for prediction. A typical characteristic of the side-fusion network is that the side-outputs will also be supervised by the ground-truth. [10] added additional short-connections from deeper to shallower side-outputs to better utilize the location information in higher-level features. [20] built a multi-resolution $4 \times 5$ grid structure to efficiently combine local and global information and proposed a Bayesian loss that penalized errors on the boundary to enforce spatial coherence. [43] proposed to discard low-level features to reduce the computational complexity of aggregation modules and utilize generated attention maps to refine high-level features to improve performance. [44] enhanced high-level context information with channel-wise attention and low-level structural features with spatial attention.

**U-shape** network refines the coarse saliency features produced by the top-most layer by gradually aggregating the finer features from its lower layers with an extra top-down pathway. Prediction is directly made at the end of the top-down pathway. Among the well-known U-shape networks, [16], [17], [45], [46] designed various attention-based modules or mechanisms for improvement. Despite the attention, [12] employed a recurrent module to progressively refine the inner structure of its CNN over time and a specialized network to learn the local pixel-wise contextual information for boundary recovering. [46], [47] introduced different boundary-aware losses as an assistant to learn exquisite object boundaries. [23] built a gated bi-directional network to effectively exchange information among the multi-level features extracted by the backbone network. [48] proposed to integrate both top-down and bottom-up saliency inference iteratively and cooperatively for more effective optimization.

**Multi-branch** network usually has multiple output branches corresponding to different tasks. It utilizes the concept of multi-task learning to help the network with better feature extraction ability. For instance, the network designed in [11] could simultaneously estimate salient objects contours and saliency maps, which would implicitly drive the intermediate features to concentrate more on the edge pixels. Similarly, [49] stacked a series of cross refinement units to simultaneously refine multi-level features of salient object detection and edge detection. [50] supervised the network for saliency detection, additionally with foreground contour detection and edge detection to alleviate the incomplete predictions. [51] extended the salient object detection task further by utilizing eye fixations information.

The proposed approach is based on the U-shape structure. However, unlike the approaches above that solved the task by designing various network architectures, advancing attention mechanisms, or including more supervisions, we investigate how to utilize the efficient pooling techniques to solve the global information dilution problem and feature fusion problem across large scales in the U-shape structure.

## 2.2 Pooling

As a key component in modern CNNs, pooling mainly has two functions. The first is to reduce the spatial size of feature maps, thereby reducing the computational cost. The second is to enhance the translation invariance capability and help relieve the overfitting problem during optimization.

**Basic Operations.** As the two most commonly used types of pooling operations, average [52], [53] and maximum [54] pooling aims to select the average and maximum value inside the target pooling window as its output, respectively. For average pooling, the gradient is evenly distributed to all pixels in the pooling window during back-propagation. For maximum pooling, the gradient only considers the maximum value pixel while the gradients of the rest pixels inside the pooling window are set to zero. [55] proposed a mixing strategy and a gating strategy to combine maximum and average pooling, and further introduced a more complicated tree-structured self-learning pooling strategy.

**Advanced Operations.** Instead of basing on the basic average and maximum pooling, [56] presented a lossless pooling for image super-resolution that could down-sample a single-channel map to a multi-channel map with lower spatial resolution without information loss. [57] designed a local importance-based pooling that could learn adaptive and discriminative importance maps to aggregate features for down-sampling instead of the hand-crafted ones. [58] studied the shape of pooling windows and exploited a lightweight strip pooling strategy that adopted long and narrow pooling windows both vertically and horizontally.

Different from the above methods that aimed to design various pooling functions, we explore how pooling can be utilized to build a more efficient and effective salient object detection network. The proposed network can cooperate with most of the pooling operations mentioned above. We will show that the proposed network can generalize well to more sophisticated pooling operations in the experiment section, other than only the basic ones.

## 2.3 Edge Detection

Edge detection is one of the most fundamental problems in computer vision. Traditional methods, such as Canny [59], mainly

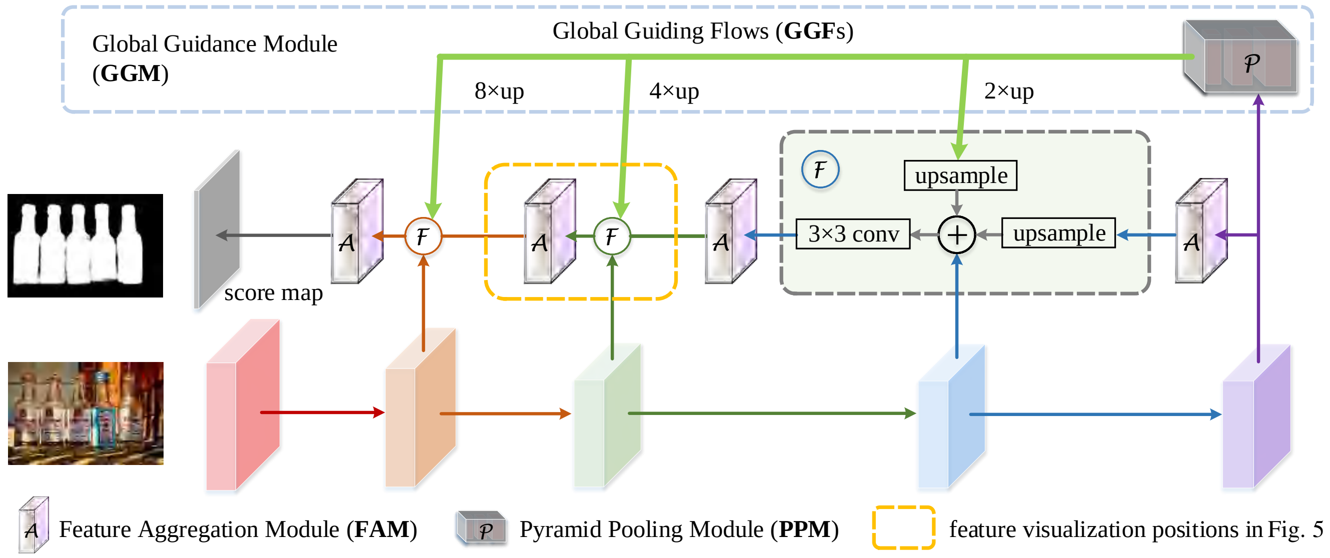

Global Guidance Module
(GGM)
Global Guiding Flows (GGFs)
8×up
4×up
2×up
P
F
upsample
3×3 conv
upsample
A
F
score map
Feature Aggregation Module (FAM)
Pyramid Pooling Module (PPM)
feature visualization positions in Fig. 5

Fig. 2: Pipeline of our salient object detection model, where high-level semantic features containing the location information of salient objects can be delivered to each pyramid level in the top-down pathway. We use a pyramid pooling module (PPM) to locate salient objects better and introduce global guiding flows to deliver the captured location information to fuse with the features at each pyramid level. After each fusion, a feature aggregation module (FAM) is connected to help reduce the aliasing effect and enrich the details.

focused on utilizing the intensity and color gradients. Later, many feature learning methods based on information theory [60]–[63] were proposed, which attempted to employ various hand-crafted features to capture information from both local and global aspects.

Due to strong features extracted by CNNs, deep learning-based methods have recently overwhelmed the traditional approaches in accuracy and speed. [64] combined CNNs with the nearest neighbor search by proposing $N^4$-Fields. [65] partitioned the contour data into sub-classes and fit each sub-class by different model parameters. [66] introduced the concept of deep supervision and applied it to CNNs by adding extra supervision to the last convolutional layer of each stage. [67] extended edge detection with the concept of semantic segmentation and proposed a model to detect and recognize the semantic categories of edge pixels simultaneously. [68] proposed to learn crisp boundaries by introducing a top-down backward refinement pathway. [69] extended HED [66] by adding side supervision to the combination of the outputs of all convolutional layers in each stage instead of the last one.

Though the proposed approach is targeted at salient object detection, we will show that it generalizes well to the edge detection task and performs comparably to the above methods specifically tailored for edge detection.

# 3 PoolNet

It has been pointed out in [10], [12], [18], [40] that high-level semantic features help discover the specific locations of salient objects. In the meantime, low- and mid-level features are also essential for improving the features extracted from deep layers from a coarse level to a fine level. Based on the above knowledge, in this section, we propose a couple of pooling-based modules capable of accurately capturing the exact positions of salient objects and sharpening their details.

## 3.1 Overall Pipeline

We build our architecture based on the feature pyramid networks (FPNs) [14], which belong to a type of classic U-shape architecture designed in a bottom-up and top-down manner. Due to its strong ability to fuse multi-level features from the backbone networks [70], [71], this type of architecture has been widely adopted in many computer vision tasks, including salient object detection. Despite so, a key problem of FPNs is that the high-level semantic information is progressively transmitted to lower layers, which makes the location information captured by deep layers gradually diluted.

Given the FPN structure, as shown in Fig. 2, we introduce a global guidance module (GGM) built upon the top of the bottom-up pathway. By aggregating the high-level information extracted by GGM into feature maps at each feature level, our goal is to explicitly notice the layers at different pyramid levels where the salient objects are. After the guidance information from GGM is merged with the features at different levels, in our original conference version, we introduce a feature aggregation module (FAM) to ensure that feature maps at different scales can be well merged. This paper further advances the original FAM structure and presents a new version: FAM+, which we found can better capture local details. In what follows, we describe the structures of the modules mentioned above and explain their functions in detail.

## 3.2 Global Guidance Module

While FPNs provide a classic architecture for combining multi-level features from the classification backbone, the problem for this type of architecture is that the high-level features will be gradually diluted when transmitted to lower layers because the top-down pathway is built upon the bottom-up backbone. It has been shown in [15], [72] that the empirical receptive fields of CNNs are much smaller than the ones in theory, especially for deeper layers. Hence, the receptive field of the whole network is not large enough to capture the global information of the input

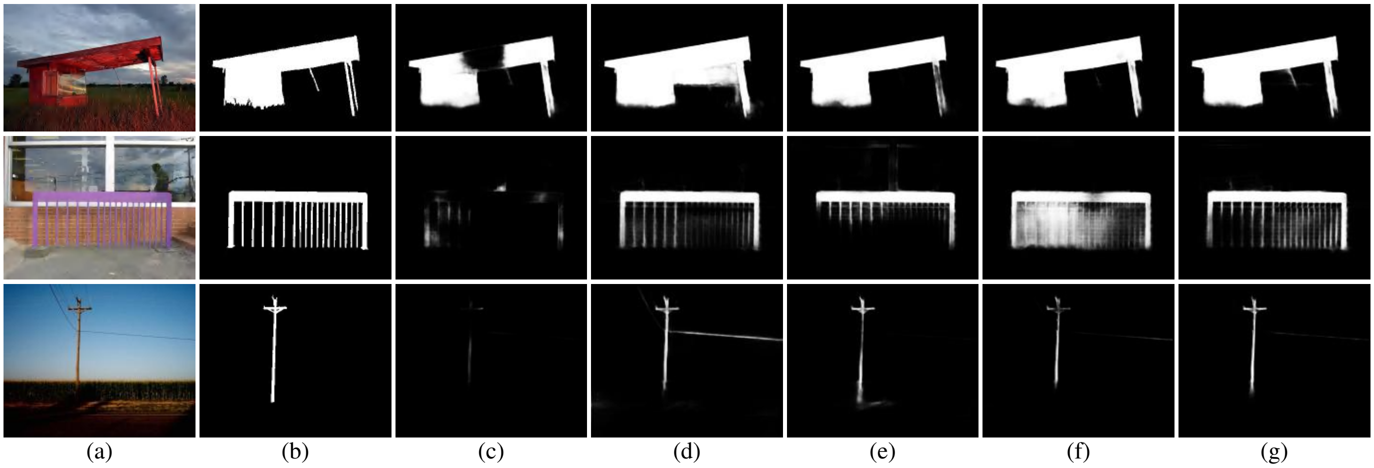


Fig. 3: Visual comparisons for salient object detection among different configurations of our approach. (a) Source image; (b) Ground truth; (c) FPN baseline [14]; (d) FPN + FAMs; (e) FPN + PPM (pyramid pooling module); (f) FPN + GGM; (g) FPN + GGM + FAMs. Adding GGM improves the ability to discover the accurate positions of salient objects substantially. More interestingly, the utilization of FAMs can further improve the quality of the resulting saliency maps in that the boundary details are well refined.

