# Lightweight Salient Object Detection via Hierarchical Visual Perception Learning

Yun Liu*, Yu-Chao Gu*, Xin-Yu Zhang*, Weiwei Wang, Ming-Ming Cheng

*Abstract*—**Recently, salient object detection (SOD) has witnessed vast progress with the rapid development of convolutional neural networks (CNNs). However, the improvement of SOD accuracy comes with the increase in network depth and width, resulting in large network size and heavy computational overhead. This prevents state-of-the-art SOD methods from being deployed into practical platforms, especially mobile devices. To promote the deployment of real-world SOD applications, we aim at developing a lightweight SOD model in this paper. Our observation comes from that the primate visual system processes visual signals hierarchically with different receptive fields and eccentricities in different visual cortex areas. Inspired by this, we propose a Hierarchical Visual Perception (HVP) module to imitate the primate visual cortex for hierarchical perception learning. With the HVP module incorporated, we design a lightweight SOD network, namely HVPNet. Extensive experiments on popular benchmarks demonstrate that HVPNet achieves highly competitive accuracy compared with state-of-the-art SOD methods while running at a 4.3fps CPU speed and a 333.2fps GPU speed with only 1.23M parameters.**

*Index Terms*—**Lightweight salient object detection, lightweight saliency detection, hierarchical visual perception.**

## I. Introduction

THE human vision system can detect the most arresting objects or regions in natural images rapidly and automatically. Salient object detection (SOD) aims at imitating such a human instinct to capture the most eye-catching area in an image. The progress in SOD has benefited a broad range of computer vision applications, including object detection [1], image retrieval [2], visual tracking [3], image thumbnailing [4], *etc*. Conventional SOD methods [5], [6] mainly rely on hand-crafted low-level features. In spite of the efficiency, the lack of representation capacity for high-level semantics makes these methods difficult to model complicated natural scenes. Due to the powerful capacity in representation learning, convolutional neural networks (CNNs), especially fully convolutional networks (FCNs), have dominated this field. Numerous CNN- and FCN-based SOD approaches [7]–[22] have pushed the state of the art forward.

However, the accuracy improvement is not free. Traditional SOD requires a strong backbone (*i.e.,* encoder) to capture both low-level fine-grained details and high-level semantic features, and a carefully calibrated decoder to recover the spatial resolution without losing spatial information, both of which may bring tremendous parameters and computational overhead [7]–[18], [22]–[32]. On the contrary, recent increased interest in mobile applications, such as cell phones, where the computational capacity, memory space, and energy support are limited, cannot deploy these large SOD models. The cost of deploying these large models on the servers is also high. This inspires us to consider the efficiency and the number of network parameters as important as the accuracy in the evaluation of SOD methods.

With the aforementioned consideration, we aim at designing a lightweight SOD model to promote practical SOD systems. Although lightweight network architecture has been studied in other vision tasks, such as image classification [33]–[36], directly applying lightweight backbone networks for SOD leads to suboptimal performance. This is because SOD has special requirements in multi-scale learning as described above, while lightweight backbone networks, such as MobileNets [33], [34] and ShuffleNets [35], [36], focus on capturing high-level semantics and are less powerful in multi-scale learning than traditional large networks that are deeper, wider, and more in the number of convolution filters. Therefore, lightweight SOD is still a challenging problem, and the key is how to effectively learn multi-scale contexts in a lightweight setting.

We get inspiration from the primate visual system to tackle this problem because modeling human visual perception for scene interpretation is a strong trend in computer vision [37]. About 55 percent of the neocortex of the primate brain is associated with vision [38], and the processing pipeline is in a hierarchical structure [39]–[41]. Multi-scale visual signals are hierarchically processed in different cortex areas that have different population receptive fields (pRFs) [42]. Wandell *et al.* [42] found that the pRF size increases with eccentricity in retinotopic maps. A recent study [43] attempts to simulate the size and eccentricity of pRF using the kernel size and dilation rate of the convolution layer, respectively, so that the kernel size and dilation rate have a similar positive functional relation as that of the size and eccentricity of pRF. A simple way to simulate the primate visual system is the parallel organization of various pRF. However, this ignores the visual hierarchy in the visual cortex, which has been studied in the conventional computer vision, *i.e.,* before deep learning [44], [45]. In this paper, we propose the Hierarchical Visual Perception (HVP) module to simulate the structure of the primate visual cortex. The HVP module uses a densely-connected structure to imitate the visual hierarchies and dilated convolution to imitate the

