REVISITING MULTI-LEVEL FEATURE FUSION: A SIMPLE YET EFFECTIVE NETWORK FOR SALIENT OBJECT DETECTION

Yu Qiu[1] *Yun Liu*[2] *Xiaoxu Ma*[2] *Lei Liu*[1] *Hongcan Gao*[2] *Jing Xu*[1]

[1]College of Artificial Intelligence, [2]College of Computer Science, Nankai University

ABSTRACT

It is widely accepted that the top sides of neural networks convey high-level semantic features and the bottom sides contain low-level details. Therefore, most of recent salient object detection models aim at designing effective fusion strategies for the side-output features of convolutional neural networks (CNNs). Although significant progress has been achieved in this direction, the network architectures become more and more complex, which will make the future improvement difficult and heavily engineered. Moreover, the manually designed fusion strategies would be sub-optimal due to the large search space of possible solutions. To address above problems, we propose an Automatic Top-Down Fusion (ATDF) model, in which the global information at the top sides are flowed into bottom sides to guide the learning of low layers. We design a novel module at each side to control the information flowed into a specific side, called *valve* module, by which each side is expected to receive the necessary top information. We perform extensive experiments to demonstrate that ATDF is simple yet effective and thus opens a new path for saliency detection. Code is available at `https://github.com/yun-liu/ATDF`.

Index Terms— Salient object detection, saliency detection, simple yet effective network, multi-level feature fusion

1. INTRODUCTION

Salient object detection aims at detecting the most conspicuous and eye-attracting regions in an image [1]. It can be used as a pre-processing step for many computer vision tasks, including visual tracking [2], image retrieval [3], and weakly supervised learning [4]. Although many methods have been proposed to push forward the state of the arts [5–16], it is still a challenging problem to accurately detect salient objects.

Recently, convolutional neural networks (CNNs), which can intelligently extract multi-level features directly from raw images, have significantly boosted the performance of saliency detection [6, 11, 13, 17]. Generally, the top layers of CNNs contain semantic information that can better locate where the salient objects are, while the bottom layers contain precise fine details such as object boundaries [13, 18]. Both the semantic information and fine details are important to saliency detection. Hence most existing methods mainly focus on how to integrate the multi-level side-output features using carefully designed connections [9, 12–14, 16, 17]. For examples, Hou *et al.* [13] fused several predictions, each of which was obtained by fusing some specific side outputs with short connections. However, their integration was not robust due to their manually selected connection combinations from lots of possible solutions. Moreover, Zhang *et al.* [9] first integrated multi-level feature maps into multiple resolutions and then adaptively learned to combine these feature maps at each resolution. However, it is difficult for the manually designed fusion strategies to take full advantage of the complementary top semantic information and bottom fine details.

Based on above analyses, we present an Automatic Top-Down Fusion (ATDF) model which is able to automatically flow the global information at the top sides of CNNs into bottom sides. Each side adds a novel *valve* module to receive the specifically useful and instructive global information to guide its learning. Hence the top semantic information can guide the learning of bottom layers, and the bottom side outputs can accurately predict both the location and details of salient objects. By this way, ATDF is able to take good advantage of the automatic fusion of multi-level features extracted from CNNs. Experiments on five challenging datasets demonstrate that our method favorably outperforms existing state-of-the-art methods in terms of popular evaluation metrics.

2. METHOD

2.1. Overall Framework

In this paper, we propose an Automatic Top-Down Fusion (ATDF) model to address the problems of sub-optimal multi-level side-output feature fusion in manually designed fusion strategies. As shown in Fig. 1, the main architecture of ATDF is beneficial from the encoder-decoder networks. Our model uses VGG16 net [19] as the backbone model by making two modifications: (i) We remove the final fully connected layers to serve for image-to-image translation; (ii) We remain the last pooling layer and meanwhile additionally introduce a max pooling layer (the top block in the left part of Fig. 1) to enlarge the receptive field and obtain global information.