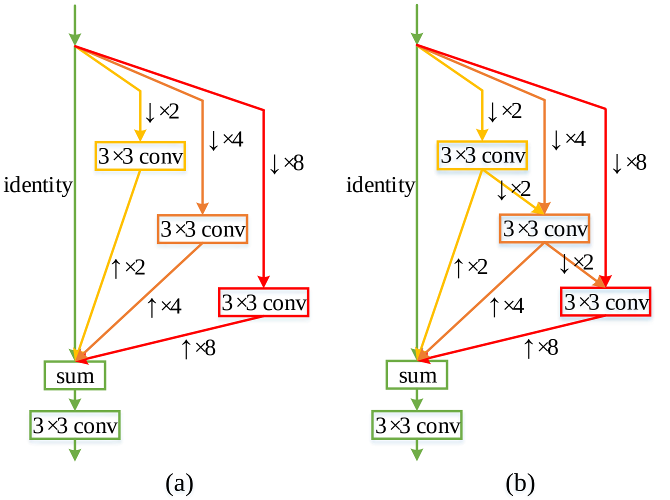


Fig. 4: Detailed illustrations of our feature aggregation module (FAM) and its advanced version (FAM+). (a) Original FAM, which comprises four parallel sub-branches and each of which works in an individual scale-space. After up-sampling, all these sub-branches are combined via summation and then fed into a convolutional layer. (b) Proposed FAM+, which introduces a series of short connections between different sub-branches to build internal communications explicitly.

images sufficiently. The immediate effect on this is that only parts of the salient objects can be discovered, as shown in Fig. 3c. To eliminate the lack of high-level semantic information for fine-level feature maps in the top-down pathway, we introduce a global guidance module (GGM). It contains a modified pyramid pooling module [15], [40] and a series of global guiding flows to explicitly make feature maps at each level be aware of the locations of the salient objects, as shown in Fig. 2.

To be more specific, the pyramid pooling module in our GGM consists of four sub-branches to capture the context information of the input images. The first and last sub-branches are an identity mapping layer and a global average pooling layer. For the two middle sub-branches, we adopt the adaptive average pooling layer[2] to ensure that their output feature maps are with spatial sizes $3\times3$ and $5\times5$, respectively. Given the pyramid pooling module, what we need to do now is to guarantee that the guidance information produced by it can be reasonably fused with the feature maps at different levels in the top-down pathway.

Quite different from the previous work [40], which simply views the pyramid pooling module as a part of the FPNs, our pyramid pooling module is independent of the FPNs. By introducing a series of global guiding flows (identity mappings), the high-level semantic information can be easily delivered to feature maps at various levels (see the green arrows in Fig. 2). In this way, we explicitly transmit the global guidance information into each part of the top-down pathway to ensure that the location information will not be diluted when building FPNs.

To better demonstrate the effectiveness of our GGM, we show some visual comparisons, where some saliency maps produced by a VGGNet version of FPNs[3] are illustrated in Fig. 3c. It can be easily found that with only the FPN baseline, it is difficult to locate salient objects for some complex scenes. There are also some results in which only parts of the salient object are detected. However, when our GGM is incorporated, the quality of the resulting saliency maps is greatly improved despite some losses in boundary details. As shown in Fig. 3f, salient objects can be precisely discovered, verifying the importance of GGM.

### 3.3 Feature Aggregation Module

The utilization of our GGM allows global guidance information to be delivered to feature maps at different pyramid levels. However, a new question that deserves asking is how to make the coarse-level feature maps from GGM seamlessly merge with the feature maps at different pyramid scales. Taking the VGGNet version of FPNs as an example, the feature maps corresponding to $C = \{C_2, C_3, C_4, C_5\}$ in the pyramid have down-sampling rates $\{2, 4, 8, 16\}$ corresponding to the size of the input image,

[2] https://pytorch.org/docs/stable/nn.html#adaptiveavgpool2d

[3] Similar to [14], we use the feature maps outputted by conv2, conv3, conv4, conv5 which are denoted by $\{C_2, C_3, C_4, C_5\}$ to build the feature pyramid upon the VGGNet [71]. The channel numbers corresponding to $\{C_2, C_3, C_4, C_5\}$ are set to $\{128, 256, 512, 512\}$, respectively.

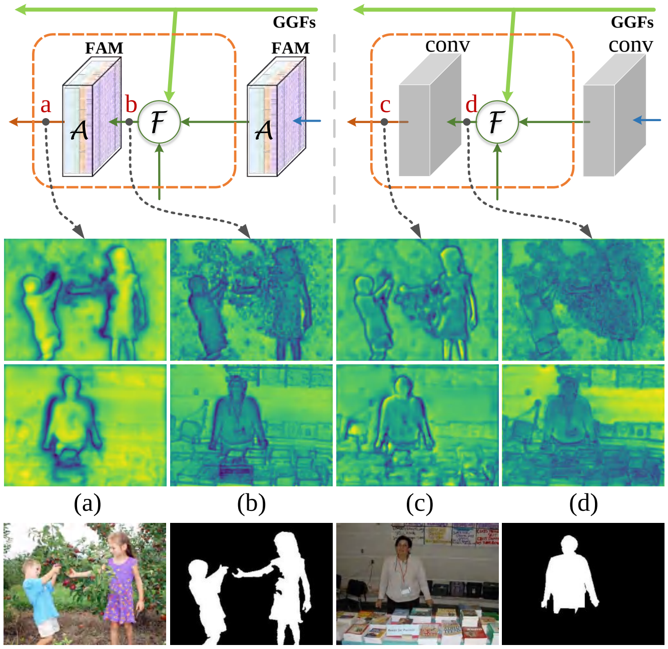


Fig. 5: Visualization of feature maps around FAMs. Feature maps shown on the left are from models with FAMs, while feature maps displayed on the right are from the models replacing FAMs with two convolution layers. The last row is source images and the corresponding ground-truth annotations. (a-d) are visualizations of feature maps at different places. As can be seen, when our FAMs are used, feature maps after FAMs can more precisely capture the location and detailed information of the salient objects (Column a), in comparison with those after two convolutional layers (Column c). Better and clearer effect can be observed when viewed in color.

respectively. In the original top-down pathway of FPNs, feature maps with coarser resolutions are up-sampled by a factor of 2. Therefore, adding a convolutional layer with kernel size $3\times3$ after the merging operation can effectively reduce the aliasing effect of up-sampling. However, some of the global guiding flows need larger up-sampling rates (*e.g.*, 8). It is essential to bridge the big spatial gaps between the global guiding flows and the feature maps of different scales efficiently and effectively.

To this end, we present a series of feature aggregation modules (FAMs), each of which contains four sub-branches, as illustrated in Fig. 4a. As can be seen, the input feature maps are first transformed into different scale-spaces by being fed into multiple average pooling layers with varying down-sampling rates. The feature maps from different sub-branches after convolutional transformations are then up-sampled and used as residuals to merge with the input feature maps. A $3 \times 3$ convolutional layer is attached after fusion. In general, FAM offers two advantages. On the one hand, it assists our model in reducing the aliasing effect caused by up-sampling operations, especially when the up-sampling rate is large (*e.g.*, 8). On the other hand, it allows each spatial location to view the local context in different scale-spaces, further enlarging the receptive field of the whole network. *To the best of our knowledge*, this is the first work revealing that reasonable utilization of pooling techniques helps reduce the aliasing effect of up-sampling, especially when the up-sampling rate is large.

To verify the effectiveness of our proposed FAMs, we visualize the feature maps near the FAMs in Fig. 5. By comparing the left part (w/ FAMs) with the right part (w/o FAMs), feature maps after FAMs (Column a) can better capture the salient objects than those without FAMs (Column c). In addition to visualizing the intermediate feature maps, we also show some saliency maps produced by models with different settings in Fig. 3. By comparing the results in Column f (w/o FAMs) and Column g (w/ FAMs), it can be easily found that introducing FAM multiple times allows our network to sharpen the details of the salient objects better. This phenomenon is especially clear by observing the second row of Fig. 3. All the discussion above verifies the significant effect of our FAMs on better fusing feature maps at different scales. In our experiment section, we will give more numerical results.

### 3.4 Advanced Feature Aggregation Module

In our conference version, we have investigated the effect of our FAMs on sharpening the object details and improving the model performance. In this part, we demonstrate that reasonably modeling the dependencies between sub-branches in the FAMs is beneficial to the saliency results as well. To distinguish with the FAMs described above, we call our new design advanced feature aggregation module, or FAM+ for short.

The structure of our FAM+ has been shown in Figure 4(b). Compared to our original FAM, which independently conducts feature transformations in different scale-spaces, we add a series of short connections (down-sampling operations) between adjacent sub-branches in FAM+. More concretely, the fine-level feature maps after convolutional transformation are not only directly up-sampled for fusion but also sent to the coarser-level sub-branch to conduct a new convolutional transformation. This design explicitly establishes internal communications between adjacent sub-branches, allowing the output features to be more discriminative. Compared to the original FAM, FAM+ requires no extra learnable parameters but leads to better performance. We will give more numerical results in Sec. 5.

### 3.5 Discussion

Our feature aggregation module (FAM) is proposed to bridge the large spatial gaps between the local contextual information and global guidance information with efficient pooling techniques. Existing feature aggregation modules solved this problem mostly by increasing the convolutional operations' kernel sizes or dilation rates. However, large kernel sizes mean more parameters and MAdds, while large dilation rates require more memory and slow down the speed. The pooling operations used in our FAM, on the contrary, introduce no additional parameters while reducing the spatial resolution of the feature maps and making the subsequent convolution operations have less memory and computational burden. In addition to the efficiency advantage, the pooling operations introduce more translation invariance and prevent overfitting. In general, FAM can reduce the aliasing effect caused by up-sampling operations, as shown in Fig. 5. It can also enlarge the receptive field of the whole network and obtain more precise location information and better performance, as shown in Fig. 3 and Table 1. We also advance FAM by explicitly building internal communications between the adjacent sub-branches and present FAM+, producing richer feature representations. FAM+ does not introduce any learnable parameters but largely improves the performance.

# 4 PoolNet-M

Due to the large number of parameters and MAdds in the backbone (compared to light-weighted models [73]), it is difficult to deploy PoolNet to applications on mobile devices directly. In many real-world scenarios, such as mobile phones and robotics, it's of great importance to carry out the detection and segmentation algorithms in a timely fashion on a computationally limited platform. As an alternative, in this section, we propose a light-weighted version of PoolNet by rethinking the trade-off between efficiency and effectiveness, which is abbreviated as PoolNet-M.

In this paper, we take the famous and successful MobileNetV2 [73] as an exemplar backbone to re-design our PoolNet-M. It is also worth mentioning that any other light-weighted classification networks can also be considered. MobileNetV2 contains an initial fully convolutional layer with 32 filters, followed by 19 inverted residual blocks divided into seven stages and a final fully-connected layer for classification. We remove the last fully-connected layer, change the strides of the $3 \times 3$ convolutional layers in the 6th stage to 1, and increase the dilation rates of the $3 \times 3$ convolutional layers in the last two stages to 2 to maintain the large receptive field. To build FPNs, we use feature maps from the last layers of stages {1, 2, 3, 5, 7}, which have down-sampling rates of {2, 4, 8, 16, 16}, respectively. Regarding the computational cost, five $1 \times 1$ convolutional layers with channel numbers {12, 16, 24, 36, 72} are connected to the stages mentioned above.

Directly using standard $3 \times 3$ convolutions in the FAM or FAM+ and the pyramid pooling module would introduce lots of learnable parameters. Depthwise separable convolutions [73] eliminate this problem by ingeniously combining the depthwise and pointwise convolutions for their capabilities in building spatial and inter-channel dependencies, respectively. To further reduce computational cost while keeping high performance, MobileNetV2 adopts inverted residual blocks, which reduce the channel numbers of $1 \times 1$ pointwise convolutions but expands the $3 \times 3$ depthwise bottleneck. Likewise, to make our model lighter, we use inverted residual blocks to replace the $3 \times 3$ convolutional layers in both the pyramid pooling module and FAM (FAM+). The input and output channels in the inverted residual blocks at each level are set to {12, 16, 24, 36, 72}, respectively, and the expansion rates are all set to 3. We will show in Sec. 5 that the new-designed PoolNet-M (PoolNet-M+) can achieve comparable performance against those existing state-of-the-arts with a large reduction of the learnable parameters and MAdds.

