# An Iterative and Cooperative Top-down and Bottom-up Inference Network for Salient Object Detection

Wenguan Wang [1], Jianbing Shen*[1], Ming-Ming Cheng [2], Ling Shao [1]
[1]Inception Institute of Artificial Intelligence, UAE [2]CS, Nankai University, China
wenguanwang.ai@gmail.com, shenjianbingcg@gmail.com, cmm@nankai.edu.cn, ling.shao@ieee.org

## Abstract

*This paper presents a salient object detection method that integrates both top-down and bottom-up saliency inference in an iterative and cooperative manner. The top-down process is used for coarse-to-fine saliency estimation, where high-level saliency is gradually integrated with finer lower-layer features to obtain a fine-grained result. The bottom-up process infers the high-level, but rough saliency through gradually using upper-layer, semantically-richer features. These two processes are alternatively performed, where the bottom-up process uses the fine-grained saliency obtained from the top-down process to yield enhanced high-level saliency estimate, and the top-down process, in turn, is further benefited from the improved high-level information. The network layers in the bottom-up/top-down processes are equipped with recurrent mechanisms for layer-wise, step-by-step optimization. Thus, saliency information is effectively encouraged to flow in a bottom-up, top-down and intra-layer manner. We show that most other saliency models based on fully convolutional networks (FCNs) are essentially variants of our model. Extensive experiments on several famous benchmarks clearly demonstrate the superior performance, good generalization, and powerful learning ability of our proposed saliency inference framework.*

## 1. Introduction

Salient object detection (SOD) aims to highlight the most visually important object(s) during human perception of a visual stimulus. This task is essential for understanding the visual world from the perspective of human visual attention behavior. This paper proposes a novel, biologically-inspired network architecture: an iterative top-down and bottom-up saliency inference network, which imitates the interactive top-down and bottom-up processes of human perception. During the top-down inference, the rough high-level saliency (Fig. 1(a)) from the semantically-rich, upper-layer features is improved by progressively integrating finer, low-level features. The resulting finest saliency (Fig. 1(b)) is then leveraged, in turn, to guide improved high-level saliency estimation in a bottom-up way. This produces a new high-level estimate (Fig. 1(c)), which is better than the original (Fig. 1(a)) and further guides a more accurate finest saliency estimation (Fig. 1(d)) in a new top-down process. This whole procedure can be repeated, alternating between the top-down and bottom-up processes, until the final, optimal saliency estimate is obtained (Fig. 1(f)). In this way, the top-down and bottom-up inferences are interlaced to work in an iterative and cooperative fashion. In particular, two aspects inspire our network design, as follows.

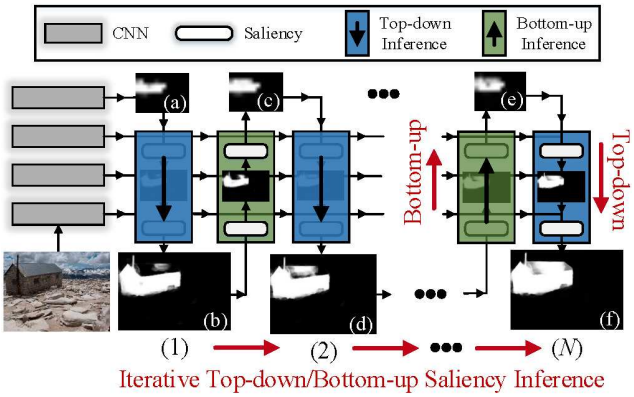


Figure 1: High-layer saliency (a) is used to guide fine-grained saliency estimation in a top-down fashion. The finest saliency (b) is then adapted to refine higher-layer saliency (c) in a bottom-up manner. Alternating these two processes (1)→(2)→···→($N$) results in an iterative and cooperative top-down and bottom-up inference network.

Firstly, our model is motivated by studies of the human perception system by the psychology and cognitive communities. To explain human visual information-processing, two types of perceptive processes are distinguished: bottom-up and top-down processing. In the top-down procedure [8], high-level semantic knowledge or contextual information guides information processing. The bottom-up process [7], in contrast, is carried out in

*Corresponding author: *Jianbing Shen*.

the opposite direction; from the stimulus to high-level visual understanding, with each successive stage performing an ever more complex analysis of the input. Most psychologists [13, 12] would now argue that both top-down and bottom-up processes are involved in perception. This prompts us to explore both systems for saliency inference in a *joint and iterative way*. Since it is well-known that top-network layers carry semantically-rich information, while bottom layers involve low-level details, we consider top-layer saliency to be a globally harmonious interpretation of a visual scene, which is used to guide fine-grained saliency estimation in a top-down manner. Conversely, if starting from the finest saliency, we can leverage it to refine the upper-layer saliency in a bottom-up fashion.

Recent advances in FCN-based SOD models are the second inspiration for our network design. Their success is due to their progressive integration of rough upper-layer saliency with low-level but spatially defined features. The final estimate is thus the finest and most accurate. This prompts us to consider *why not leverage the final, finest saliency to reversely improve the previous, upper-layer estimates through a bottom-up procedure, and then repeat the coarse-to-fine, top-down process to obtain a more accurate, finest estimate?* With this intuitive yet significant insight, we develop a powerful, general saliency inference network that deploys the two processes in *a cooperative manner*. This network uses the fine-grained saliency from the top-down process to refine higher-layer estimates, in a bottom-up direction. The refined high-level saliency is then used to further encourage more efficient top-down inference. As seen, such a design efficiently promotes the exchange of saliency information between different layers, resulting in better learning and inference abilities. Even more significantly, our model is equipped with a complete and iterative feed-forward/feed-back inference strategy, rather than the commonly-used feed-forward networks [26, 46, 41, 20] or UNet-like models [10, 45, 24, 46, 11], which provide only simple coarse-to-fine inference. Feed-forward saliency models are limited due to their lack of a feed-back strategy. Although UNet-like SOD models partially solve this, they do not consider the interaction and integration of the top-down and bottom-up processes. From the perspective of network design, most previous FCN-based saliency models can be considered as specific forms of our proposed saliency inference network. A more detailed discussion of this will be given in §2.2. Although some previous works [1, 31, 23] have applied top-down and bottom-up strategies, they are either not learnt in an end-to-end trainable manner or do not let the two processes cooperate iteratively with each other.

For higher effectiveness, iterative saliency inference is performed on two levels. Macroscopically, multiple top-down/bottom-up inference layers are stacked together, to enable top-down/bottom-up inference to be performed iteratively. On a micro-level, our saliency inference module is achieved by a recurrent network, which shares the spirit of [39, 34, 22, 35], performing step-by-step saliency estimation within each individual layer. By adding supervision to all the side-outputs of each iteration, our model can be efficiently trained by a *deeper supervision strategy*.