The resulting VGG16 net (encoder) is composed of six

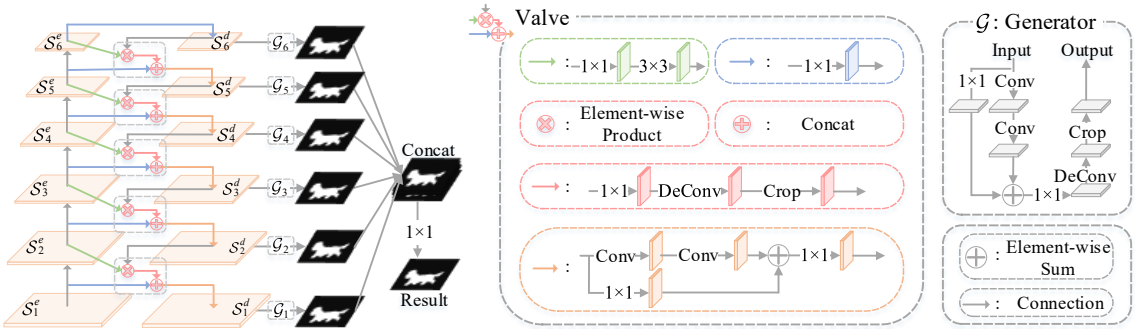

Fig. 1. The overall framework of our method. The **left part** represents the base network of ATDF, which builds a fully convolutional network (FCN) using VGG16 net [19]. The *valve* module in the **middle part** is designed to adaptively control the flow of semantic information. Moreover, we adopt deep supervision by adding side loss after the corresponding *generator* for each side output. 3×3 and 1×1 mean convolution layers with kernel sizes of 3×3 and 1×1, respectively.

blocks, which are denoted as $\{\mathcal{S}_1^e, \mathcal{S}_2^e, \mathcal{S}_3^e, \mathcal{S}_4^e, \mathcal{S}_5^e, \mathcal{S}_6^e\}$ from bottom to top, respectively. Note that $\mathcal{S}_6^e$ represents the added pooling layer. The receptive fields of $\mathcal{S}_1^e$ to $\mathcal{S}_6^e$ are gradually enlarged to capture more high-level information, so that $\mathcal{S}_6^e$ is considered to contain the most rich global semantic information. Symmetrically, the decoder is in a top-down view, which can be defined as the automatic top-down fusion path, containing six blocks, *i.e.* $\{\mathcal{S}_6^d, \mathcal{S}_5^d, \mathcal{S}_4^d, \mathcal{S}_3^d, \mathcal{S}_2^d, \mathcal{S}_1^d\}$. In the contracting path (encoder), the resolution of the feature map in each block is the half of the preceding one, and contrarily in the expanding path (decoder). In the expanding path, each feature map $\mathcal{S}_i^d$ originates from the preceding one $\mathcal{S}_{i+1}^d$ and the $\mathcal{S}_{i+1}^e$ and $\mathcal{S}_i^e$ in the contracting path, controlled by a *valve* module. Specially, $\mathcal{S}_6^d$ is the first side of expanding path and only originates from $\mathcal{S}_6^e$ in the contracting path.

The *valve* module can adaptively determine the flow of the useful high-level information from top sides to bottom sides, which will be further presented in Section 2.2. We adopt deep supervision [13] at the intermediate sides to perform predictions. Specifically, we add a *generator* module to generate saliency maps at each side, which will be introduced in Section 2.3. The predicted maps from all sides are concatenated to obtain the final saliency map. The proposed ATDF is trained end-to-end using the cross-entropy loss between the predicted saliency map and the ground truth.

2.2. The Valve Module

Different from existing methods [9, 13], our ATDF is able to automatically flow necessary semantic information from the top sides to bottom sides, guiding the learning of lower layers. To achieve this goal, we design a *valve* module, which make the high-level information flowing in the expanding path is non-redundant without manual intervention. By this way, the integrated features are complementary and only target the saliency detection in various scenes.