This work was supported in part by the Major Project for New Generation of AI under Grant 2018AAA0100400; in part by NSFC under Grant 61922046; in part by the National Youth Talent Support Program; and in part by the Tianjin Natural Science Foundation under Grant 17JCJQJC43700. *(Yun Liu, Yu-Chao Gu, and Xin-Yu Zhang contributed equally to this work.) (Corresponding author: Ming-Ming Cheng.)*

Yun Liu, Yu-Chao Gu, and Ming-Ming Cheng are with the College of Computer Science, Nankai University, Tianjin 300350, China (e-mail: cmm@nankai.edu.cn).

Xin-Yu Zhang and Weiwei Wang are with the School of Mathematical Science, Nankai University, Tianjin 300350, China.

pRF. Experimental results suggest that using the kernel sizes and dilation rates in the descending order performs best, which is consistent with Hochstein and Ahissar's Reverse Hierarchy Theory (RHT) [40] that claims visual perception begins at the higher levels and travels to the lower areas. With the HVP module and attention mechanism incorporated, we design a lightweight SOD network, namely HVPNet. Extensive evaluation on popular benchmarks demonstrates that HVPNet with only 1.23M parameters achieves highly competitive accuracy compared with state-of-the-art methods while running at a CPU speed of 4.3fps and a GPU speed of 333.2fps for $336 \times 336$ images.

We summarize our contribution as follows:

- We propose a novel Hierarchical Visual Perception (HVP) module to imitate the primate visual hierarchies for better multi-scale learning.
- With the HVP module and attention mechanism incorporated, we design HVPNet that is the first lightweight SOD network as we know.
- We conduct extensive experiments to investigate and evaluate the proposed HVPNet that thus serves as a strong baseline for future lightweight SOD research.

## II. Related Work

In this section, we first summarize recent progress in SOD and then review literature about lightweight deep learning.

*a) Salient Object Detection:* Traditional SOD methods [5], [6], [46]–[48] mainly rely on hand-crafted features and heuristic priors. Due to the restricted representation capacity, hand-crafted features have been gradually replaced by deep learning. Thanks to the powerful capacity of feature representation learning of CNNs and FCNs, this field has witnessed the burst of numerous CNN- and FCN-based methods [7]–[12], [14], [15], [17]–[32], [49]–[53] in the last five years.

Most efforts of these deep models are on how to effectively fuse the multi-scale information of various side-outputs [54], [55]. Some methods [56]–[60] directly concatenate or sum the side-output features. Some methods [61], [62] perform saliency prediction using side-output features and then fuse all side-output prediction to obtain the final saliency map. Most methods [7]–[20], [22]–[31] use the encoder-decoder structure, in which the encoder is usually backbone networks for image classification [63], and the decoder is responsible for side-output feature fusion. Some clever designs have attracted much attention in this field. For example, PiCANet [12] proposes to apply bidirectional LSTM to compute global contexts. RAS [61] presents a reverse attention mechanism to manage side-outputs in a top-down way.

While the accuracy has been improved with the increase in network depth and width, the substantial computational overhead and large network size have hindered state-of-the-art SOD methods from being deployed into practical systems, especially for mobile devices. For example, the recent state-of-the-art method, EGNet [64], has 108M parameters, which exceeds the tolerance of most mobile devices. Instead of continuing to go in this direction, in this paper, we pave a new path for SOD, *i.e.,* lightweight SOD, which has great potential to promote SOD into more practical applications. Our proposed HVPNet performs comparably with state-of-the-art methods while maintaining high efficiency and small network size.