To summarize, our model has several essential characteristics and advantages: **i)** it provides deeper insight into the SOD task by imitating top-down and bottom-up human perception processes; **ii)** it provides an end-to-end framework that learns top-down/bottom-up saliency inference in a joint and iterative way; **iii)** it provides a powerful tool that extends deep SOD models through a complete and iterative feed-forward and feed-back strategy, making it sufficiently general and flexible to encompass most other FCN-based saliency models. Finally, the promising results on several famous SOD benchmarks [42, 43, 18, 21, 25, 33] demonstrate effectiveness of our network design.

## 2. Related Work

In §2.1, we first review representative deep learning based SOD models. Then, in §2.2, to demonstrate the advantage of our proposed model, we provide a detailed discussion on its relation to other FCN-based saliency models.

### 2.1. Deep Learning Based SOD Models

With the rapid development of deep learning techniques, deep neural network based SOD models have substantially improved. Early attempts focused on *fully-connected networks*, which leverage compact image regions, such as over-segments [18], superpixels [48, 19, 17, 11], or object proposals [32]), as basic processing units. The deep features extracted from the image regions are fed into a fully-connected classifier for saliency score prediction. Compared to previous, non-deep learning SOD methods, the fully-connected SOD schemes display improved performance. However, they suffer from loss of spatial information and are quite time-consuming since they perform saliency inference in a segment-wise manner.

To overcome the aforementioned drawbacks, later SOD models directly map an input image to a pixel-wise saliency prediction, using *fully convolutional networks (FCNs)*. Several works [10, 45, 24, 46, 11, 44] tried to integrate multi-layer features together to preserve high-level semantic information as well as spatial details. For example, in [10], a skip-layer structure was proposed to build connections between different network layers for a more comprehensive saliency representation. Others emphasized the value of visual fixation [14, 39, 4] for explicit, object-level saliency understanding. In addition, some [15, 34, 22, 35, 39] attempted to leverage recurrent neural networks to optimize saliency estimation step-by-step. Differentiable attention models were also explored in several designs [44, 23, 37, 3].

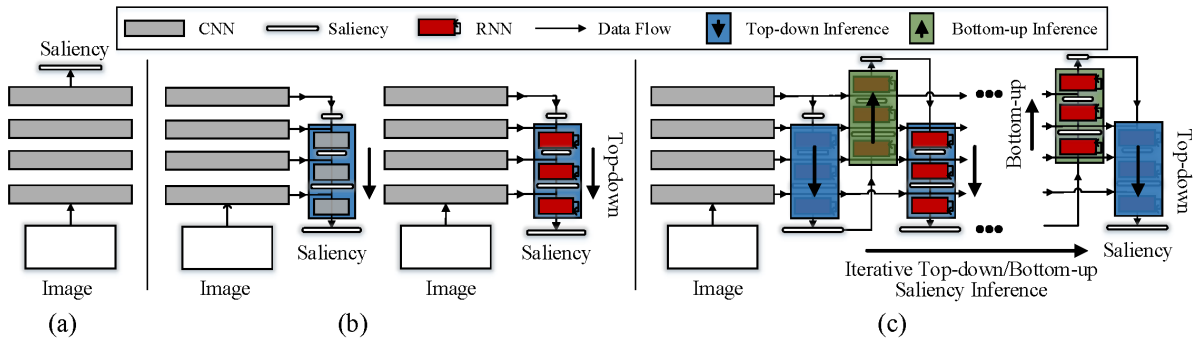


Figure 2: Schematization of previous FCN-based saliency models and our iterative top-down and bottom-up inference network. (a) Typical feed-forward network based SOD models only perform feed-forward inference. (b) Top-down inference based SOD models carry out top-down refinement via CNNs (left) or RNNs (right). (c) Our model iteratively performs top-down and bottom-up inference, and thus the two processes can benefit each other. See §2.2 for more details.

FCN-based SOD models generally display impressive results, demonstrating the advantages of applying neural networks to this task. However, they do not typically consider the top-down process and seldom explore the top-down and bottom-up inference in a joint, iterative, and end-to-end manner. More discussion regarding this will be detailed in the next section.

### 2.2. Relation to Previous FCN-based SOD Models

We shall now discuss the relation between our proposed model and general FCN-based saliency methods, from the perspective of network architecture. This will better situate our work with respect to previous studies and help highlight our contributions. As shown in Fig. 2, we classify previous FCN based models into two categories: *feed-forward network based* models, and *top-down inference based* models.

*Feed-forward network based* models [26, 46, 41, 20, 30, 40] are built upon a feed-forward, fully convolutional encoder-decoder architecture, *i.e.,* several sequentially stacked convolution and deconvolution layers. As shown in Fig. 2 (a), such a network design is straightforward and widely used, but has the disadvantage of losing much spatial detail due to the pooling operation within the encoder.

*Top-down inference based* methods, mainly inspired by the UNet [28] and top-down segmentation model [27], perform saliency inference in a top-down refinement fashion, *i.e.,* gradually optimizing the rough, upper-layer estimate through the finer, inferior-layer features. These methods can be further categorized into two classes. The first one [10, 45, 24, 46, 11, 44, 23, 37, 3, 38], as shown on the left of Fig. 2 (b), is a fully convolutional network, where top-down inference is achieved by convolution layer. The second class [15, 34, 22, 35, 39], as illustrated on the right of Fig. 2 (b), introduce recurrent mechanisms to iteratively optimize each side-out by updating the hidden states of the RNN. While all of these methods benefit from top-down inference, few consider using it in conjunction with a bottom-up process. Liu *et al*. [23] are the exception to this. However, while they do also consider bottom-up saliency propagation, the two processes are not carried out in an iterative and cooperative manner.

Fig. 2 (c) gives the core scheme of our *iterative top-down and bottom-up saliency inference network*. As shown, the model first performs top-down, coarse-to-fine saliency inference to obtain the finest saliency estimate (shown in the blue rectangle). The bottom-layer output is then leveraged to refine the upper-layer estimates in reverse (shown in the green rectangle). Using the more accurate top-level saliency estimate, a more efficient top-down inference can be performed. These top-down and bottom-up processes are iteratively performed to produce improved, step-wise results. The top-down/bottom-up inference modules are formed by the RNN. In this way, the proposed model inherits the advantages of the aforementioned top-down inference based saliency models, while going a step further and using the top-down/bottom-up processes in a joint, iterative and end-to-end manner. Most previous FCN-based SOD models can be viewed as special cases of our proposed model.

## 3. Our Method

In §3.1 we first present the formulations of our iterative top-down and bottom-up saliency inference model. In §3.2, we discuss possible variations and implementation details.