The architecture of the *valve* module is illustrated in the middle part of Fig. 1. The ATDF has five *valve* modules which are denoted as $\{\mathcal{V}_5, \mathcal{V}_4, \mathcal{V}_3, \mathcal{V}_2, \mathcal{V}_1\}$ from top to bottom, respectively. Each *valve* $\mathcal{V}_i$ ($i \in \{1, 2, 3, 4, 5\}$) has three inputs, *i.e.* $\mathcal{S}_{i+1}^d$ in the expanding path, $\mathcal{S}_{i+1}^e$ and $\mathcal{S}_i^e$ in the contracting path. After a series of operations for the inputs, the next block $\mathcal{S}_i^d$ is obtained from the valve $\mathcal{V}_i$.

The *valve* module mainly contains three operations. For the *valve* $\mathcal{V}_i$ ($i \in \{1, 2, 3, 4, 5\}$), the input $\mathcal{S}_{i+1}^d$ is first connected to a 1×1 convolution layer, followed by a deconvolution layer with the fixed *bilinear* kernel. After this step, the $\mathcal{S}_{i+1}^d$ is upsampled by a factor of 2. Then we perform a crop operation to make $\mathcal{S}_{i+1}^d$ the same size as $\mathcal{S}_i^e$. We denote the result of this step as $\mathcal{V}_1^i$ and formulate this step as

$$\mathcal{V}_i^1 = Crop(DeConv(Conv(\mathcal{S}_{i+1}^d))). \tag{1}$$

Secondly, we simply perform a linear transformation for the input $\mathcal{S}_i^e$ using 1×1 convolution without non-linearization. Then, we concatenate the result of above transformation and $\mathcal{V}_i^1$ to obtain $\mathcal{V}_i^2$. For a clear presentation, we can formulate this step as

$$\mathcal{V}_i^2 = Concat(Conv(\mathcal{S}_i^e), \mathcal{V}_i^1). \tag{2}$$

Finally, a residual block with two sequential convolution layers is connected to the feature map $\mathcal{V}_i^2$. Specifically, the element-wise addition is performed on the feature map obtained after two convolution layers and $\mathcal{V}_i^2$. These two sequential convolution layers are with kernel size 5×5 for $\{\mathcal{V}^5, \mathcal{V}^4\}$ and kernel size 3×3 for $\{\mathcal{V}^3, \mathcal{V}^2, \mathcal{V}^1\}$. The numbers of output channels are 256, 256, 128, 128 and 64 from $\mathcal{V}^5$ to $\mathcal{V}^1$, respectively. We formulate this step as

$$\mathcal{V}_i = \sigma(Conv(\sigma(Conv(\mathcal{V}_i^2)))) + \mathcal{V}_i^2 \tag{3}$$

where $\sigma(\cdot)$ denotes non-linearization, *i.e.* ReLU here.

We design a weight function to control the proportion of the top information ($\mathcal{S}_{i+1}^d$) flowing into the lower side, in which the information flowing rate is in the range of

$[0, 1]$. The weight function is defined as a convolutional layer with sigmoid activation. The formula of the weight function is $\mathcal{W}_i = Sigmoid(Conv(\mathcal{S}^e_{i+1}))$, where the kernel size of convolution layer is 3×3. We replace the original formula of $\mathcal{V}^1_i$ and $\mathcal{V}^2_i$ by $\mathcal{V}^{1\prime}_i = \mathcal{W}_i \otimes \mathcal{V}^1_i$ and $\mathcal{V}^{2\prime}_i = Catcat(Conv(\mathcal{S}^e_i), \mathcal{V}^{1\prime}_i)$, respectively. The $\otimes$ is element-wise product. The final output of valve module $\mathcal{V}_i{}'$, *i.e.* the obtained next block $\mathcal{S}^d_i$, can be expressed as

$$\mathcal{S}^d_i = \mathcal{V}_i{}' = \sigma(Conv(\sigma(Conv(\mathcal{V}^{2\prime}_i)))) + \mathcal{V}^{2\prime}_i. \quad (4)$$

The optimization of the valve is from top to bottom, so that the learning of weight function is also from top to bottom.