# 5 Experiments

This section first describes the experiment setups, including the implementation details, the datasets used, and the evaluation metrics. We then conduct a series of ablation studies to demonstrate the impact of each component of our proposed approach on the performance. Moreover, we report the performance of our approach under different settings and compare them with previous state-of-the-art methods.

## 5.1 Experiment Setup

**Implementation Details.** The proposed framework is implemented based on the PyTorch repository[4]. All experiments are carried out on a workstation with an Intel Xeon 12-core CPU (3.6GHz), 64GB RAM, and a single NVIDIA RTX-2080Ti GPU. The backbone parameters of our network (*e.g.*, VGG-16 [71], ResNet-50 [70], and MobileNetV2 [73]) are initialized with the corresponding models pre-trained on the ImageNet dataset [77], and the remaining ones are randomly initialized. We train the networks with heavy backbones for 36 epochs in total, and the initial learning rate is set to 5e-5, which is divided by 10 after 27 epochs. In comparison, the light-weighted network is trained for 60 epochs in total with an initial learning rate of 1e-4, which is divided by 10 after 50 epochs. For all experiments, the Adam [78] optimizer with a weight decay of 5e-4 is used, and the training batch size is set to 10. We use random rotation and horizontal flipping for data augmentation. In both training and testing phases, input images are resized to $384 \times 384$. Unlike our conference version, we do not use any additional training data in this paper as all the edge-related parts have been removed.

**Datasets.** To evaluate the performance of our proposed framework, we conduct experiments on five commonly used datasets, including ECSSD [79], PASCAL-S [74], DUT-OMRON [75], HKU-IS [29], and DUTS-TE [76].

**Loss Function.** The standard binary cross entropy loss is used as commonly done, which is defined as follows:

$$loss(S,G) = -\frac{1}{N}\sum_{k=1}^{N}[G_k log(S_k)+(1-G_k)log(1-S_k)], \quad (1)$$

where $S$ and $G$ denote the predicted map and the ground truth, respectively, while $k$ is the index of pixels and $N$ is the number of pixels in $S$.

**Evaluation Criteria.** We evaluate the performance of our approach and other methods using four widely-used metrics: precision-recall (PR) curves, F-measure score, mean absolute error (MAE), and Structural measure (S-measure) [80].

- The precision value is the ratio of ground truth salient pixels in the predicted salient region, while the recall value is the percentage of the detected salient pixels in all ground truth areas. The precision and recall values are calculated by comparing the predicted saliency map after thresholding with the corresponding ground truth. We plot the precision-recall curve at different thresholds using the average precision and recall of all images in the dataset.
- F-measure, denoted as $F_\beta$, is an overall performance measurement and is computed by the weighted harmonic mean of the precision and recall:

$$F_\beta = \frac{(1+\beta^2) \times Precision \times Recall}{\beta^2 \times Precision + Recall}, \quad (2)$$

where $\beta^2$ is set to 0.3 as done in previous work to weight precision more than recall. We report the maximum F-measure from all precision-recall pairs, which is a good summary of the method's detection performance [37].

- The MAE score is defined as the average pixel-wise absolute difference between the binary ground truth and the saliency map. It indicates how similar a saliency map $S$ is when compared to the ground truth $G$:

$$MAE = \frac{1}{W \times H}\sum_{x=1}^{W}\sum_{y=1}^{H}|S(x,y) - G(x,y)|, \quad (3)$$

[4] https://pytorch.org

| No. | FPN | GGM | | FAM/ | PASCAL-S [74] | | | DUT-OMRON [75] | | | HKU-IS [29] | | | DUTS-TE [76] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PPM | GGFs | FAM+ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ |
| 1 | ✓ | | | - | 0.825 | 0.090 | 0.816 | 0.760 | 0.071 | 0.779 | 0.910 | 0.041 | 0.889 | 0.830 | 0.054 | 0.835 |
| 2 | ✓ | ✓ | | - | 0.848 | 0.081 | 0.835 | 0.796 | 0.062 | 0.814 | 0.917 | 0.038 | 0.898 | 0.856 | 0.048 | 0.858 |
| 3 | ✓ | | ✓ | - | 0.833 | 0.087 | 0.823 | 0.773 | 0.068 | 0.791 | 0.913 | 0.040 | 0.892 | 0.839 | 0.051 | 0.844 |
| 4 | ✓ | ✓ | ✓ | - | 0.856 | 0.075 | 0.848 | 0.799 | 0.062 | 0.810 | 0.925 | 0.038 | 0.907 | 0.860 | 0.047 | 0.865 |
| 5 | ✓ | | | FAM | 0.855 | 0.080 | 0.840 | 0.808 | 0.061 | 0.821 | 0.923 | 0.039 | 0.903 | 0.863 | 0.049 | 0.865 |
| 6 | ✓ | ✓ | ✓ | FAM | 0.866 | 0.075 | 0.849 | 0.813 | 0.060 | 0.830 | 0.927 | 0.036 | 0.908 | 0.870 | 0.045 | 0.871 |
| 7 | ✓ | | | FAM+ | 0.856 | 0.080 | 0.841 | **0.817** | **0.059** | 0.826 | 0.925 | 0.036 | 0.907 | 0.872 | 0.045 | 0.869 |
| 8 | ✓ | ✓ | ✓ | FAM+ | **0.872** | **0.070** | **0.857** | **0.817** | **0.059** | **0.832** | **0.931** | **0.035** | **0.914** | **0.878** | **0.043** | **0.880** |

Table 1: Ablation analysis for the proposed GGM, FAM, and FAM+. As can be observed, each component in our architecture plays an important role and contributes to the performance. Especially, our new FAM+ works better than the original FAM in most cases. The best performance in each column is highlighted in **bold**.

where $W$ and $H$ denote the width and height of saliency map $S$, respectively.

- S-measure evaluates the structural similarity between the real-valued saliency map and its binary ground-truth. It considers the object-aware ($S_o$) and region-aware ($S_r$) structure similarities simultaneously:

$$S_m = \alpha \times S_o + (1 - \alpha) \times S_r, \tag{4}$$

where $\alpha$ is empirically set to 0.5.

## 5.2 Ablation Studies

We experiment with different module design options and network configurations to illustrate the effectiveness of each component of our method. By default, our ablation experiments are performed based on VGG-16 and the DUTS-TR [76] dataset unless special explanations. We test models under different settings of our approach on four challenging datasets: PASCAL-S, DUT-OMRON, HKU-IS, and DUTS-TE.

### 5.2.1 *Effectiveness of PoolNet*

In this subsection, except for different combinations of GGM and FAMs (or FAM+s), all other network configurations are kept the same.

**GGM Only.** The addition of GGM (the 4th row in Table 1) gives performance gains in all terms of F-measure, MAE, and S-measure on all four datasets over the FPN baseline. The global guidance information produced by GGM allows our network to focus more on the integrity of salient objects, greatly improving the quality of the resulting saliency maps. As shown in Table 1, simply adding GGM to the baseline FPN has a performance gain of around 4% on the DUT-OMRON dataset (0.799 *v.s.* 0.760) in F-measure and more than 3% on the same dataset in S-measure. A similar phenomenon can also be observed on the other three datasets. From Fig. 3 (Column f *v.s.* Column c), it can be easily found that the utilization of GGM helps discover more accurately where the salient objects are. Therefore, the details of the salient objects can be sharpened, which might be wrongly estimated as background for models with limited receptive fields (*e.g.*, Column c in the last row of Fig. 3).

**FAMs Only.** Simply embedding FAMs into the FPN baseline (the 5th *v.s.* 1st rows in Table 1) is helpful on almost all four datasets. For instance, compared to the results with no FAM incorporated, adding FAMs improves the F-measure scores on the DUT-OMRON and DUTS-TE datasets by 4.8% and 3.3%, respectively. It is because that the pooling operations inside FAMs also enlarge the receptive field of the whole network compared to the models without them. It can also be observed from the visual examples displayed in Fig. 3 (Column d *v.s.* Column c), where the predictions with FAMs involved have more integral segmentation results. Moreover, the FPN baseline, even with GGM incorporated, still needs to merge feature maps from different levels. The further improvement after adding FAMs (the 6th *v.s.* 5th rows in Table 1) indicates the effectiveness of our FAMs for solving the aliasing effect of up-sampling. The visual results in Fig. 3 (Column g *v.s.* Column f) also verify the above argument. As can be seen, models with FAMs introduced can better sharpen the details of the detected salient objects. However, for Column f, the corresponding model does not possess this ability to render the object boundaries clearly. The coarse-level features from deep layers after up-sampling cannot be well fused with fine-level features when no FAM involves, raising undesired aliasing effect and poor boundary quality.

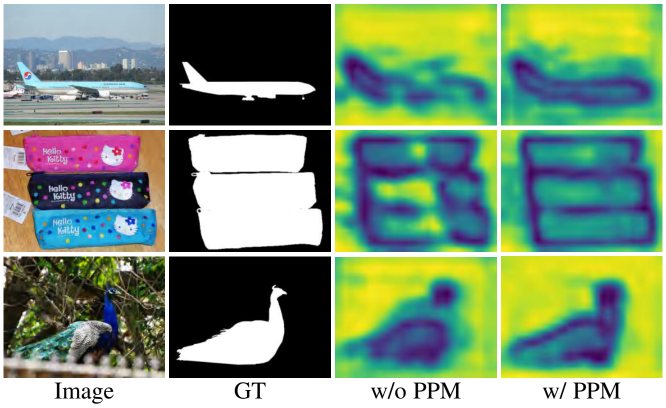


Fig. 6: Feature maps outputted by the last layer of the bottom-up pathway. As can be seen, when the PPM is incorporated, our network can more accurately locate the salient objects, even their boundaries. On the contrary, when removing the PPM, the location information of salient objects loses a lot. It demonstrates that leveraging PPM is indeed helpful for segmenting the complete salient objects due to its effective way of increasing the receptive field of our network.

**GGM & FAMs.** By introducing both GGM and FAMs into the FPN baseline (the 6th row in Table 1), the performance compared

| No. | FAM+ | | | | PASCAL-S [74] | | | DUT-OMRON [75] | | | HKU-IS [29] | | | DUTS-TE [76] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\mathcal{P}^1$ | $\mathcal{P}^2$ | $\mathcal{P}^4$ | $\mathcal{P}^8$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ |
| 1 | ✓ | | | | 0.856 | 0.075 | 0.848 | 0.799 | 0.062 | 0.810 | 0.925 | 0.038 | 0.907 | 0.860 | 0.047 | 0.865 |
| 2 | ✓ | ✓ | | | 0.866 | 0.071 | 0.853 | 0.806 | 0.061 | 0.819 | **0.931** | **0.035** | 0.910 | 0.873 | **0.043** | 0.874 |
| 3 | ✓ | ✓ | ✓ | | **0.872** | **0.070** | 0.855 | 0.816 | **0.059** | 0.824 | **0.931** | **0.035** | 0.913 | 0.873 | **0.043** | 0.874 |
| 4 | ✓ | ✓ | ✓ | ✓ | **0.872** | **0.070** | **0.857** | **0.817** | **0.059** | **0.832** | **0.931** | **0.035** | **0.914** | **0.878** | **0.043** | **0.880** |

Table 2: Ablation analysis on the importance of each sub-branch in FAM. $\mathcal{P}^i$ denotes average pooling layer of kernel size $i \times i$ and stride $i$, aside from $\mathcal{P}^1$, which denotes the identity mapping sub-branch. The best performance in each column is highlighted in **bold**.

| No. | Pooling Type | | PASCAL-S [74] | | | DUT-OMRON [75] | | | HKU-IS [29] | | | DUTS-TE [76] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GGM | FAM+ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ |
| 1 | Max | Max | 0.868 | 0.069 | 0.852 | 0.812 | 0.061 | 0.823 | **0.931** | **0.035** | **0.915** | 0.869 | 0.046 | 0.865 |
| 2 | Max | Avg | 0.868 | 0.070 | 0.857 | 0.812 | 0.063 | 0.818 | 0.930 | **0.035** | **0.915** | 0.872 | 0.045 | 0.871 |
| 3 | Avg | Max | **0.874** | **0.068** | **0.859** | 0.815 | 0.061 | 0.821 | **0.931** | **0.035** | 0.914 | 0.874 | 0.044 | 0.873 |
| 4 | Avg | Avg | 0.872 | 0.070 | 0.857 | **0.817** | **0.059** | **0.832** | **0.931** | **0.035** | 0.914 | **0.878** | **0.043** | **0.880** |

Table 3: Ablation analysis on the impact of different types of pooling operations used in GGM and FAM+. As can be observed, when using the average pooling operations in both GGM and FAM+, better overall performances can be achieved. The best performance in each column is highlighted in **bold**.

to the models with either GGM or FAMs incorporated can be further enhanced in all terms of F-measure, MAE, and S-measure. This phenomenon demonstrates that our GGM and FAM are two complementary modules. On the one hand, the utilization of the proposed GGM allows our approach to possess a strong capability of accurately discovering salient objects and keep the detected objects more integral. On the other hand, our FAMs can help refine the details of the discovered salient objects. As illustrated in Fig. 3, by comparing Column g and Column d, we can observe that adding GGM can locate the salient regions more precisely. By comparing Column g and Column f, our approach with both GGM and FAMs can capture more detailed information about the boundaries of salient objects. More qualitative results can be found in Fig. 9.