### 3.1. Iterative Top-down and Bottom-up Saliency Inference Model

A convolutional neural network, typically borrowed from the convolutional parts of VGGNet [29] or ResNet [9], is first used as a saliency feature extractor. It consists of a series of convolution layers, interleaved with pooling and non-linear activation operations, enabling it to capture multiple levels of abstraction from a visual stimulus. The features from the bottom layers encode richer, low-level information, while the upper-layer activations carry more high-level semantics, with less spatial detail. We view the gradual

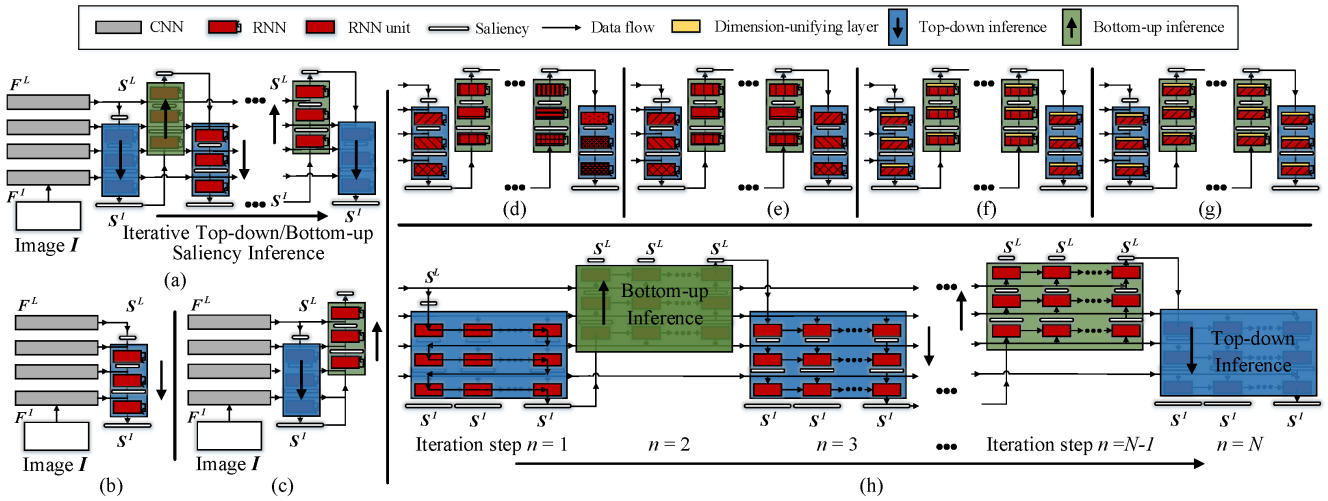


Figure 3: Detailed network architecture of the suggested iterative top-down and bottom-up inference network. (a) Simplified schematization of our model. (b) Illustration of top-down saliency inference. (c) Illustration of top-down inference cascaded with bottom-up inference. (d)-(g) Illustration of different weight-sharing strategies, where RNN-based top-down/bottom-up inference layers of different weights are labeled with different textures. See §3.1 for full discussions. (h) Detailed network architecture of the suggested model with RNN-based top-down/bottom-up inference layers, where the recurrent nature of the RNN is leveraged to gradually improve saliency estimation within each layer. See §3.2 for details.

integration of upper-layer saliency estimates with inferior-layer features as *top-down saliency inference*. This produces finer saliency because more detailed spatial information is used. Similarly, a finer saliency can be combined with an upper-layer feature to produce a higher-level but more coarse saliency, referred as *bottom-up saliency inference*. The *central hypothesis* is that the highly-accurate, finest saliency obtained from the top-down inference can be used to provide a more accurate high-level saliency estimate through a bottom-up process; the more accurate high-level saliency further enables more efficient bottom-up inference. In this way, these two processes are performed in a joint and iterative manner to encourage one another.

Let $\mathbf{I} \in \mathbb{R}^{W\times H\times 3}$ and $\mathcal{F}$ denote an input image and the feature extractor network, respectively. Hierarchical saliency features with $L$ levels are obtained by: $\{\mathbf{F}^l\}_{l=1}^{L} = \mathcal{F}(\mathbf{I})$, where $\mathbf{F}^l \in \mathbb{R}^{w^l\times h^l\times c^l}$. As shown in Fig. 3 (a), in our model, the top-layer saliency $\mathbf{S}^L \in [0,1]^{w^L\times h^L}$ is first estimated as a high-level saliency interpretation of the visual scene, as it utilizes the most semantically-rich feature $\mathbf{F}^L$. This is used to guide fine-grained saliency estimation in a top-down manner.

**Top-down saliency inference.** For an inferior $l$-th level ($l \in \{1,\dots,L-1\}$), given the upper-layer saliency estimate $\mathbf{S}^{l+1} \in [0,1]^{w^{l+1}\times h^{l+1}}$, the corresponding finer saliency $\mathbf{S}^l \in [0,1]^{w^l\times h^l}$ can be formulated through a top-down inference layer $\mathcal{T}^l : \mathbb{R}^{w^l\times h^l\times (c^l+1)} \mapsto [0,1]^{w^l\times h^l}$:

$$\mathbf{S}^l = \mathcal{T}^l([\mathcal{U}(\mathbf{S}^{l+1}), \mathbf{F}^l]), \tag{1}$$

where $\mathcal{U}(\cdot)$ is upsampling operation that resizes the upper-layer saliency $\mathbf{S}^{l+1}$ into the current spatial resolution $w^l \times h^l$. $[\cdot,\cdot]$ denotes the concatenation operation. The top-down inference layer is composed of several conv (or RNN) and activation layers, details of which will be elaborated in §3.2.

As discussed in §2.2, most previous FCN-based SOD methods can be modeled by Eq. 1, *i.e.* they perform top-down inference layer-by-layer until obtaining the final, finest saliency estimation $\mathbf{S}^1 \in [0,1]^{w^1\times h^1}$ (see Fig. 3 (a)). One fundamental characteristics that distinguishes our method from others is that, after completion of the top-down inference process, we further use the finest saliency $\mathbf{S}^1$ to reversely optimize upper-layer saliency estimates in a bottom-up manner. Previous methods have extensively and empirically confirmed that the finest saliency achieves the best performance. Thus, we can predict that applying a bottom-up inference process to $\mathbf{S}^1$ will yield better higher-layer saliency estimates. A bottom-up saliency inference is thus derived, as follows.