2.3. The Generator Module

The semantic information is learned in the top sides and gradually flows to the bottom sides along the expanding path. The ATDF should be optimized from top to down if we want to achieve automatic top-down information fusion. Therefore, instead of imposing supervision at the final layer of the expanding path, we use the scheme of adding deep supervision for all side outputs. As mentioned above, the *valve* module can automatically control the flow of global information from top sides to bottom sides, so we can obtain hybrid hierarchical feature maps. To further improve the capability of aggregated hierarchical information for saliency prediction, we adopt six *generator* modules $\{\mathcal{G}_6, \mathcal{G}_5, \mathcal{G}_4, \mathcal{G}_3, \mathcal{G}_2, \mathcal{G}_1\}$ after six sides, *i.e.* $\{\mathcal{S}^d_6, \mathcal{S}^d_5, \mathcal{S}^d_4, \mathcal{S}^d_3, \mathcal{S}^d_2, \mathcal{S}^d_1\}$, respectively. These *generator* modules can further process the feature maps to better fuse the high-level semantic information and low-level spatial details for accurate saliency prediction at specific scales.

The architecture of the *generator* is displayed in the right part of Fig. 1. The output $\mathcal{S}^d_i$ of each side in the expanding path is first connected to a residual convolution block with two sequential convolution layers, which are with kernel size of 7×7 for $\mathcal{S}^d_6$, 5×5 for $\{\mathcal{S}^d_5, \mathcal{S}^d_4\}$, and kernel size 3×3 for $\{\mathcal{S}^d_3, \mathcal{S}^d_2, \mathcal{S}^d_1\}$. The numbers of channels for them are 512, 256, 256, 128, 128 and 64, respectively. The element-wise addition is then performed on the feature map after above convolution layers and $\mathcal{S}^d_i$ to build the residual path. We add a 1×1 convolution layer without non-linearization to change the channel number to 1 for each side. This 1-channel feature map is the saliency prediction at each side. We upsample the saliency map to the same size as original image using a deconvolution layer with the fixed *bilinear* kernel. For better description, we summarize the operations of the *generator* module as the following formula:

$$\begin{aligned} \mathcal{F}_i &= \sigma(Conv(\sigma(Conv(\mathcal{S}^d_i)))) + Conv(\mathcal{S}^d_i), \\ \mathcal{G}_i &= Crop(DeConv(Conv(\mathcal{F}_i))). \end{aligned} \quad (5)$$

With the *generator* modules, we can obtain a saliency prediction map at each side, after which we add the deep supervision. Finally, we concatenate the predicted maps at all sides to obtain the final saliency prediction map.

3. EXPERIMENTS

3.1. Experimental Setup

Implementation Details. We implement the proposed network using the Caffe framework. The parameters of the layers contained in VGG16 [19] are initialized by the pretrained ImageNet model. We optimize the proposed network using SGD. The learning rate policy is *poly*, in which the current learning rate equals the base one multiplying $(1 - curr\_iter/max\_iter)^{power}$. Here, the hyper parameters $power$ is set to 0.9, and we run 20000 SGD iterations (max_iter) in total. The initial learning rate is set to 2.5e-8. We set the momentum and weight decay to 0.9 and 0.0005, respectively. All the experiments are performed on a TITAN Xp GPU.

Datasets. To fine-tune our model, we follow recent studies [12, 14] to utilize the DUTS [20] training dataset. We extensively evaluate our method on the DUTS test set and other five popular datasets including ECSSD [21], SOD [22], HKU-IS [23], THUR15K [24] and DUT-OMRON [25].