**FAM+ *v.s.* FAM.** As shown in Table 5, when taking the VGG-16 as the backbone, our network with GGM and FAMs has already achieved better performance than previous state-of-the-art methods. Here, we show that simply adjusting the structure of our original FAM by introducing internal communications, as described in Sec. 3.4 can further boost the performance. Table 1 shows the results when replacing the original FAM with our newly proposed FAM+. By comparing results from the 7th *v.s.* 5th rows, or the 8th *v.s.* 6th rows, we can easily observe that the utilization of FAM+ leads to steady improvements on all four datasets and both circumstances, either with GGM incorporated or not, respectively. Specifically, the results on the challenging PASCAL-S and DUTS-TE datasets can be boosted by $\sim 1\%$ in terms of F-measure and S-measure. It reflects that building internal communications within the original FAM does matter for achieving higher performance.

### 5.2.2 Ablation on GGM

To better understand the constitution of our proposed GGM, we perform two ablation experiments, which correspond to the 2nd and 3rd rows in Table 1, respectively. We first remove the pyramid pooling module while connecting the feature map $C_5$ to each pyramid level in the top-down pathway (*i.e.*, keeping the global guiding flows only). This operation degrades the performance of our approach by more than 2% in F-measure (Row 3 *v.s.* Row 4). Furthermore, we attempt to drop out all the global guiding flows by directly connecting the pyramid pooling module to $C_5$ and building the FPN. This modification makes the performance decline as well compared to the results with the entire GGM considered (Row 2 *v.s.* Row 4). In Fig. 6, we also show some visualizations of the feature maps outputted by the last layer of the bottom-up pathway. Apparently, the pyramid pooling module is capable of better capturing the locations of salient objects and guaranteeing their integrity. These experiments indicate that both the pyramid pooling module and global guidance flows play important roles in our GGM, and the absence of either of them is harmful to the performance of our approach.

### 5.2.3 Ablation on FAM+

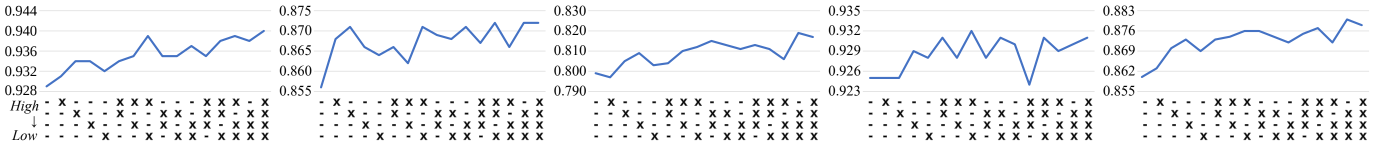
As described above, FAM+ is an effective tool to reduce the aliasing effect caused by up-sampling, especially when the rate is large (*e.g.*, 4 or 8), and meanwhile, to further enlarge the receptive field of the model. In our default setting, we adopt three different down-sampling rates (*i.e.*, 2, 4, and 8), as shown in Fig. 4. We set the largest kernel size of the pooling layers to $8 \times 8$ by considering the sizes of $C_5$ and the feature map from the last layer of the backbone. It has been explained that the FAM+ is designed to smooth the stride gaps when fusing feature maps with different resolutions. To demonstrate that the three scales we adopt are necessary, we conduct a series of ablation experiments. As shown in the first three rows of Table 2, when we gradually increase the number of pooling sub-branches with larger down-sampling rates in FAM+ (with the identity mapping sub-branch unchanged), the performances on four datasets vary with an overall increasing trending. When all the pooling sub-branches are combined (the last row in Table 2), the results can be maximized. We can conclude from the above analysis that richer combinations of down-sampling rates in FAM+ usually bring better overall performances and robustness across datasets. It also proves the effectiveness of integrating cross-scale feature representations.

### 5.2.4 Ablation on Pooling Operations

Pooling techniques play a fundamental role in the proposed method. Here, we investigate how different types of pooling operations and their combinations perform. We firstly focus on two basic yet most common types of pooling operations: average

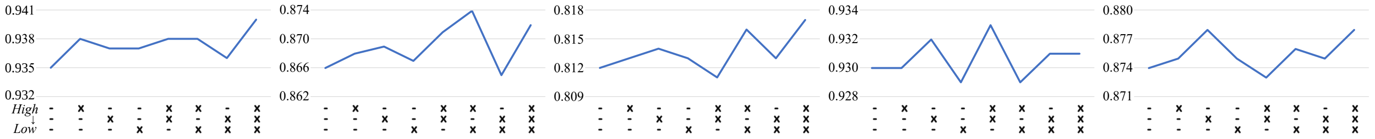
| No. | Pooling Type | FPS | PASCAL-S [74] | | | DUT-OMRON [75] | | | HKU-IS [29] | | | DUTS-TE [76] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ |
| 1 | Avg | 48 | 0.872 | 0.070 | 0.857 | 0.817 | 0.059 | 0.832 | 0.931 | 0.035 | 0.914 | 0.878 | 0.043 | 0.880 |
| 2 | Mixed [55] | 29 | 0.872 | 0.070 | 0.859 | 0.816 | 0.060 | 0.835 | 0.932 | 0.034 | 0.915 | 0.873 | 0.044 | 0.878 |
| 3 | Gated [55] | 31 | 0.876 | **0.068** | **0.862** | 0.821 | 0.060 | 0.835 | 0.934 | 0.034 | 0.915 | 0.875 | 0.045 | 0.879 |
| 4 | Tree [55] | 7 | **0.880** | **0.068** | **0.862** | 0.818 | 0.060 | 0.834 | 0.934 | 0.034 | 0.916 | 0.878 | 0.043 | 0.880 |
| 5 | Lossless [56] | 30 | 0.852 | 0.079 | 0.841 | 0.792 | 0.069 | 0.801 | 0.922 | 0.040 | 0.904 | 0.857 | 0.050 | 0.859 |
| 6 | LIP [57] | 32 | 0.873 | 0.070 | 0.859 | 0.825 | 0.059 | **0.836** | **0.935** | **0.033** | **0.918** | 0.878 | 0.043 | 0.881 |
| 7 | Strip [58] | 27 | 0.879 | 0.069 | 0.858 | **0.830** | **0.056** | 0.833 | **0.935** | 0.034 | 0.916 | **0.885** | **0.041** | **0.882** |

Table 4: Ablation analysis on the impact of cooperating existing smarter pooling operations. The basic average pooling operation shows a good trade-off between effectiveness and efficiency. The best performance in each column is highlighted in **bold**.



(a) ECSSD [79] (b) PASCAL-S [74] (c) DUT-OMRON [75] (d) HKU-IS [29] (e) DUTS-TE [76]

Fig. 7: Ablation analysis on how different combinations of FAM+s influence the performances. The vertical axes represent the F-measure values, and the horizontal axes show the combination of FAM+s on different positions. In the horizontal axes, 'x' means a FAM+, and '-' means two $3 \times 3$ convolution layers. There are four different positions in total, ranging from high- to low-levels.



(a) ECSSD [79] (b) PASCAL-S [74] (c) DUT-OMRON [75] (d) HKU-IS [29] (e) DUTS-TE [76]

Fig. 8: Ablation analysis on how different combinations of GGFs influence the performances. The vertical axes represent the F-measure values, and the horizontal axes show the combination of GGFs leading to different stages. In the horizontal axes, 'x' means a GGF, and '-' means no connection. There are three stages in total, ranging from high- to low-levels.

pooling and max pooling. We attempt to replace all the adaptive average pooling operators in GGM with adaptive max pooling operators and (or) all the average pooling operators in FAM+ with max pooling operators to see the performance difference. We keep all other configurations unchanged and show the results in Table 3. As can be seen from the 1st *v.s.* 4th rows, using the max pooling operators in both the GGM and FAM+ modules, yields an average performance decrease of about 0.7% in terms of S-measure across four datasets. The decreases in the challenging DUT-OMRON and DUTS-TE datasets are especially evident. Replacing either one of the average pooling operators in the GGM and FAM+ modules with max pooling operators causes performance drops in different degrees. By comparing the 3rd *v.s.* 2nd rows, we can see that changing for average pooling in the GGM module brings more benefits than the FAM+ module. Generally, using average pooling operators in both the GGM and FAM+ modules achieves the best overall results. We argue that the above phenomena may be because, unlike max pooling, average pooling can better capture local contextual information as it builds connections among locations within the whole pooling window.

Considering that various smarter pooling operations have been proposed, we also conduct experiments that cooperate our method with them in Table 4. The 2nd-4th rows are from methods that use different strategies to combine the average and max pooling operations. By comparing them with the 1st row, we find that more complicated structures do not necessarily mean better performance. Though the most complicated tree-structured pooling operation performs slightly better, it slows the speed dramatically by 84%. Similar phenomena occur with the 5th and 6th rows, which use adaptive pixel-level strategies. We can find that the top six methods provide the same receptive field sizes as their pooling windows are of the same shapes. Differently, Strip pooling [58] (last row) achieves better performance by using a band shape pooling window to perform average pooling, which enlarges the receptive field size though sacrificing some efficiency. From the above analysis, we argue that average pooling is sufficient and more efficient when having the same pooling window sizes. However, selecting and adjusting the pooling window sizes to achieve a better trade-off between accuracy and efficiency remains to be studied.