**Bottom-up saliency inference.** In the $l$-th network layer ($l \in \{2,\dots,L\}$), a bottom-up saliency inference layer is designed as $\mathcal{B}^l : \mathbb{R}^{w^l\times h^l\times (c^l+1)} \mapsto [0,1]^{w^l\times h^l}$, which uses the inferior-layer saliency estimate $\mathbf{S}^{l-1} \in [0,1]^{w^{l-1}\times h^{l-1}}$ and current-layer saliency feature $\mathbf{F}^l \in \mathbb{R}^{w^l\times h^l\times c^l}$ as inputs, and produces a refined saliency estimate $\mathbf{S}^l \in [0,1]^{w^l\times h^l}$:

$$\mathbf{S}^l \leftarrow \mathcal{B}^l([\mathcal{D}(\mathbf{S}^{l-1}), \mathbf{F}^l]), \tag{2}$$

where '$\leftarrow$' indicates the updating process, and $\mathcal{B}(\cdot)$ is the necessary downsampling operation. As shown in Fig. 3 (c), during the bottom-up inference process, the inferior saliency is combined with the current-layer feature to develop a better high-level saliency.

**Iterative top-down and bottom-up inference.** The

bottom-up inference can be used to achieve a more accurate top-level saliency $S^L$. In other words, the more precise starting point enables us to perform more accurate top-down inference. Repeating this process leads to an iterative top-down and bottom-up saliency inference framework.

More formally, for the $l$-th network layer, the corresponding saliency estimate $\mathbf{S}^l$ can be iteratively updated by repeatedly performing top-down and bottom-up inference:

$$\mathbf{S}^l \leftarrow \mathcal{T}^l(\ldots\mathcal{T}^l(\mathcal{B}^{l+1}(\mathcal{T}^l(\ldots)))). \tag{3}$$

As such, this model mimics the interactive top-down and bottom-up processes of the human perception system by arranging them in a cascaded and cooperative manner (Fig. 3 (a)). Fig. 1 and 4 (3*rd*, 4*th* and 5*th* rows) visualize the improved results after several iterations. Detailed quantitative studies can be found in §4.3.

**Feature-sharing and weight-sharing.** Before delving into a detailed overview of the network implementations, it is worth discussing some important characteristics of our model. One significant element is the sharing of features $\{\mathbf{F}^l\}_{l=1}^L$ between the top-down and bottom-up inference stages and across different iterations. This provides the advantage of high computational efficiency, since features are only calculated once, regardless of the number of iterations.

Another important characteristic of our model is weight-sharing. Rather than simply learning each top-down/bottom-up inference layer in different iterations individually (Fig. 3 (d)), for a given $l$-th level, we introduce weight-sharing between bottom-up (or top-down) inference layers in different iteration steps (Fig. 3 (e)). Additionally, by changing our network design slightly (Fig. 3 (f)), we can enforce parameter-sharing among different bottom-up (or top-down) inference layers across all iterations and layers. Specifically, since the bottom-up (and top-down) inference layers are parameterized by a set of conv kernels, we can add *dimension-unifying* layers that unite the channel-dimensions of the input features of all bottom-up (and top-down) inference layers so that they are the same. The dimension-unifying layer $\mathcal{R}$ can be implemented by a $1\times1$ conv kernel, as a feature-dimensionality reduction function. As such, the number of parameters of $\mathcal{R}$ is linear to the dimensions of the input feature channel. Accordingly, Eqs. 1-2 can be reformulated as follows:

$$\begin{aligned}\mathbf{S}^l &= \mathcal{T}^l(\mathcal{R}^l([\mathcal{U}(\mathbf{S}^{l+1}), \mathbf{F}^l])),\\ \mathbf{S}^l &\leftarrow \mathcal{B}^l(\mathcal{R}^l([\mathcal{D}(\mathbf{S}^{l-1}), \mathbf{F}^l])).\end{aligned} \tag{4}$$

Furthermore, if pursuing an extremely light-weight form, we can even employ parameter-sharing among all the bottom-up $\{\mathcal{B}^l\}_{l=1}^L$ and top-down inference layers $\{\mathcal{T}^l\}_{l=1}^L$, across all iteration steps (Fig. 3 (g)). This can be achieved by letting the dimension-unifying layers compress each input of the bottom-up/top-down inference layers to have the same channel-dimension. In such a situation, iterative top-down and bottom-up inference is achieved by a light-weighted network with few parameters. In summary, through weight-sharing, the proposed iterative top-down and bottom-up inference layers can be used as add-ons to modern network architectures. This demonstrates the general and flexible nature of our proposed model.

### 3.2. Possible Variants and Detailed Architecture

Interestingly, our model offers a general, unified and flexible framework, instead of being limited to a specific network architecture. This enables us to explore several variants by implementing the top-down/bottom-up inference layers in different ways. We continue by next formalizing these possible configurations.

**CNN-based top-down/bottom-up inference.** The most straightforward way to implement the top-down/bottom-up inference layers is by modeling them as a stack of conv layers. By equipping them with non-linear activations, the conv layers can learn complex inference functions that leverage upper- or inferior-layer saliency and current-layer feature for fine-grained (top-down process) or higher-level (bottom-up process) saliency inference in a non-linear way.

**RNN-based top-down/bottom-up inference.** Sharing a similar spirit to RNN-based SOD models [15, 34, 22, 35, 39], we can also employ an RNN in top-down/bottom-up inference layers. This introduces a feed-back mechanism into each inference layer, enabling more efficient propagation of information. With the recurrent model, the saliency inference is performed step-by-step for each layer, at each iteration. Additionally, to preserve spatial details, convolutional RNNs, rather than fully connected ones, should be adopted. A convRNN can be constructed by replacing dot products with spatial convolutions. In this way, the top-down inference in Eq. 1 can be formulated as:

$$\begin{aligned}\mathbf{h}_t &= \mathcal{T}^l_{covRNN}([\mathcal{U}(\mathbf{S}^{l+1}), \mathbf{F}^l], \mathbf{h}_{t-1}),\\ \mathbf{S}^l_t &= \mathcal{T}^l_{Readout}(\mathbf{h}_t),\\ \mathbf{S}^l &= \mathbf{S}^l_T.\end{aligned} \tag{5}$$

where $\mathbf{h}_t$ indicates the hidden state at step $t \in [1, \ldots, T]$, $\mathcal{T}^l_{Readout}(\cdot)$ is a *readout function* ($1\times1$ conv with $sigmoid$) that outputs saliency $\mathbf{S}^l_t$ from $\mathbf{h}_t$, and the final $\mathbf{S}^l_T$ at step $T$ is used as the output saliency $\mathbf{S}^l$ of the current top-down inference layer $\mathcal{T}^l$. Note that, for a given top-down inference layer $\mathcal{T}^l$, the input features are the same for each step, since it operates on static images. The top-down inference layer in Eq. 2 can be calculated in a similar way. Please see Fig. 3 (h) for a detailed illustration.