Evaluation Criteria. In this paper, we adopt two evaluation metrics to evaluate the performance of saliency detection models: the max F-measure score and mean absolute error (MAE). With a threshold value, the predicted saliency map can be converted into a binary map, with which we can compute the precision and recall values. The performance on a dataset is the average over all images. The F-measure score is computed as $F_\beta = \frac{(1+\beta^2) \times Precision \times Recall}{\beta^2 \times Precision + Recall}$, in which β^2 is typically set to 0.3 to emphasize more on precision. We report max F_β across all thresholds. Furthermore, given a saliency map P and the corresponding ground truth G that are normalized to [0, 1], MAE can be calculated as $MAE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} |P(i,j) - G(i,j)|$ where H and W are image height and width, respectively. $P(i, j)$ and $G(i, j)$ denote the saliency score at location (i, j).

3.2. Performance Comparison

We compare our proposed salient object detector with 12 recent state-of-the-art models, including LEGS [5], ELD [6], RFCN [7], DCL [8], Amulet [9], UCF [10], NLDF [11], SRM [12], DSS [13], PiCA [14], C2S [15] and RAS [16]. All these methods are tested using their released code and pretrained models provided by the authors with default settings.

Quantitative Evaluation. We summarize the numeric comparison with respect to the F_β and MAE on six datasets in Table 1. ATDF significantly outperforms other competitors in almost all cases, which demonstrates its effectiveness. We also use the ResNet as ATDF's backbone net and ATDF also achieves significantly better performance than all baselines.

Qualitative Evaluation. Fig. 2 shows the qualitative comparison between ATDF and other methods. We can see that our model can successfully segment the objects with fine details, leading to better saliency predictions in various scenarios.

Methods	SOD		HKU-IS		ECSSD		DUT-OMRON		THUR15K		DUTS-test	
	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE
VGG16 backbone												
LEGS [5]	0.733	0.194	0.766	0.119	0.830	0.118	0.668	0.134	0.663	0.126	0.652	0.137
ELD [6]	0.758	0.154	0.837	0.074	0.866	0.081	0.700	0.092	0.726	0.095	0.727	0.092
RFCN [7]	0.802	0.161	0.892	0.080	0.896	0.097	0.738	0.095	0.754	0.100	0.782	0.089
DCL [8]	0.831	0.131	0.892	0.063	0.895	0.080	0.733	0.095	0.747	0.096	0.785	0.082
Amulet [9]	0.795	0.144	0.897	0.051	0.913	0.061	0.743	0.098	0.755	0.094	0.778	0.085
UCF [10]	0.805	0.148	0.888	0.062	0.901	0.071	0.730	0.120	0.758	0.112	0.772	0.112
NLDF [11]	0.837	0.123	0.902	0.048	0.902	0.066	0.753	0.080	0.762	0.080	0.806	0.065
DSS [13]	**0.842**	**0.122**	**0.913**	**0.041**	0.915	**0.056**	**0.774**	**0.066**	0.770	**0.074**	0.827	**0.056**
PiCA [14]	0.836	**0.102**	**0.916**	**0.042**	**0.923**	**0.049**	0.766	0.068	**0.783**	0.083	**0.837**	**0.054**
C2S [15]	0.819	**0.122**	0.898	0.046	0.907	0.057	0.759	0.072	**0.775**	0.083	0.811	0.062
RAS [16]	**0.847**	0.123	**0.913**	0.045	**0.916**	0.058	**0.785**	**0.063**	0.772	**0.075**	**0.831**	0.059
ATDF (ours)	**0.859**	**0.114**	**0.927**	**0.032**	**0.931**	**0.044**	**0.795**	**0.055**	**0.796**	**0.066**	**0.863**	**0.042**
ResNet backbone												
SRM [12]	0.840	0.126	0.906	0.046	0.914	0.056	0.769	0.069	0.778	0.077	0.826	0.059
PiCA [14]	0.852	**0.103**	0.917	0.043	0.929	0.049	0.789	0.065	0.788	0.081	0.853	0.050
ATDF (ours)	**0.862**	0.110	**0.933**	**0.031**	**0.939**	**0.040**	**0.814**	**0.051**	**0.801**	**0.064**	**0.877**	**0.037**

Table 1. Comparison between our ATDF and 12 state-of-the-art methods in terms of F_β (the larger the better) and MAE (the smaller the better) on six datasets. We highlight the top three results of each column in **red**, **green** and **blue**, respectively.