### 5.2.5 Different Combinations of FAM+s and GGFs

As illustrated in Fig. 2, in the proposed PoolNet+, we use a FAM+ to aggregate the feature maps with different receptive fields at each stage in the top-down pathway. Additionally, the global information collected by the PPM is guided into the above feature aggregation processes with a series of GGFs. By far, we treat all the FAM+s placed at different positions as a whole, so are the GGFs. To understand how each FAM+ (GGF) and different combinations of FAM+s (GGFs) influence the performance and analyze their universality across different datasets, we decompose the FAM+s (GGFs) and carry out a series of ablation experiments in this subsection.

| Method | Params | MAdds | ECSSD [79] | | | PASCAL-S [74] | | | DUT-OMRON [75] | | | HKU-IS [29] | | | DUTS-TE [76] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (M) | (G) | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $S_m \uparrow$ |
| **VGG-16 backbone** | | | | | | | | | | | | | | | | | |
| DCL$_{16}$ [41] | 66.25 | - | 0.896 | 0.080 | 0.869 | 0.805 | 0.115 | 0.800 | 0.733 | 0.094 | 0.762 | 0.893 | 0.063 | 0.871 | 0.786 | 0.081 | 0.803 |
| RFCN$_{16}$ [19] | - | - | 0.898 | 0.097 | 0.856 | 0.827 | 0.118 | 0.808 | 0.747 | 0.094 | 0.774 | 0.895 | 0.079 | 0.860 | 0.786 | 0.090 | 0.793 |
| SBF$_{17}$ [81] | 93.90 | 15.90 | 0.855 | 0.092 | 0.830 | 0.763 | 0.133 | 0.758 | 0.687 | 0.109 | 0.748 | - | - | - | - | - | - |
| WSS$_{17}$ [76] | 14.70 | 15.40 | 0.855 | 0.106 | 0.806 | 0.771 | 0.140 | 0.740 | 0.694 | 0.110 | 0.726 | 0.862 | 0.079 | 0.819 | 0.740 | 0.099 | 0.743 |
| MSR$_{17}$ [11] | - | - | 0.903 | 0.059 | 0.887 | 0.839 | 0.083 | 0.835 | 0.790 | 0.073 | 0.805 | 0.907 | 0.043 | 0.896 | 0.824 | 0.062 | 0.834 |
| DSS$_{17}$ [10] | 62.23 | 52.20 | 0.906 | 0.064 | 0.880 | 0.821 | 0.101 | 0.804 | 0.760 | 0.074 | 0.789 | 0.900 | 0.050 | 0.881 | 0.813 | 0.065 | 0.826 |
| NLDF$_{17}$ [20] | 35.48 | - | 0.903 | 0.065 | 0.870 | 0.822 | 0.098 | 0.805 | 0.753 | 0.079 | 0.770 | 0.902 | 0.048 | 0.878 | 0.816 | 0.065 | 0.816 |
| Amulet$_{17}$ [21] | 33.16 | 20.70 | 0.911 | 0.062 | 0.876 | 0.826 | 0.092 | 0.816 | 0.737 | 0.083 | 0.784 | 0.889 | 0.052 | 0.866 | 0.773 | 0.075 | 0.800 |
| C2SNet$_{18}$ [82] | 137.05 | 20.50 | 0.910 | 0.055 | 0.894 | 0.842 | 0.082 | 0.836 | 0.757 | 0.072 | 0.798 | 0.896 | 0.048 | 0.883 | 0.807 | 0.062 | 0.828 |
| PAGR$_{18}$ [16] | - | - | 0.924 | 0.064 | 0.883 | 0.847 | 0.089 | 0.822 | 0.771 | 0.071 | 0.775 | 0.919 | 0.047 | 0.889 | 0.854 | 0.055 | 0.839 |
| RAS$_{18}$ [45] | 20.23 | 15.90 | 0.918 | 0.059 | 0.888 | 0.829 | 0.101 | 0.799 | 0.786 | 0.062 | 0.814 | 0.913 | 0.045 | 0.887 | 0.831 | 0.059 | 0.839 |
| BMPM$_{18}$ [23] | - | - | 0.926 | 0.048 | 0.905 | 0.854 | 0.074 | 0.845 | 0.793 | 0.063 | 0.809 | 0.922 | 0.039 | 0.907 | 0.854 | 0.048 | 0.862 |
| JDFPR$_{19}$ [83] | 87.61 | - | 0.925 | 0.052 | 0.902 | 0.854 | 0.082 | 0.841 | 0.802 | **0.057** | 0.821 | 0.920 | 0.039 | 0.903 | 0.833 | 0.058 | 0.836 |
| PAGE$_{19}$ [84] | - | - | 0.928 | 0.046 | 0.906 | 0.848 | 0.076 | 0.842 | 0.791 | 0.062 | 0.825 | 0.920 | 0.036 | 0.904 | 0.838 | 0.051 | 0.855 |
| AFNet$_{19}$ [46] | 25.78 | - | 0.932 | 0.045 | 0.907 | 0.861 | **0.070** | 0.849 | **0.817** | 0.058 | 0.825 | 0.926 | 0.036 | 0.906 | 0.867 | 0.045 | 0.867 |
| **PoolNet-V** | 52.51 | 48.81 | 0.935 | 0.046 | 0.909 | 0.866 | 0.075 | 0.849 | 0.813 | 0.060 | 0.830 | 0.927 | 0.036 | 0.908 | 0.870 | 0.045 | 0.871 |
| **PoolNet-V+** | 26.31 | 27.51 | **0.940** | **0.044** | **0.914** | **0.872** | **0.070** | **0.857** | **0.817** | 0.059 | **0.832** | **0.931** | **0.035** | **0.914** | **0.878** | **0.043** | **0.880** |
| **ResNet-50 backbone** | | | | | | | | | | | | | | | | | |
| SRM$_{17}$ [40] | 53.14 | - | 0.916 | 0.056 | 0.891 | 0.838 | 0.084 | 0.834 | 0.769 | 0.069 | 0.798 | 0.906 | 0.046 | 0.887 | 0.826 | 0.058 | 0.836 |
| DGRL$_{18}$ [12] | - | - | 0.921 | 0.043 | 0.899 | 0.844 | 0.072 | 0.836 | 0.774 | 0.062 | 0.806 | 0.910 | 0.036 | 0.895 | 0.828 | 0.049 | 0.842 |
| PiCANet$_{18}$ [17] | 47.22 | 54.06 | 0.932 | 0.048 | 0.912 | 0.864 | 0.075 | 0.854 | 0.820 | 0.064 | 0.830 | 0.920 | 0.044 | 0.904 | 0.863 | 0.050 | 0.868 |
| ICTB$_{19}$ [48] | - | - | 0.935 | 0.045 | 0.912 | 0.855 | 0.071 | 0.850 | 0.811 | 0.060 | 0.837 | 0.925 | 0.037 | 0.909 | 0.855 | 0.043 | 0.865 |
| CPD$_{19}$ [43] | 47.85 | - | 0.936 | 0.042 | 0.913 | 0.859 | 0.071 | 0.848 | 0.796 | 0.056 | 0.825 | 0.925 | 0.034 | 0.907 | 0.865 | 0.043 | 0.869 |
| CSNet$_{20}$ [85] | 36.37 | 11.75 | 0.940 | 0.041 | 0.914 | 0.866 | 0.073 | 0.851 | 0.821 | **0.055** | 0.831 | 0.930 | 0.033 | 0.911 | 0.881 | 0.040 | 0.879 |
| **PoolNet-R** | 68.26 | 38.19 | 0.940 | 0.042 | 0.914 | 0.863 | 0.075 | 0.849 | 0.830 | **0.055** | 0.834 | 0.934 | **0.032** | 0.917 | 0.886 | 0.040 | 0.883 |
| **PoolNet-R+** | 34.12 | 14.03 | **0.949** | **0.040** | **0.925** | **0.879** | **0.068** | **0.864** | **0.831** | 0.056 | **0.842** | **0.941** | 0.034 | **0.921** | **0.894** | **0.039** | **0.890** |
| **MobileNetV2 backbone** | | | | | | | | | | | | | | | | | |
| **PoolNet-M** | 3.00 | 1.20 | 0.932 | **0.048** | 0.902 | 0.847 | 0.083 | 0.835 | 0.818 | 0.058 | 0.821 | 0.924 | 0.038 | 0.902 | 0.866 | **0.046** | 0.862 |
| **PoolNet-M+** | 3.00 | 1.20 | **0.938** | **0.048** | **0.909** | **0.864** | **0.078** | **0.844** | **0.830** | **0.056** | **0.830** | **0.930** | **0.037** | **0.909** | **0.872** | **0.046** | **0.868** |

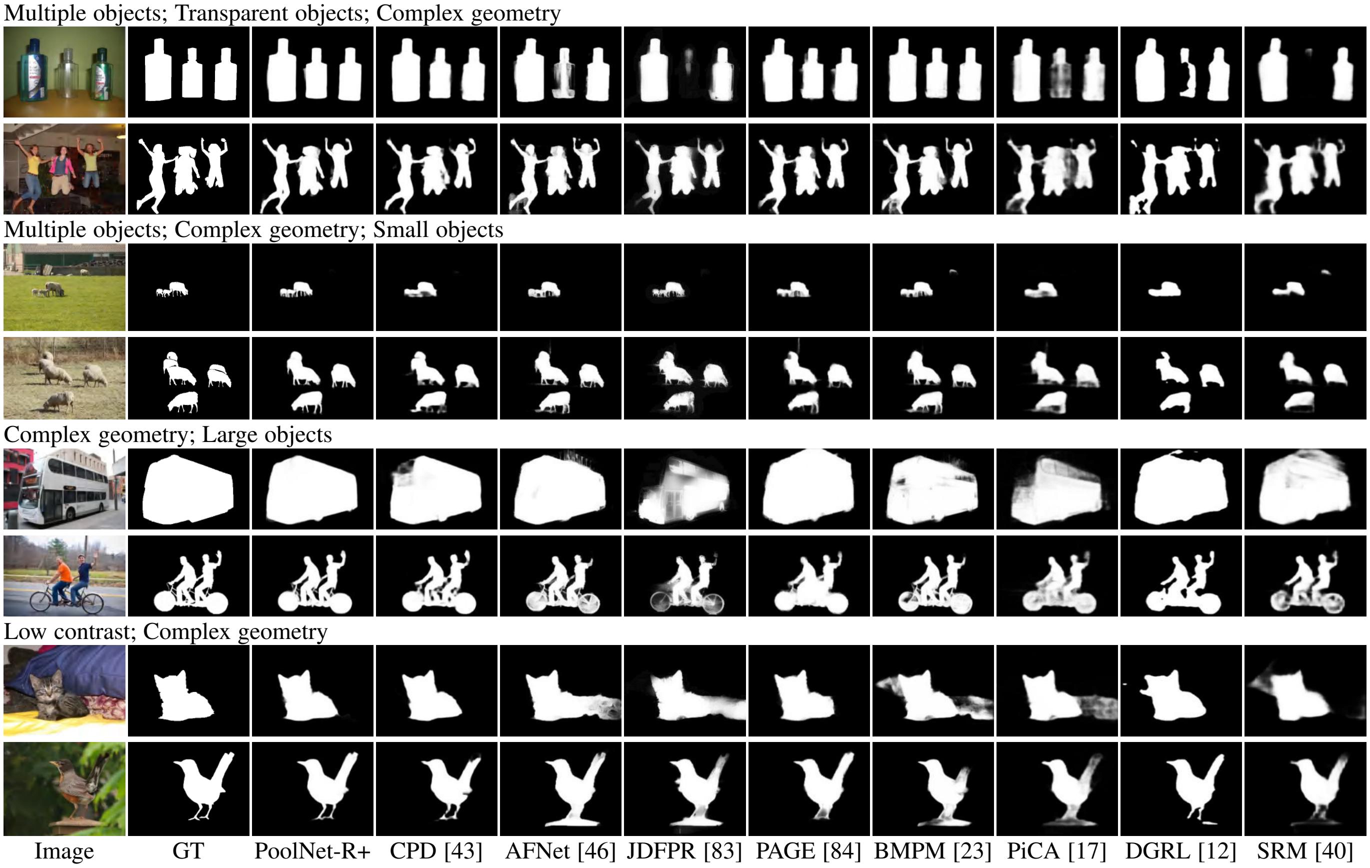
Table 5: Quantitative salient object detection results on five widely used datasets. The best results with different backbones are highlighted in **bold**, respectively. As can be seen, our approach achieves the best results on nearly all datasets and metrics.

**FAM+s.** When basing on the VGG-16 network, there are four suitable positions to place FAM+, resulting in 16 different combinations in total. For better illustration, we plot the relations between different combinations of FAM+s and the corresponding F-measure scores on five datasets in Fig. 7. The overall trending on most of the datasets is that more FAM+s have better average performances. We also observe that when there is only one FAM+, it is better to put it on either of the two middle stages rather than on the highest or lowest stage. Interestingly, if there are two FAM+s, a more appropriate solution is to place them at the highest and lowest stages, respectively.