Here, we use the advantages of the recurrent nature of the RNN to iteratively optimize the saliency features of static images, rather than to model the temporal dependency of sequential data. Our model is not limited to specific RNN structures. In our experiments (§4.3), we implement top-down/bottom-up inference layers as convRNN, convGRU, and convLSTM, respectively. Additionally, the network

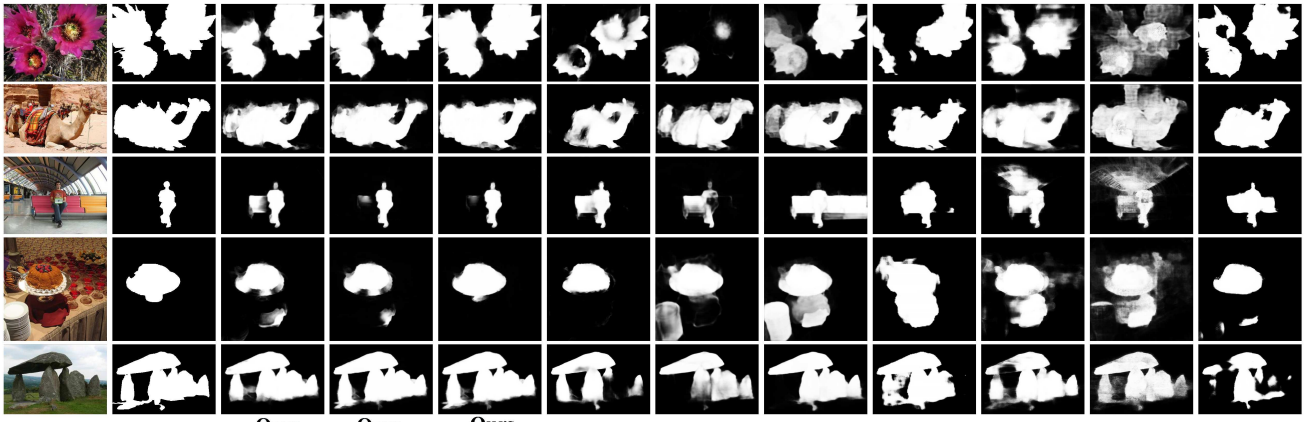


Figure 4: Qualitative comparison of seven state-of-the-art deep SOD models: RAS [3], PAGR [47], PICA [23], FSN [4], AMU [45], UCF [46], HEDS [10] and our model. It can be observed that the proposed model generates more accurate saliency estimations. Additionally, the saliency estimates are gradually improved with an increasing number of bottom-up and top-down iterations (from the 3*rd* row to the 5*th* row). Best viewed in color.

with CNN-based top-down/bottom-up inferences simply be considered as a special case of the one based on RNNs, where the total number of update steps $T$ of each RNN is set to 1. Thus, we equip our model with RNN-based top-down/bottom-up inference layers, and test the performance with different update steps $T$ in §4.3.

**Training with deeper supervision.** Lee *et al*. [16] proposed a deep supervision strategy, *i.e*., adding supervisions into each network side-output layer. This has been widely used in previous FCN-based SOD models. In our approach, due to the iteration of bottom-up/top-down inference and the use of RNN models, we propose a *deeper supervision* training strategy. In other words, we provide further supervision to the outputs from each iteration (of the bottom-up/top-down inferences) and each update step (of the RNN units). Formally, let $\mathbf{S}_{t,n}^{l,k} \in [0,1]^{w^l \times h^l}$ denote the saliency from the $t$-th step of the RNN unit of the $k \in \{\text{bottom-up}, \text{top-down}\}$ process, with the $l$-th network layer, during the $n$-th iteration. $\mathbf{G}^l \in \{0,1\}^{w^l \times h^l}$ denote the corresponding ground-truth. Our model can be trained by minimizing the following loss:

$$\mathcal{L}(\{\mathbf{S}_{t,n}^{l,k}\}_{l,k,t,n}, \{\mathbf{G}^l\}_l) = \underbrace{\sum_{l=1}^{L} \sum_{k \in \{\text{bottom-up}, \text{top-down}\}} \sum_{n=1}^{N} \sum_{t=1}^{T} \mathcal{L}_C(\mathbf{S}_{t,i}^{l,k}, \mathbf{G}^l)}_{\text{deeper supervision}}, \quad (6)$$

where $\mathcal{L}_C(\cdot)$ indicates the *cross-entropy* loss, which is widely used in SOD field. Our deeper supervision strategy improves the discriminativeness of each top-down and bottom-up inference layer. It also encourages the RNN unit to learn saliency representations faster and more efficiently, through the supervision of its intermediate outputs. This has never been explored in previous SOD models.

**Implementation details.** To demonstrate the generalization ability of our model, we use the conv parts of VGGNet [29] or ResNet50 [9] as our feature extractor. In our implementation, for simplicity, each conv kernels of the convRNN unit in the bottom-up/top-down inference layers is set as $3 \times 3$ with 16 channels. The channels of the input features of the bottom-up/top-down inference layers are uniformly compressed to 64. The *dimension-unifying* layer $\mathcal{R}$ is acquired through a $1 \times 1$ conv layer. For these layers, $RELU$ is used as the activation function. We set the total number of update steps $T$ of the RNN units to 2 and the total iteration steps $N$ of the top-down and bottom-up process to 3. In the next section, we will present more detailed experiments to explain the choices for the feature extraction network (*e.g*., VGGNet, ResNet50), the bottom-up/top-down inference layer variants (*e.g*., convRNN, convLSTM, convGRU), the total update steps $T$ and the iteration steps $N$.

Our model is fully differentiable and can thus be efficiently trained in an end-to-end manner. Following [45, 17, 34], we train our model using the THUS10K [5] dataset, which has $10,000$ images with pixel-wise annotations. The training images are uniformly resized to $224 \times 224$. Data augmentation techniques (*e.g*., flipping, cropping) are also performed. The networks are trained over 10 epochs, for a total of about 12 hours, using an Nvidia TITAN X GPU. The output $\mathbf{S}_{T,N}^{1,\text{top-down}} \in [0,1]^{224 \times 224}$ of the last top-down inference layer after the final update step $T$ and final iteration step $N$ is used as our final saliency result.