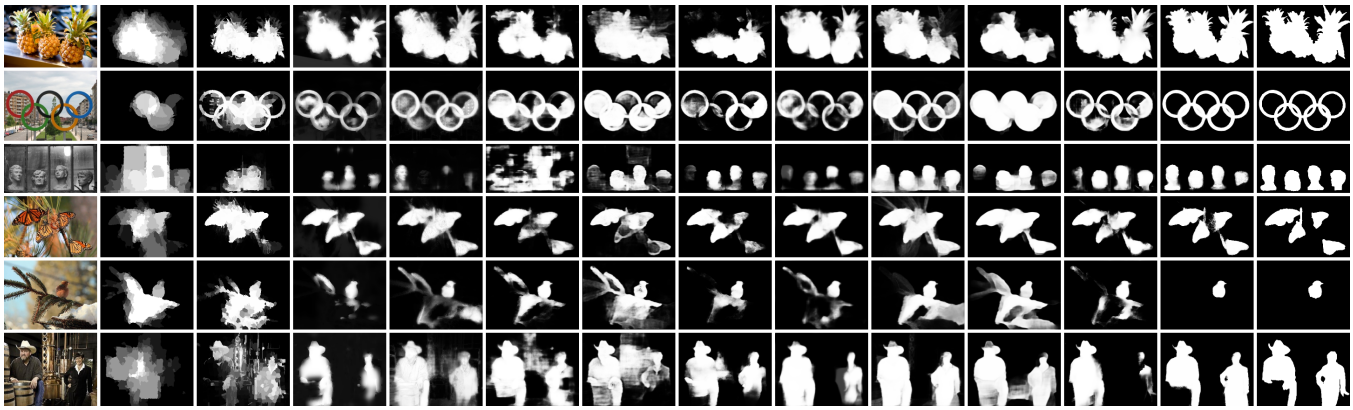

Fig. 2. Qualitative comparison of ATDF and 11 state-of-the-art methods.

Discussion of Weight Function. Without the weight function in the *valve* module, we obtain (F_β, MAE) of (0.847, 0.115), (0.922, 0.033), (0.927, 0.044), (0.781, 0.061), (0.789, 0.072), and (0.851, 0.047) on six datasets as the same order of Table 1. Hence weight function can significantly benefit the proposed saliency detector.

4. CONCLUSIONS

In this paper, we propose a novel Automatic Top-Down Fusion (ATDF) model for saliency detection, which can automatically flow the global information at the top sides into bottom sides to guide the learning of bottom sides. To do this, we design a *valve* module to filter out redundant and remain the necessary top global information for each side. ATDF is simple yet effective in integrating the multi-level convolutional features of deep networks. Therefore, ATDF overcomes the problem of the sub-optimal saliency detection in recent manually designed fusion strategies. Extensive experiments on six datasets demonstrate that ATDF outperforms 12 recent state-of-the-art methods in terms of various evaluation metrics.

5. ACKNOWLEDGMENT

This work is supported by Science and Technology Planning Project of Tianjin (17JCZDJC30700 and 18ZXZNGX00310), Fundamental Research Funds for the Central Universities of Nankai University (63191402).

6. REFERENCES

[1] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu, "Global contrast based salient region detection," *IEEE TPAMI*, vol. 37, no. 3, pp. 569–582, 2015.

[2] Vijay Mahadevan and Nuno Vasconcelos, "Saliency-based discriminant tracking," in *IEEE CVPR*, 2009.