We can also observe that the ECSSD and HKU-IS datasets are more sensitive to the absence of FAM+ at the lowest stage while the PASCAL-S, DUT-OMRON, and DUTS-TE datasets are more sensitive to the second-highest stage instead. We conclude that most of the samples in the ECSSD and HKU-IS datasets have only one small salient object. It is vital to narrow the gap between the local contextual information and global guidance information as the latter may lose the location information of the salient object due to large-scale down-sampling. Conversely, the distributions of the PASCAL-S, DUT-OMRON, and DUTS-TE datasets are more closer to the real world, hence having more big-sized samples. In that case, FAM+ at a higher stage can more effectively enlarge the network's overall receptive field and help better locate the salient objects.

**GGFs.** There are three suitable locations where GGFs can be connected, leading to a total of eight possible circumstances. We also plot the relations between different combinations of GGFs and the corresponding F-measure scores on five datasets in Fig. 8. From the curves, we can see that the performances are roughly the same on the ECSSD, PASCAL-S, and DUT-OMRON datasets when no or only one GGF is introduced. In most cases, if one of the three GGFs is removed from the model, the performances drop more or less. Especially when the GGF to the highest stage is removed, the performances on four datasets decrease dramatically. The ECSSD and PASCAL-S datasets are most sensitive to the absence of the GGF to the highest stage, while the DUT-OMRON and DUTS-TE datasets are more sensitive to the absence of the GGF to the lowest stage. And the HKU-IS dataset is more likely to be influenced by the GGF to the middle stage. The above phenomena show that the global information vanishing problem has different manifestations when targeting different datasets.

Generally, when preserving all the FAM+s and GGFs in the network, it has the most stable and robust performances across all datasets. We hope the above analysis can help the researcher with more insights on designing network structures that can be widely applied to datasets of various distributions.



Multiple objects; Transparent objects; Complex geometry

Multiple objects; Complex geometry; Small objects

Complex geometry; Large objects

Low contrast; Complex geometry

Fig. 9: Qualitative comparisons to previous state-of-the-art methods. Compared to other methods, our approach is capable of not only locating the integral salient objects but also well refining the details of the detected salient objects. It makes our resulting saliency maps very close to the ground-truth annotations.

### 5.3 Comparisons to the State-of-the-Arts

In this subsection, we compare our proposed PoolNet+ with 21 previous state-of-the-art and recent real-time methods. For fair comparisons, the saliency maps produced by these methods are generated by the original code released by the corresponding authors or directly provided by them. Moreover, all results are directly from the single-model testing without relying on any post-processing tools. All the predicted saliency maps are evaluated with the same evaluation code and environment.

#### 5.3.1 Quantitative Comparisons

In this part, we quantitatively compare our approach with previous state-of-the-art methods. The results can be found in Table 5. We report results on the VGG-16 [71], ResNet-50 [70], and MobileNetV2 [73] networks. From Table 5, we can observe that our original version of models (*i.e.*, PoolNet-V and PoolNet-R) already outperform almost all previous state-of-the-art methods on most of the datasets when depending on the same backbone networks. To be specific, when the ResNet-50 backbone is used, our PoolNet-R improves the state-of-the-art method CSNet [85] in both F-measure and S-measure on the DUT-OMRON, HKU-IS, and DUTS-TE datasets. Our VGG-16 version PoolNet (PoolNet-V) also performs better than AFNet, which takes the VGG-16 network as the backbone. When taking the newly proposed FAM+ into account, our improved version PoolNet+ achieves even better performance under both VGG-16 and ResNet-50, setting new state-of-the-art results on almost all datasets. Compared to our ResNet-50 version PoolNet-R+ (with 34.1M parameters and 14.0G MAdds), the light-weighted version PoolNet-M+ contains merely 3.0M parameters and 1.2G MAdds (less than 10%) but still achieves good results. Moreover, as shown in Table 5, results by PoolNet-M+ on five datasets are better than those produced by heavy models, such as ICTB [48] and CPD [43], which rely on the ResNet-50 backbone. It demonstrates that with a large amount of parameters and MAdds reduction, our PoolNet-M+ not only runs at a very fast speed but also achieves better results than most of the heavy models that exploit more powerful classification networks as backbones.

#### 5.3.2 Visual Comparisons

To further explain the advantages of our approach, we show some qualitative results produced by PoolNet-R+ and other previous state-of-the-art methods in Fig. 9. Each image is associated with different properties, including transparent objects, multiple objects, small objects, large objects, complex geometry, and low contrast, as done in [10]. Our goal is to demonstrate that our approach can work more robustly and better under different circumstances. It can be easily seen that our approach not only highlights the right salient objects but also maintains their sharp boundaries in almost all circumstances. The other methods, however, sometimes fail when dealing with complex scenes, especially when the salient objects are with complex geometry (the 6th row in Fig. 9). It is mainly because our GGM can more precisely locate the salient objects while FAM+ can better fuse features at different scales,

| | **PoolNet-M+** | **PoolNet-R+** | CPD [43] | AFNet [46] | PAGE [84] |
|---|---|---|---|---|---|
| Size | 300 × 400 | 300 × 400 | 352 × 352 | 224 × 224 | 224 × 224 |
| FPS | 66 | 53 | 27 | 31 | 25 |
| | PiCANet [17] | SRM [40] | Amulet [21] | DGRL [12] | NLDF [20] |
| Size | 224 × 224 | 353 × 453 | 256 × 256 | 384 × 384 | 300 × 400 |
| FPS | 8 | 16 | 20 | 8 | 12 |
| | DSS [10] | RAS [45] | C2SNet [82] | WSS [76] | SBF [81] |
| Size | 300 × 400 | 300 × 400 | 300 × 400 | 300 × 400 | 300 × 400 |
| FPS | 12 | 39 | 32 | 52 | 36 |

Table 6: Average speed (FPS) comparisons among PoolNet-M+, PoolNet-R+, the previous state-of-the-art, and recent real-time methods. Because of the efficient pooling techniques, our approachs run much faster than all other methods when having similar numbers of parameters and MAdds. Notice that PoolNet-M+ achieves significantly better results than other alternative methods while only needs less than 10% computational resources.

and thus the main parts and details of salient objects can be well captured.

### 5.3.3 Speed Analysis

Average speed (FPS) comparisons with the previous state-of-the-art and recent real-time salient object detection methods (tested in the same environment) are reported in Table 6. Compared to the previous fastest approach, WSS [76], which has a running speed of 52 FPS when the input images are of $300 \times 400$ resolution, our heavy PoolNet-R+ achieves a comparable running speed (53 FPS) when tested on images of the same resolution. But PoolNet-R+ performs dramatically better than WSS on all five datasets. Even compared to the previous best-performing approach with lower running speed, *e.g.*, CPD [43], PoolNet-R+ still gets better results in terms of both performance and speed, as shown in Table 5. The data in Table 6 shows that PoolNetR+ sometimes has a larger number of parameters but requires less computational resources (MAdds) and runs faster. Moreover, our light-weighted version, PoolNet-M+, is even more efficient with less than 20% parameters and 10% MAdds than to WSS and RAS but achieves much better performance, as shown in Table. 5. It can also be found that in Table 6 our PoolNet-M+ can run at a speed of 66 FPS when processing an image with a resolution of $300 \times 400$. These facts verify that our approach achieves the best results on salient object detection and runs at a very fast speed. It is primarily because our pooling-based designs make the following operators occupy less computational cost than the previous methods, as the feature maps are spatially down-sampled by a large scale, leading to substantial improvement.

# 6 Discussion

## 6.1 Reduction of Parameters and MAdds

A non-negligible drawback of the models proposed in our conference version [1] (*i.e.*, PoolNet-V and PoolNet-R) is their huge computational burdens. As shown in Table 5, both PoolNet-V and PoolNet-R have large amounts of parameters and MAdds. This subsection shows that more than half of the computational burdens can be seamlessly cut off without sacrificing the performances. We mainly base on two observations, 1) salient object detection is a low-level vision problem, so there is no need for an extremely diverse feature space, especially in deeper stages; 2) too many $3 \times 3$ convolutional layers can sometimes be redundant [70]. We take the PoolNet-R for example, which uses the ResNet-50 network as the backbone. To be more specific, we reduce the computational burden in the following three ways:

| | Version | Total | backbone | PPM | GGFs | FAMs | Others |
|---|---|---|---|---|---|---|---|
| #Params | Previous | 68.26 | 23.51 | 11.27 | 5.31 | 18.14 | 10.04 |
| (M) | Now | 34.12 | 23.51 | 1.31 | 0.20 | 6.20 | 2.90 |
| #MAdds | Previous | 37.58 | 6.24 | 2.37 | 12.76 | 1.65 | 14.56 |
| (G) | Now | 14.03 | 6.24 | 0.22 | 0.59 | 0.98 | 6.00 |

Table 7: Comparisons of network's composition of parameters and MAdds. We take the ResNet-50 network as the backbone for example. The compared two models include the one proposed in the previous conference version (PoolNet-R) and its corresponding computational burden reduced version.

- When building the feature pyramid, the channel numbers of the feature maps outputted by the intermediate stages of ResNet-50 are mapped from $\{128, 256, 256, 512, 512\}$ to $\{128, 128, 256, 256, 256\}$, respectively.
- The feature integration operation after the pooling sub-branches in PPM is changed from concat to element-wise summation.
- The kernel sizes of the last convolution layer in FAM+ and all convolution layers in GGFs are changed from $3 \times 3$ to $1 \times 1$.

The compositions of parameters and MAdds of PoolNet-R before and after the computational burden reduction process are listed in Table 7. As can be noticed, the model after computational burden reduction requires 50% and 63% fewer parameters and MAdds, respectively. The reduction rates are especially significant for the PPM and GGFs parts. It is worth noting that the performances of the two models listed in Table 7 are basically the same (an average fluctuation of $\sim 0.2\%$ in $F_\beta$). The above results indicate that the redundant computational costs can be effectively reduced regarding the target task's characteristics by carefully tailoring network structure.

## 6.2 Efficiency Analysis

This subsection analyzes the efficiency of the proposed PoolNet-R+ by decomposing its components and comparing them with the FPN baseline. Without loss of generality, the PoolNet-R+ can be decomposed into four parts: backbone, PPM, GGFs, FAM+s, and other essential components to build up the network. We build the FPN baseline to include the backbone and other essential components. The PPM and GGFs modules are excluded, and the FAM+s are replaced with two $3 \times 3$ convolution layers. As shown in Table 8, the 1st row is the FPN baseline. By comparing the 2nd and 3rd rows to the 1st row, we can find out that both the PPM and GGFs modules increase the number of parameters, MAdds, and inference latency slightly. However, adopting the FAM+s results in fewer MAdds (the 4th *v.s.* 1st rows), though more parameters are introduced, which indicates that more parameters do not necessarily mean more calculations.

An inevitable question that occurs simultaneously is why less amount of MAdds leads to more latency (the 2nd *v.s.* 3rd, 4th *v.s.* 1st rows)? We argue that it is related to the optimization and underlying implementation of the algorithms on different platforms. For instance, there are multiple parallel paths in both the PPM and FAM+ modules, which are performed, however,

| No. | FPN | PPM | GGFs | FAM+s | Params(M) | MAdds(G) | Latency(ms) |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | | | | 28.47 | 15.26 | 15.45 |
| 2 | ✓ | ✓ | | | $29.79_{+1.32}$ | $15.48_{+0.22}$ | $16.11_{+0.66}$ |
| 3 | ✓ | | ✓ | | $28.67_{+0.20}$ | $15.84_{+0.58}$ | $15.84_{+0.39}$ |
| 4 | ✓ | | | ✓ | $32.60_{+4.13}$ | $13.23_{-2.03}$ | $15.96_{+0.51}$ |
| 5 | ✓ | ✓ | ✓ | ✓ | $34.12_{+5.65}$ | $14.03_{-1.23}$ | $16.93_{+1.48}$ |

Table 8: Decomposition of each component's influence on the network's efficiency. We take the ResNet-50 network as the backbone for example. The models w/o FAM+s (Rows 1-3) use two $3 \times 3$ convolution layers instead. The subscripts in the last three columns represent the relative changes compared to the 1st row. We measure the MAdds and latency with an input tensor of shape $1 \times 3 \times 224 \times 224$ on a single RTX 2080Ti GPU.

serially on PyTorch. In FAM+, the input feature maps are firstly spatially down-sampled before being further processed. The core parts of FAM+ (operations before the last $3 \times 3$ convolution layer) require fewer MAdds even than a single $3 \times 3$ convolution layer. In general, the PoolNet-R+ (the last row) requires 8.1% less computational burden in theory compared to the FPN baseline, though with more modules introduced. The PoolNet-R+ also obtains dramatically better performances. Considering the above analysis, we expect the proposed PoolNet+ to have a faster running speed in the future with the help of more engineering optimization efforts.