# 4. Experiments

## 4.1. Experimental Setup

**Datasets.** For the performance evaluation, we use six famous SOD benchmarks: ECCSD [42], DUT-OMRON [43], HKU-IS [18], PASCAL-S [21], SOD [25] and DUT-STE [33]. All these datasets are human-labeled with pixel-wise ground-truth for quantitative evaluations.

| Methods | ECCSD [42] | | DUT-OMRON [43] | | HKU-IS [18] | | PASCAL-S [21] | | SOD [25] | | DUTSTE [33] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ |
| VGGNet backbone | | | | | | | | | | | | |
| MDF [18] | .831 | .108 | .694 | .092 | .860 | .129 | .764 | .145 | .785 | .155 | .657 | .114 |
| LEGS [32] | .831 | .119 | .723 | .133 | .812 | .101 | .749 | .155 | .691 | .197 | .611 | .137 |
| ELD [17] | .865 | .080 | .700 | .092 | .844 | .071 | .767 | .121 | .760 | .154 | .697 | .092 |
| MC [48] | .822 | .107 | .702 | .088 | .781 | .098 | .721 | .147 | - | - | - | - |
| SU [14] | .88 | .06 | .68 | .07 | - | - | .77 | .10 | - | - | - | - |
| RFCN [34] | .898 | .109 | .701 | .111 | .895 | .089 | .827 | .118 | .805 | .161 | .752 | .090 |
| DHS [22] | .905 | .061 | - | - | .892 | .052 | .820 | .091 | .793 | .127 | .799 | .065 |
| KSR [36] | .801 | .133 | .742 | .157 | .759 | .120 | .649 | .137 | .698 | .199 | .660 | .123 |
| NLDF [24] | .905 | .063 | .753 | .080 | .902 | .048 | .831 | .112 | .808 | .130 | .777 | .066 |
| DLS [11] | .825 | .090 | .714 | .093 | .806 | .072 | .719 | .136 | - | - | - | - |
| AMU [45] | .889 | .059 | .733 | .097 | .918 | .052 | .834 | .103 | .773 | .145 | .750 | .085 |
| UCF [46] | .868 | .078 | .713 | .132 | .905 | .074 | .771 | .128 | .776 | .169 | .742 | .117 |
| FSN [4] | .910 | .053 | .741 | .073 | .895 | .044 | .827 | .095 | .781 | .127 | .761 | .066 |
| PICA [23] | .919 | .047 | .762 | .068 | .908 | .042 | .845 | .079 | .812 | .104 | .826 | .054 |
| DGRL [37] | .916 | .043 | .750 | .068 | .902 | **.037** | .837 | .076 | .805 | .106 | - | - |
| PAGR [47] | .904 | .061 | - | - | .897 | .048 | .815 | .094 | - | - | .807 | .059 |
| RAS [3] | .908 | .056 | .758 | .068 | .900 | .045 | .804 | .105 | .809 | .124 | .807 | .059 |
| C2S [20] | .902 | .054 | .731 | .080 | .887 | .046 | .834 | .082 | .786 | .124 | .783 | .062 |
| Ours (VGGNet) | **.921** | **.041** | **.770** | **.060** | **.919** | .040 | **.847** | **.073** | **.815** | **.102** | **.830** | **.050** |
| VGGNet backbone + CRF | | | | | | | | | | | | |
| DCL [19] | .898 | .071 | .732 | .087 | .907 | .048 | .822 | .108 | .784 | .126 | .742 | .150 |
| HEDS [10] | .915 | .053 | .714 | .093 | .913 | .040 | .830 | .112 | .802 | .126 | .799 | .065 |
| PICA-C [23] | .919 | .036 | .753 | .059 | .911 | .031 | .841 | .079 | .794 | .102 | .819 | .046 |
| Ours (CRF) | **.923** | **.032** | **.772** | **.055** | **.920** | **.030** | **.849** | **.072** | **.817** | **.098** | **.832** | **.045** |
| ResNet backbone | | | | | | | | | | | | |
| SRM [35] | .910 | .056 | .707 | .069 | .892 | .046 | .783 | .127 | .792 | .132 | .798 | .059 |
| PICA-R [23] | .924 | .047 | .773 | .065 | .906 | .043 | .844 | .087 | .817 | .109 | **.838** | .051 |
| Ours (ResNet) | **.926** | **.040** | **.780** | **.059** | **.920** | **.038** | **.848** | **.072** | **.820** | **.100** | .836 | **.048** |
| ResNet backbone + CRF | | | | | | | | | | | | |
| PICA-RC [23] | .929 | .035 | .772 | **.054** | .913 | .031 | .844 | .077 | .813 | .100 | **.840** | .041 |
| Ours (ResNet+CRF) | **.931** | **.032** | **.781** | .056 | **.920** | **.030** | **.848** | **.071** | **.821** | **.095** | **.840** | **.040** |

Table 1: Quantitative results with maximum F-measure and MAE on six widely used SOD benchmarks: ECCSD [42], DUT-OMRON [43], HKU-IS [18], PASCAL-S [21], SOD [25] and DUTSTE [33]. The leading entries in each group are **boldfaced**. See §4.2 for details.

**Evaluation metrics.** Two widely adopted metrics: F-measure and mean absolute error (MAE) are used in our experiments. See [2, 6] for more detailed definitions.

## 4.2. Performance Comparison

Our model is compared with 21 state-of-the-art deep SOD methods [18, 32, 19, 17, 48, 14, 34, 22, 10, 36, 24, 11, 45, 46, 35, 4, 23, 37, 44, 3, 20]. For a fair comparison, we use either the implementations with the recommended parameters or the saliency maps provided by the authors. [39] is excluded as it uses more training data.

**Quantitative evaluation.** Because some methods [19, 10, 23] are post-processed using fully connected conditional random field (CRF) or different backbones (*e.g.*, VGGNet [29], ResNet50 [9]), we also report our results with these settings. In Table 1, we show the results of quantitative comparisons for the Maximum F-measure and MAE. We observe that our model consistently outperforms all other competitors, across all metrics.

**Qualitative evaluation.** Results for qualitative comparisons are given in Fig. 4; showing the proposed model is applicable to several challenging scenes well, including large foregrounds, complex backgrounds and objects with similar appearances to backgrounds, *etc*.

| Methods | SOD [25] | | DUTSTE [33] | |
|---|---|---|---|---|
| | F-score ↑ | MAE ↓ | F-score ↑ | MAE ↓ |
| VGGNet+CNN | .809 | .109 | .821 | .058 |
| VGGNet+convRNN | .814 | **.099** | .828 | .053 |
| VGGNet+convLSTM | .811 | .106 | .825 | .056 |
| VGGNet+convGRU | **.815** | .102 | **.830** | **.050** |

Table 2: Ablation study for different implementations of top-down and bottom-up saliency inference layers. VGGNet is used as the backbone. The update stages of the RNN units are uniformly set as 2. The iteration of top-down and bottom-up saliency inference is performed 3 times.

## 4.3. Ablation Studies

In this section, we perform extensive ablation studies to comprehensively assess the effectiveness of each essential component, as well as the overall performance of the different variations and implementations of our model.

**Feature extractor network.** From Table 1 we can observe that the model with the ResNet backbone outperforms the one built upon VGG-16.