[3] Yue Gao, Meng Wang, Zheng-Jun Zha, Jialie Shen, Xuelong Li, and Xindong Wu, "Visual-textual joint relevance learning for tag-based social image search," *IEEE TIP*, vol. 22, no. 1, pp. 363–376, 2013.

[4] Yunchao Wei, Xiaodan Liang, Yunpeng Chen, Xiaohui Shen, Ming-Ming Cheng, Jiashi Feng, Yao Zhao, and Shuicheng Yan, "STC: A simple to complex framework for weakly-supervised semantic segmentation," *IEEE TPAMI*, vol. 39, no. 11, pp. 2314–2320, 2017.

[5] Lijun Wang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang, "Deep networks for saliency detection via local estimation and global search," in *IEEE CVPR*, 2015, pp. 3183–3192.

[6] Gayoung Lee, Yu-Wing Tai, and Junmo Kim, "Deep saliency with encoded low level distance map and high level features," in *IEEE CVPR*, 2016, pp. 660–668.

[7] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan, "Saliency detection with recurrent fully convolutional networks," in *ECCV*, 2016, pp. 825–841.

[8] Guanbin Li and Yizhou Yu, "Deep contrast learning for salient object detection," in *IEEE CVPR*, 2016, pp. 478–487.

[9] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Xiang Ruan, "Amulet: Aggregating multi-level convolutional features for salient object detection," in *IEEE ICCV*, 2017, pp. 202–211.

[10] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Baocai Yin, "Learning uncertain convolutional features for accurate saliency detection," in *IEEE ICCV*, 2017, pp. 212–221.

[11] Zhiming Luo, Akshaya Kumar Mishra, Andrew Achkar, Justin A Eichel, Shaozi Li, and Pierre-Marc Jodoin, "Non-local deep features for salient object detection," in *IEEE CVPR*, 2017, pp. 6609–6617.

[12] Tiantian Wang, Ali Borji, Lihe Zhang, Pingping Zhang, and Huchuan Lu, "A stagewise refinement model for detecting salient objects in images," in *IEEE ICCV*, 2017, pp. 4019–4028.

[13] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip Torr, "Deeply supervised salient object detection with short connections," in *IEEE CVPR*, 2017, pp. 5300–5309.

[14] Nian Liu, Junwei Han, and Ming-Hsuan Yang, "PiCANet: Learning pixel-wise contextual attention for saliency detection," in *IEEE CVPR*, 2018, pp. 3089–3098.

[15] Xin Li, Fan Yang, Hong Cheng, Wei Liu, and Dinggang Shen, "Contour knowledge transfer for salient object detection," in *ECCV*, 2018, pp. 355–370.

[16] Shuhan Chen, Xiuli Tan, Ben Wang, and Xuelong Hu, "Reverse attention for salient object detection," in *ECCV*, 2018.

[17] Lu Zhang, Ju Dai, Huchuan Lu, You He, and Gang Wang, "A bi-directional message passing model for salient object detection," in *IEEE CVPR*, 2018, pp. 1741–1750.

[18] Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Jia-Wang Bian, Le Zhang, Xiang Bai, and Jinhui Tang, "Richer convolutional features for edge detection," *IEEE TPAMI*, 2019.

[19] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[20] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan, "Learning to detect salient objects with image-level supervision," in *IEEE CVPR*, 2017, pp. 136–145.

[21] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia, "Hierarchical saliency detection," in *IEEE CVPR*, 2013, pp. 1155–1162.

[22] Vida Movahedi and James H Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *IEEE CVPR*, 2010, pp. 49–56.

[23] Guanbin Li and Yizhou Yu, "Visual saliency based on multiscale deep features," in *IEEE CVPR*, 2015, pp. 5455–5463.

[24] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, and Shi-Min Hu, "Salientshape: Group saliency in image collections," *The Visual Computer*, vol. 30, no. 4, pp. 443–453, 2014.

[25] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang, "Saliency detection via graph-based manifold ranking," in *IEEE CVPR*, 2013, pp. 3166–3173.