### 6.3 Failure Case Analysis

We show some failure predictions of our approach in Fig. 10. Generally, these failure cases can be categorized into four circumstances. The first one is having complex backgrounds, as shown in the samples from the first two rows. The second circumstance is the low contrast between foreground and background, shown in the 2nd two rows. One common defect in the above two circumstances is that the salient object cannot be completely segmented out, in which some small parts of the salient object are missed. Another defect is that the main body of the salient object cannot be detected, or some non-salient regions are miss-predicted as salient. The third circumstance is occlusion, as shown in the 3rd two rows. In these cases, part of the salient object, especially around the regions being occluded, cannot be integrally extracted. The last type of failure cases is caused by transparent objects, as shown in the last two rows. Although our approach can detect some parts of the target transparent objects in most cases, it is still difficult to segment out the complete salient objects. In most of the above cases, it is hard to distinguish the boundaries between the foreground and background, even for humans.

To remedy the above problem, we argue that there are three possible ways. First of all, a promising solution is to enlarge the scale of the training dataset, which is straightforward as the CNN-based models learn all the knowledge from the dataset. If the model sees enough samples during the training phase, it will perform better when facing similar scenes. A large training set that includes as diverse scenes as possible and has a data distribution closer to the real world will always help. Secondly, including more prior knowledge on the segmentation level so that pixels with similar colors or textures can be detected together as a region. The characteristics of CNNs determine that the input image is processed pixel-wisely, where the learnable weights decide the correlations of two positions in the prediction map. The segment-level prior can alleviate the parts missing and blurring problems mentioned above as the similar pixels are correlated. It can also serve as a post-processing step to further refine the predicted maps. Designing more advanced models that have more powerful feature extracting capabilities can be another solution. More diverse and rich feature representations usually mean a higher possibility of correcting the mis-prediction made by previous models.

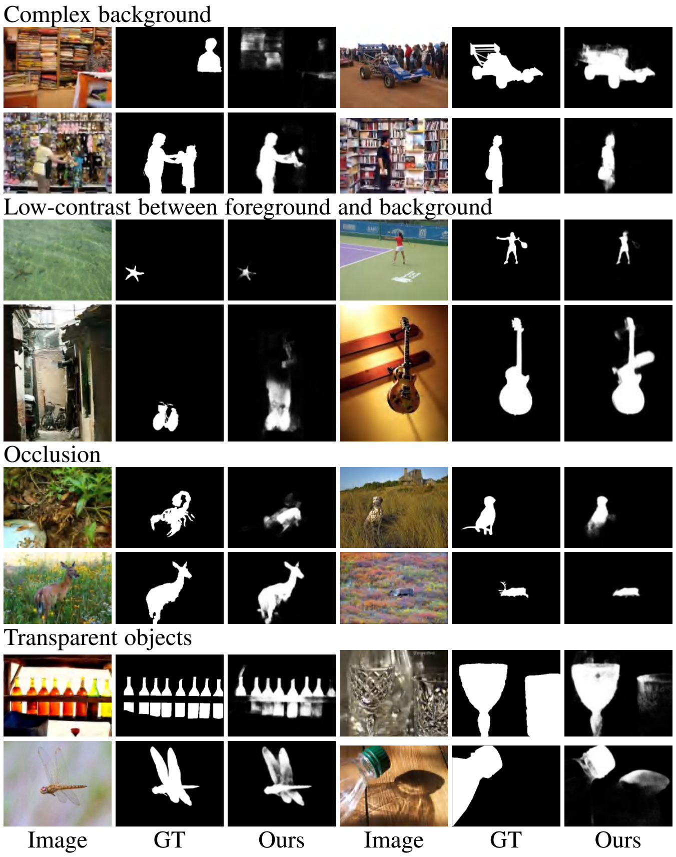


Fig. 10: Failure cases selected from multiple datasets. These failure cases can be categorized into four typical circumstances.

## 7 Generalization to Edge Detection

This section investigates the generalization ability of the proposed approach by applying it to another popular low-level vision task: edge detection. We compare our edge detection results with eight recent popular CNN-based methods focusing on edge detection.

### 7.1 Implementation Details

Except for the batch normalization layers, we apply the proposed PoolNet-R+ network for edge detection without modification. Similar to [69], we take input images of arbitrary sizes for both training and testing, and the training batch size is set to one. Also, a deep-supervision strategy is applied. We remove the batch normalization layers in PoolNet-R+ except for the ones in the backbone network (*i.e.*, ResNet-50), whose parameters are frozen during training and testing. We initialize the backbone parameters with the corresponding weights pre-trained on the ImageNet dataset and the rest randomly. The whole training period takes 12 epochs, and the initial learning rate is set to 5e-5, divided

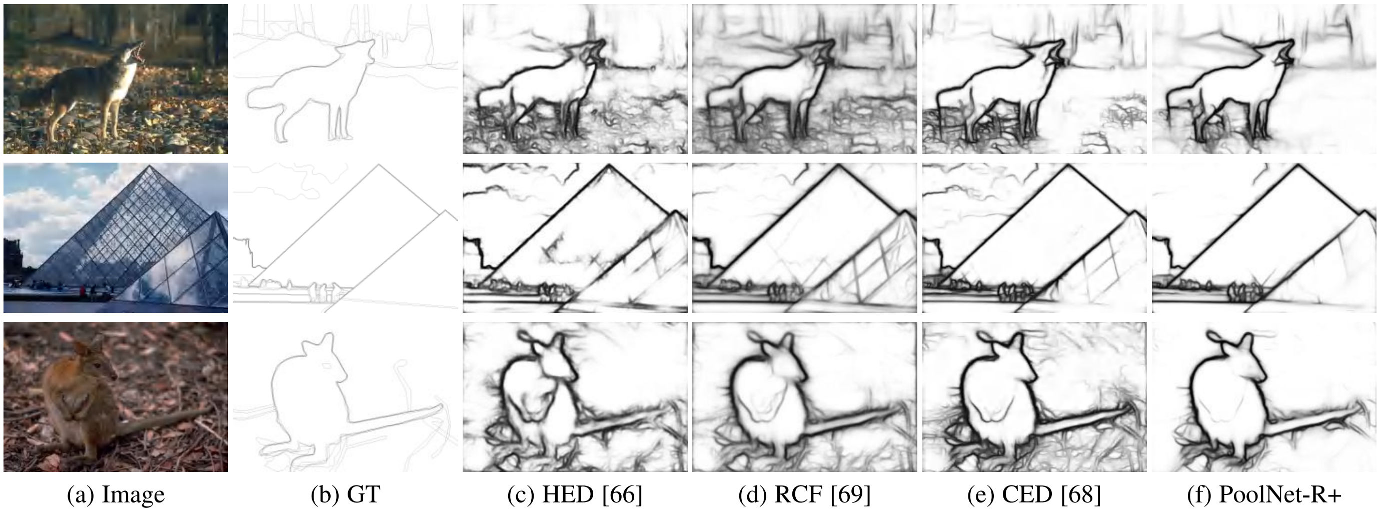


Fig. 11: Visual comparisons with several recent state-of-the-art edge detectors. As can be seen, our proposed approach can generate a cleaner background and capture weak object boundaries compared to the other three methods. This phenomenon is especially clear for the second image. All the images are from the BSDS 500 dataset [61].

| Method | ODS | OIS |
|---|---|---|
| DeepContour [65] | 0.756 | 0.773 |
| HED [66] | 0.788 | 0.808 |
| CEDN [87] | 0.788 | 0.804 |
| RDS [88] | 0.792 | 0.810 |
| COB [89] | 0.793 | 0.820 |
| DCNN+sPb [90] | 0.813 | 0.831 |
| RCF [69] | 0.811 | 0.830 |
| CED [68] | 0.815 | 0.833 |
| PoolNet-R+ | 0.819 | 0.834 |

Table 9: Quantitative comparison of our approach with existing edge detection methods.

by 10 after 9 epochs. We use the Adam [78] optimizer with a weight decay of 5e-4 for optimization. We train and evaluate our results on the BSDS 500 set [61], containing 200 training, 100 validation, and 200 testing images, each with accurately annotated boundaries. Besides, our training set also incorporates the images from the PASCAL Context Dataset [86] and performs data augmentation as in [66], [69] for fair comparisons. We use the fixed contour threshold (ODS) and per-image best threshold (OIS) for evaluation similar to previous work. Before evaluation, we apply the standard non-maximal suppression algorithm to get thinned edges.

### 7.2 Quantitative Comparisons

In Table 9, we show quantitative results by a series of recent CNN-based methods and ours. As can be seen, by simply applying PoolNet-R+ that is designed for salient object detection to edge detection, our edge results are better than most of the previous CNN-based models and are even comparable to the state-of-the-art model. It implies that PoolNet is also beneficial to the edge detection task. We want to emphasize that although the goal of designing PoolNet is to improve the performance of salient object detection, our ultimate model can also produce promising edge predictions. We further found that PoolNet can also be well generalized to other low-level visual tasks. We apply PoolNet to the RGB-D salient object detection and camouflaged object detection tasks where it also performs favorably. More details can be found in the supplementary material. It is expected that the progress we make in designing PoolNet is helpful for future research in other directions.

### 7.3 Visual Comparisons

In Fig. 11, we show some visual comparisons between PoolNet and other three popular methods, including HED [66], RCF [69] and CED [68]. Thanks to the powerful features learned by PoolNet, even without network modification, our approach still performs well in detecting the real boundaries of objects compared to the best method that is specifically tailored. As shown in column (f) in Fig. 11, PoolNet can give light predictions to the edges that are not the real boundaries of objects but concentrate more on the genuine object boundaries as more global information is lead in by the GGM module. We believe this characteristic can make our approach more helpful in real-world applications than other methods.

## 8 Conclusion

This paper explores the potential of efficient pooling techniques on salient object detection by designing two simple pooling-based modules. Considering the vital importance of precisely locating the salient objects, we design a global guidance module (GGM) to enlarge the valid receptive field of the bottom-up pathway and ensure the guiding role of the location information in the top-down pathway. An advanced feature aggregation module (FAM+) is further proposed to bridge the gap between the local contextual information and global guidance information. Extensive experiments on five popular salient object detection benchmarks demonstrate that the proposed method confidently outperforms the state-of-the-art methods. Furthermore, to meet the needs of extremely low computational overhead on mobile devices, we present a light-weighted version, called PoolNet-M+, which achieves good performance with $\sim 10\times$ fewer parameters and MAdds and runs even faster.

We carry out a series of carefully designed ablation experiments on multiple datasets from three aspects to understand how and why the proposed two modules work. First, we regard GGM and all the FAM+s as two integral parts to verify their influence from the network structure level. Since there are multiple positions in the network to place the proposed two modules, we then decompose the components in GGM and each FAM+ to validate their contributions from the module level. Finally, we compare different design choices of the two modules from the operation level. Along with the numbers and curves, we also visualize the intermediate feature maps under various circumstances to illustrate the influence intuitively.

We analyze the efficiency of the proposed method and show that more than half of the computational cost can be cut off without harming the performances. To demonstrate the generalization ability of the proposed structure, we apply it to three related and popular low-level vision tasks, including edge detection, RGB-D salient object detection, and camouflaged object detection. We show that with little modification, the proposed structure achieves substantial improvements over the state-of-the-art methods on the three tasks on multiple datasets, respectively. We hope our design principles and experiments could provide promising future research directions in salient object detection and other related vision tasks.

**Acknowledgments.** This research was supported by Major Project for New Generation of AI under Grant No. 2018AAA0100400 and NSFC (61922046).

## REFERENCES

[1] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, and J. Jiang, "A simple pooling-based design for real-time salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[2] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Int. Conf. Mach. Learn.*, 2015, pp. 597–606.