**Different implementations of top-down and bottom-**

| Side-outputs | N=1 | | N=2 | | N=3 | | N=4 | | N=5 |
|---|---|---|---|---|---|---|---|---|---|
| | T-D | B-U | T-D | B-U | T-D | B-U | T-D | B-U | T-D |
| $S^5$ (*conv 5*) | .133 | .116 | - | .098 | - | .096 | - | .097 | - |
| $S^4$ (*conv 4*) | .108 | .093 | .086 | .080 | .076 | .076 | .078 | .080 | .081 |
| $S^3$ (*conv 3*) | .086 | .081 | .075 | .068 | .063 | .064 | .066 | .066 | .068 |
| $S^2$ (*conv 2*) | .071 | .068 | .064 | .061 | .059 | .058 | .059 | .060 | .062 |
| $S^1$ (*conv 1*) | .062 | - | .054 | - | .050 | - | .051 | - | .053 |

Table 3: Ablation study for the side-outputs using different iterations of joint top-down and bottom-up inference. Performance is evaluated on DUTSTE [33] with MAE. VGGNet + convGRU is used as the basic setting.

| Update Steps of RNN units | SOD [25] | | DUTSTE [33] | |
|---|---|---|---|---|
| | F-score | MAE | F-score | MAE |
| T =1 | .810 | .112 | .819 | .060 |
| T =2 | .815 | **.102** | **.830** | **.050** |
| T =3 | **.817** | .105 | .828 | .053 |
| T =4 | .812 | .111 | .823 | .056 |

Table 4: Ablation study for various update steps of the RNN units of the top-down/bottom-up inference layers. VGGNet+convGRU+($N=3$) is used as the basic setting.

**up saliency inference layers.** Next, we study the performance of different implementations of our top-down and bottom-up saliency inference layers. Based on the discussions in §3.2, we derive four baselines: *VGGNet+CNN*, *VGGNet+convRNN*, *VGGNet+convLSTM*, and *VGGNet+convGRU*, using a VGGNet backbone. From Table 2 we find that the RNN variants perform better than the one based on CNNs. This is because RNN units introduce a feed-back mechanism into each inference layer, enabling intra-layer step-wise optimization, while CNN constituents only allow feed-forward estimation. We also find slight changes in performance among the different RNN variants. The best choice appears to be the basic RNN or the GRU. This is perhaps because there is no need to model long-term dependency. It is also more difficult to train the LSTM due to its larger number of parameters. In the following sections, we apply convGRU as the implementations for the top-down/bottom-up saliency inference layers.

**Effectiveness of iterative and cooperative top-down and bottom-up saliency inference.** To demonstrate our central notion that iterative top-down and bottom-up saliency inference can enhance performance, we present the MAE scores of different side-outputs during different iteration steps in Table 3. It is clearly shown that, within each top-down (*T-D*) and bottom-up (*B-U*) iteration, the saliency estimates from the *T-D* are improved through the use of *B-U*. Additionally, the results gradually improve with more iterations ($N \leq 3$).

**Side-outputs.** When we look at the statistics in Table 3, we find that, with upper-layer guidance, the inferior layers gradually produce better results, demonstrating the effectiveness of our coarse-to-fine process.

**Update steps of the RNN in the top-down/bottom-up saliency inference layers.** We now assess the performance using different update steps for the RNN in the top-down/bottom-up inference layers. From Table 4, we see that increasing the number of update steps $T$ for the RNN units (up to 2 or 3 steps) can lead to enhanced performance. Additionally, when $T = 1$, the corresponding convGRU-based top-down/bottom-up saliency inference layers can be viewed as a CNN-based implementation.

| Parameter Sharing | Model Size (MB) | SOD [25] | | DUTSTE [33] | |
|---|---|---|---|---|---|
| | | F-score | MAE | F-score | MAE |
| Strategy *i* (Fig. 3 (d)) | 448 | **.817** | .103 | **.832** | .051 |
| Strategy *ii* (Fig. 3 (e)) | 402 | .815 | **.102** | .830 | **.050** |
| Strategy *iii* (Fig. 3 (f)) | 375 | .812 | .104 | .825 | .053 |
| Strategy *iv* (Fig. 3 (g)) | 364 | .807 | .103 | .828 | .056 |

Table 5: Ablation study for different parameter-sharing strategies. VGGNet+convGRU+($N$=3)+($T$=2) is used as the basic setting.

| Methods | SOD [25] | | DUTSTE [33] | |
|---|---|---|---|---|
| | F-score | MAE | F-score | MAE |
| VGGNet+convGRU+deeper sup. | **.815** | **.102** | **.830** | **.050** |
| *w/o* deeper sup. | .807 | .106 | .823 | .054 |

Table 6: Ablation study for the deeper supervision strategy using a VGGNet+convGRU setting.

**Parameter-sharing.** In §3.1 and Fig. 3 (b)-(e), we discussed several possible ways of introducing parameter-sharing. We report the performance for each strategy in Table 5. As can be seen, the model with *strategy i* shows relatively high performance, but has the largest number of parameters, while *strategy iv* is the most light-weight. The model with *strategy ii* achieves the best trade-off between performance and model size.

**Deeper supervision.** Our model is trained using a deeper supervision strategy (detailed in §3.2). In other words, for each side-out layer, supervisions are fed into each RNN update-step outputs during each top-down/bottom-up iteration. Table 6 shows that such a strategy enhances performance. This is because the errors in the loss can be directly back-propagated into each iteration of the top-down and bottom-up inferences and update steps of the RNN units.

## 5. Conclusion

We proposed a unified framework, the iterative top-down and bottom-up inference network, for salient object detection. The model learns top-down, coarse-to-fine saliency inference and bottom-up, shallow-to-deep saliency inference in an iterative and end-to-end manner. Most importantly, the two processes work in a cooperative way, *i.e.*, the finest saliency from the top-down process is used to guide higher-layer saliency estimates in a bottom-up manner, while the refined top-layer saliency further promotes an efficient top-down process. We studied several variations of the model and parameter-sharing strategies in depth. Extensive results showed the superior performance of our model.

**Acknowledgements** This work was supported in part by NSFC (61572264) and Tianjin Natural Science Foundation (17JCJQJC43700, 18ZXZNGX00110).