[3] W. Wang, J. Shen, and H. Ling, "A deep network solution for attention and aesthetics aware photo cropping," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1531–1544, 2018.

[4] M.-M. Cheng, F.-L. Zhang, N. J. Mitra, X. Huang, and S.-M. Hu, "Repfinder: finding approximately repeated scene elements for image editing," *ACM Trans. Graphics*, vol. 29, no. 4, p. 83, 2010.

[5] Y. Gao, M. Wang, D. Tao, R. Ji, and Q. Dai, "3-D object retrieval and recognition with hypergraph analysis," *IEEE Trans. Image Process.*, vol. 21, no. 9, pp. 4290–4303, 2012.

[6] W. Wang, J. Shen, R. Yang, and F. Porikli, "Saliency-aware video object segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 20–33, 2017.

[7] C. Craye, D. Filliat, and J.-F. Goudou, "Environment exploration for object-based visual saliency learning," in *ICRA*, 2016, pp. 2303–2309.

[8] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, J. Feng, Y. Zhao, and S. Yan, "Stc: A simple to complex framework for weakly-supervised semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2016.

[9] G. Sun, W. Wang, J. Dai, and L. Van Gool, "Mining cross-image semantics for weakly supervised semantic segmentation," in *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 347–365.

[10] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr, "Deeply supervised salient object detection with short connections," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 4, pp. 815–828, 2019.

[11] G. Li, Y. Xie, L. Lin, and Y. Yu, "Instance-level salient object segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2386–2395.

[12] T. Wang, L. Zhang, S. Wang, H. Lu, G. Yang, X. Ruan, and A. Borji, "Detect globally, refine locally: A novel approach to saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 3127–3135.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234–241.

[14] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

[15] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

[16] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang, "Progressive attention guided recurrent network for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 714–722.

[17] N. Liu, J. Han, and M.-H. Yang, "Picanet: Learning pixel-wise contextual attention for saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 3089–3098.

[18] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[19] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, "Saliency detection with recurrent fully convolutional networks," in *Eur. Conf. Comput. Vis.*, 2016.

[20] Z. Luo, A. K. Mishra, A. Achkar, J. A. Eichel, S. Li, and P.-M. Jodoin, "Non-local deep features for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

[21] P. Zhang, D. Wang, H. Lu, H. Wang, and X. Ruan, "Amulet: Aggregating multi-level convolutional features for salient object detection," in *Int. Conf. Comput. Vis.*, 2017.

[22] J.-J. Liu, Q. Hou, M.-M. Cheng, C. Wang, and J. Feng, "Improving convolutional networks with self-calibrated convolutions," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[23] L. Zhang, J. Dai, H. Lu, Y. He, and G. Wang, "A bi-directional message passing model for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 1741–1750.

[24] P. Zhang, D. Wang, H. Lu, H. Wang, and B. Yin, "Learning uncertain convolutional features for accurate saliency detection," in *Int. Conf. Comput. Vis.*, 2017, pp. 212–221.

[25] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu, "Global contrast based salient region detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 569–582, 2015.

[26] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, "Salient object detection: A discriminative regional feature integration approach," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 2083–2090.

[27] X. Li, H. Lu, L. Zhang, X. Ruan, and M.-H. Yang, "Saliency detection via dense and sparse reconstruction," in *Int. Conf. Comput. Vis.*, 2013, pp. 2976–2983.

[28] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 733–740.

[29] G. Li and Y. Yu, "Visual saliency based on multiscale deep features," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 5455–5463.

[30] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1265–1274.

[31] L. Gayoung, T. Yu-Wing, and K. Junmo, "Deep saliency with encoded low level distance map and high level features," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[32] S. He, R. W. Lau, W. Liu, Z. Huang, and Q. Yang, "Supercnn: A superpixelwise convolutional neural network for salient object detection," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 330–344, 2015.

[33] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, "Deep networks for saliency detection via local estimation and global search," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3183–3192.

[34] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech, "Unconstrained salient object detection via proposal subset optimization," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 5733–5742.

[35] J. Kim and V. Pavlovic, "A shape-based approach for salient object detection using deep learning," in *Eur. Conf. Comput. Vis.* Springer, 2016, pp. 455–470.

[36] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.

[37] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5706–5722, 2015.

[38] A. Borji, M.-M. Cheng, Q. Hou, H. Jiang, and J. Li, "Salient object detection: A survey," *Computational Visual Media*, pp. 1–34, 2019.

[39] W. Wang, Q. Lai, H. Fu, J. Shen, and H. Ling, "Salient object detection in the deep learning era: An in-depth survey," *arXiv preprint arXiv:1904.09146*, 2019.

[40] T. Wang, A. Borji, L. Zhang, P. Zhang, and H. Lu, "A stagewise refinement model for detecting salient objects in images," in *Int. Conf. Comput. Vis.*, 2017, pp. 4019–4028.

[41] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[42] Y. Zeng, P. Zhang, J. Zhang, Z. Lin, and H. Lu, "Towards high-resolution salient object detection," in *Int. Conf. Comput. Vis.*, 2019, pp. 7234–7243.

[43] Z. Wu, L. Su, and Q. Huang, "Cascaded partial decoder for fast and accurate salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[44] T. Zhao and X. Wu, "Pyramid feature attention network for saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[45] S. Chen, X. Tan, B. Wang, and X. Hu, "Reverse attention for salient object detection," in *Eur. Conf. Comput. Vis.*, 2018, pp. 234–250.

[46] M. Feng, H. Lu, and E. Ding, "Attentive feedback network for boundary-aware salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[47] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand, "Basnet: Boundary-aware salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[48] W. Wang, J. Shen, M.-M. Cheng, and L. Shao, "An iterative and cooperative top-down and bottom-up inference network for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[49] Z. Wu, L. Su, and Q. Huang, "Stacked cross refinement network for edge-aware salient object detection," in *Int. Conf. Comput. Vis.*, 2019, pp. 7264–7273.

[50] R. Wu, M. Feng, W. Guan, D. Wang, H. Lu, and E. Ding, "A mutual learning method for salient object detection with intertwined multi-supervision," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[51] W. Wang, J. Shen, X. Dong, A. Borji, and R. Yang, "Inferring salient objects from human fixations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 1913–1927, 2019.

[52] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *Adv. Neural Inform. Process. Syst.*, vol. 2, 1989.

[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[54] M. Ranzato, Y.-L. Boureau, Y. LeCun *et al.*, "Sparse feature learning for deep belief networks," *Adv. Neural Inform. Process. Syst.*, vol. 20, pp. 1185–1192, 2007.

[55] C.-Y. Lee, P. Gallagher, and Z. Tu, "Generalizing pooling functions in cnns: Mixed, gated, and tree," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 863–875, 2017.

[56] F. Toutounchi and E. Izquierdo, "Advanced super-resolution using lossless pooling convolutional networks," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1562–1568.

[57] Z. Gao, L. Wang, and G. Wu, "Lip: Local importance-based pooling," in *Int. Conf. Comput. Vis.*, 2019, pp. 3355–3364.

[58] Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, "Strip pooling: Rethinking spatial pooling for scene parsing," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[59] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, pp. 679–698, 1986.

[60] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 5, pp. 530–549, 2004.

[61] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.

[62] P. Dollar, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries," in *IEEE Conf. Comput. Vis. Pattern Recog.*, vol. 2. IEEE, 2006, pp. 1964–1971.

[63] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 8, pp. 1558–1570, 2015.

[64] Y. Ganin and V. Lempitsky, "Nˆ 4-fields: Neural network nearest neighbor fields for image transforms," in *ACCV*. Springer, 2014, pp. 536–551.

[65] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3982–3991.

[66] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Int. Conf. Comput. Vis.*, 2015, pp. 1395–1403.

[67] Z. Yu, C. Feng, M.-Y. Liu, and S. Ramalingam, "Casenet: Deep category-aware semantic edge detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 5964–5973.

[68] Y. Wang, X. Zhao, Y. Li, and K. Huang, "Deep crisp boundaries: From boundaries to higher-level tasks," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1285–1298, 2018.

[69] Y. Liu, M.-M. Cheng, X. Hu, J.-W. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1939 – 1946, 2019.

[70] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[71] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Int. Conf. Learn. Represent.*, 2015.

[72] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," in *Int. Conf. Learn. Represent.*, 2015.

[73] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 4510–4520.

[74] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 280–287.

[75] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3166–3173.

[76] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan, "Learning to detect salient objects with image-level supervision," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 136–145.

[77] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Adv. Neural Inform. Process. Syst.*, 2012.

[78] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Int. Conf. Learn. Represent.*, 2015.

[79] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1155–1162.

[80] M.-M. Cheng and D.-P. Fan, "Structure-measure: A new way to evaluate foreground maps," *Int. J. Comput. Vis.*, vol. 129, no. 9, pp. 2622–2638, 2021.

[81] D. Zhang, J. Han, and Y. Zhang, "Supervision by fusion: Towards unsupervised learning of deep salient object detector," in *Int. Conf. Comput. Vis.*, 2017, pp. 4048–4056.

[82] X. Li, F. Yang, H. Cheng, W. Liu, and D. Shen, "Contour knowledge transfer for salient object detection," in *Eur. Conf. Comput. Vis.*, 2018, pp. 355–370.

[83] Y. Xu, D. Xu, X. Hong, W. Ouyang, R. Ji, M. Xu, and G. Zhao, "Structured modeling of joint deep feature and prediction refinement for salient object detection," in *Int. Conf. Comput. Vis.*, 2019.

[84] W. Wang, S. Zhao, J. Shen, S. C. H. Hoi, and A. Borji, "Salient object detection with pyramid attention and salient edges," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[85] M.-M. Cheng, S.-H. Gao, A. Borji, Y.-Q. Tan, Z. Lin, and M. Wang, "A highly efficient model to study the semantics of salient object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.

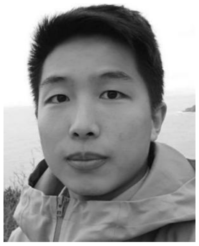
[86] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, "The role of context for object detection and semantic segmentation in the wild," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 891–898.

[87] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang, "Object contour detection with a fully convolutional encoder-decoder network," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 193–202.

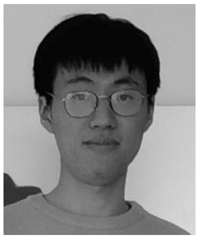
[88] Y. Liu and M. S. Lew, "Learning relaxed deep supervision for better edge detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 231–240.

[89] K.-K. Maninis, J. Pont-Tuset, P. Arbelaez, and L. Van Gool, "Convolutional oriented boundaries: From image segmentation to high-level tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.

[90] I. Kokkinos, "Pushing the boundaries of boundary detection using deep learning," *arXiv preprint arXiv:1511.07386*, 2015.



**Jiang-Jiang Liu** is currently a Ph.D. candidate with School of Computer Science, Nankai University, under the supervision of Prof. Ming-Ming Cheng. His research interests include deep learning, image processing, and computer vision.



**Qibin Hou** is an associate professor with Nankai University. He received his Ph.D. degree from School of Computer Science, Nankai University, under the supervision of Prof. Ming-Ming Cheng. Then he did two years research fellow working with Prof. Jiashi Feng at National University of Singapore. His research interests include deep learning and computer vision.

**Zhi-Ang Liu** received his B.S. degree from the School of Electrical Engineering and Automation, Harbin Institute of Technology in 2019. Currently, he is a master student in the College of Computer Science, Nankai University, supervised by Prof. Ming-Ming Cheng. His research interests include machine learning and computer vision.

**Ming-Ming Cheng** received his Ph.D. degree from Tsinghua University in 2012. Then he did two years research fellow, with Prof. Philip Torr in Oxford. He is now a professor at Nankai University, leading the Media Computing Lab. His research interests include computer graphics, computer vision, and image processing. He has published $100+$ refereed research papers, with $24,000+$ Google Scholar citations. He received research awards, including ACM China Rising Star Award, IBM Global SUR Award, *etc.* He is on the editor board of IEEE TPAMI and IEEE TIP.