## References


[1] Ali Borji. Boosting bottom-up and top-down visual features for saliency estimation. In *CVPR*, 2012. 2

[2] Ali Borji, Ming-Ming Cheng, Huaizu Jiang, and Jia Li. Salient object detection: A benchmark. *IEEE TIP*, 12(24):5706–5722, 2015. 7

[3] Shuhan Chen, Xiuli Tan, Ben Wang, and Xuelong Hu. Reverse attention for salient object detection. In *ECCV*, 2018. 2, 3, 6, 7

[4] Xiaowu Chen, Anlin Zheng, Jia Li, and Feng Lu. Look, perceive and segment: Finding the salient objects in images via two-stream fixation-semantic cnns. In *ICCV*, 2017. 2, 6, 7

[5] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE TPAMI*, 37(3):569–582, 2015. 6

[6] Deng-Ping Fan, Ming-Ming Cheng, Jiang-Jiang Liu, Shang-Hua Gao, Qibin Hou, and Ali Borji. Salient objects in clutter: Bringing salient object detection to the foreground. In *ECCV*, pages 186–202, 2018. 7

[7] James Jerome Gibson. The senses considered as perceptual systems. 1966. 1

[8] Richard Langton Gregory. The intelligent eye. 1970. 1

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 6, 7

[10] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip Torr. Deeply supervised salient object detection with short connections. In *CVPR*, 2017. 2, 3, 6, 7

[11] Ping Hu, Bing Shuai, Jun Liu, and Gang Wang. Deep level sets for salient object detection. In *CVPR*, 2017. 2, 3, 7

[12] Fumi Katsuki and Christos Constantinidis. Bottom-up and top-down attention: Different processes and overlapping neural systems. *The Neuroscientist*, 20(5):509–521, 2014. 2

[13] Min-Shik Kim and Kyle R Cave. Top-down and bottom-up attentional control: On the nature of interference from a salient distractor. *Perception & Psychophysics*, 61(6):1009–1023, 1999. 2

[14] Srinivas SS Kruthiventi, Vennela Gudisa, Jaley H Dholakiya, and R Venkatesh Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *CVPR*, 2016. 2, 7

[15] Jason Kuen, Zhenhua Wang, and Gang Wang. Recurrent attentional networks for saliency detection. In *CVPR*, 2016. 2, 3, 5

[16] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *AISTATS*, 2015. 6

[17] Gayoung Lee, Yu-Wing Tai, and Junmo Kim. Deep saliency with encoded low level distance map and high level features. In *CVPR*, 2016. 2, 6, 7

[18] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *CVPR*, 2015. 2, 6, 7

[19] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *CVPR*, 2016. 2, 7

[20] Xin Li, Fan Yang, Hong Cheng, Wei Liu, and Dinggang Shen. Contour knowledge transfer for salient object detection. In *ECCV*, 2018. 2, 3, 7

[21] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *CVPR*, 2014. 2, 6, 7

[22] Nian Liu and Junwei Han. DHSNet: Deep hierarchical saliency network for salient object detection. In *CVPR*, 2016. 2, 3, 5, 7

[23] Nian Liu, Junwei Han, and Ming-Hsuan Yang. PiCANet: Learning pixel-wise contextual attention for saliency detection. In *CVPR*, 2018. 2, 3, 6, 7

[24] Zhiming Luo, Akshaya Mishra, Andrew Achkar, Justin Eichel, Shaozi Li, and Pierre-Marc Jodoin. Non-local deep features for salient object detection. In *CVPR*, 2017. 2, 3, 7

[25] Vida Movahedi and James H Elder. Design and perceptual validation of performance measures for salient object segmentation. In *CVPR - Workshops*, 2010. 2, 6, 7, 8

[26] Junting Pan, Elisa Sayrol, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E O'Connor. Shallow and deep convolutional networks for saliency prediction. In *CVPR*, 2016. 2, 3

[27] Pedro O Pinheiro, Tsung-Yi Lin, Ronan Collobert, and Piotr Dollár. Learning to refine object segments. In *ECCV*, 2016. 3

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015. 3

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3, 6, 7

[30] Hongmei Song, Wenguan Wang, Sanyuan Zhao, Jianbing Shen, and Kin-Man Lam. Pyramid dilated deeper convlstm for video salient object detection. In *ECCV*, pages 715–731, 2018. 3

[31] Huawei Tian, Yuming Fang, Yao Zhao, Weisi Lin, Rongrong Ni, and Zhenfeng Zhu. Salient region detection by fusing bottom-up and top-down features extracted from a single image. *IEEE TIP*, 23(10):4389–4398, 2014. 2

[32] Lijun Wang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Deep networks for saliency detection via local estimation and global search. In *CVPR*, 2015. 2, 7

[33] 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 *CVPR*, 2017. 2, 6, 7, 8

[34] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan. Saliency detection with recurrent fully convolutional networks. In *ECCV*, 2016. 2, 3, 5, 6, 7

[35] Tiantian Wang, Ali Borji, Lihe Zhang, Pingping Zhang, and Huchuan Lu. A stagewise refinement model for detecting salient objects in images. In *ICCV*, 2017. 2, 3, 5, 7

[36] Tiantian Wang, Lihe Zhang, Huchuan Lu, Chong Sun, and Jinqing Qi. Kernelized subspace ranking for saliency detection. In *ECCV*, 2016. 7

[37] Tiantian Wang, Lihe Zhang, Shuo Wang, Huchuan Lu, Gang Yang, Xiang Ruan, and Ali Borji. Detect globally, refine locally: A novel approach to saliency detection. In *CVPR*, 2018. 2, 3, 7

[38] Wenguan Wang and Jianbing Shen. Deep visual attention prediction. *IEEE TIP*, 27(5):2368–2378, 2018. 3

[39] Wenguan Wang, Jianbing Shen, Xingping Dong, and Ali Borji. Salient object detection driven by fixation prediction. In *CVPR*, 2018. 2, 3, 5, 7

[40] Wenguan Wang, Jianbing Shen, Fang Guo, Ming-Ming Cheng, and Ali Borji. Revisiting video saliency: A large-scale benchmark and a new model. In *CVPR*, 2018. 3

[41] Wenguan Wang, Jianbing Shen, and Ling Shao. Video salient object detection via fully convolutional networks. *IEEE TIP*, 27(1):38–49, 2018. 2, 3

[42] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *CVPR*, 2013. 2, 6, 7

[43] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, 2013. 2, 6, 7

[44] Lu Zhang, Ju Dai, Huchuan Lu, You He, and Gang Wang. A bi-directional message passing model for salient object detection. In *CVPR*, 2018. 2, 3, 7

[45] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Xiang Ruan. Amulet: Aggregating multi-level convolutional features for salient object detection. In *ICCV*, 2017. 2, 3, 6, 7

[46] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Baocai Yin. Learning uncertain convolutional features for accurate saliency detection. In *ICCV*, 2017. 2, 3, 6, 7

[47] Xiaoning Zhang, Tiantian Wang, Jinqing Qi, Huchuan Lu, and Gang Wang. Progressive attention guided recurrent network for salient object detection. In *CVPR*, pages 714–722, 2018. 6, 7

[48] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015. 2, 7