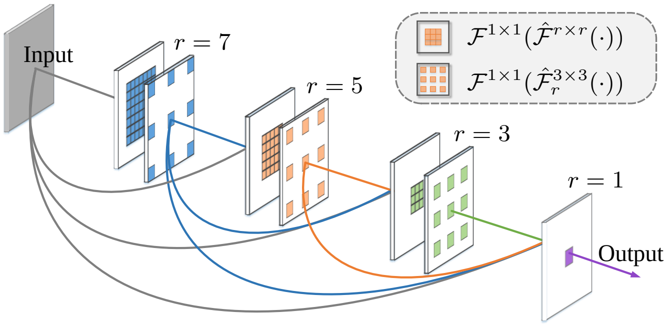
Input
$r = 7$
$r = 5$
$r = 3$
$r = 1$
Output
$\mathcal{F}^{1\times1}(\hat{\mathcal{F}}^{r\times r}(\cdot))$
$\mathcal{F}^{1\times1}(\hat{\mathcal{F}}_r^{3\times3}(\cdot))$

Fig. 1. Illustration of the proposed HVP module.

*b) Lightweight Neural Networks:* In many real-world applications, visual recognition tasks must be carried out in a timely, power-saving, and memory-friendly fashion with computational resource constraints. Although it has not been brought into SOD, many other vision tasks have built lightweight models to satisfy real-world requirements using weight quantization [65], [66], network compression [67], [68], computationally efficient architecture design [33]–[36], *etc*. Notably, for some vision tasks, such as image classification [33]–[36], lightweight networks have shown their superiority by reducing the model size and floating-point operations (FLOPs) with a little performance drop. MobileNets [33], [34] adopt depth-wise separable convolutions to approximate the representation ability of regular convolutions with significantly reduced parameters. Based on the depth-wise convolution, ShuffleNets [35], [36] utilize a channel shuffle operation to further reduce the redundancy of point-wise convolutions. We share the same spirits with prior arts [33]–[36] to build our model using depth-wise separable convolutions, while our main technical contribution comes from the observation about the hierarchical primate visual system. We propose the HVP module to imitate the primate visual hierarchies and pRF. We also explore the attention mechanism for further performance improvement. With these components incorporated, the proposed HVPNet achieves comparable performance with state-of-the-art methods while in an extremely lightweight setting.

## III. Methodology

In this section, we elaborate on our lightweight SOD network architecture. Concretely, we introduce our motivation from the primate visual system in Section III-A. Then, we present the primary building block, namely, Hierarchical Visual Perception (HVP) module in Section III-B. Other network components and the overall architecture are summarized in Section III-C and Section III-D, respectively.

### A. Motivation and Principles

Lots of neurophysiological evidence suggests that a sequence of different levels of signal processing (8 to 10 levels) constitute the hierarchical signal processing in the primate visual system [39]–[41]. The hierarchical processing exhibits straightforward superiority over the so-called flat processing that processes signals in a parallel way [45]. In fact, there is a large amount of neurophysiological evidence that cognition is associated with the concept of the deep hierarchy [69]. This is intuitive because our eye does not perceive all contents in a natural scene at first glance but recognize objects with the highest contrast to their surroundings first, which is a simple understanding of the visual hierarchy. The capacity of the primate visual system to process information at hierarchical levels has inspired computer vision research. Please refer to [70] for a summation.

On the other hand, neurons in different cortex areas have different population Receptive Field (pRF) sizes, and the pRF size increases with eccentricity in each retinotopic map [42]. The impact of the eccentricity of pRF in the visual system can be simulated by dilated convolutions [43]. Specifically, we can imitate pRF size with the kernel size and the eccentricity with dilation rate, so that the kernel size and dilation rate have the same positive functional relation as that of the size and eccentricity of pRF. However, this simple flat processing [45] for feature learning from different pRFs, *i.e.,* with parallel connections, is suboptimal, because it ignores the basic concept of the deep hierarchy in the primate visual system. In the experiments, we will demonstrate the design of the parallel connection is suboptimal for lightweight SOD.

In this paper, we propose a more realistic approach to mimic the primate visual system. We still use dilated convolution to imitate the pRF. In order to imitate the positive correlation between the size and eccentricity of pRF, a large dilation rate will correspond to the large kernel size. Instead of using simple parallel connections, we adopt serial connections for different pRFs. Since the practical connections of the primate visual system are involved, without an exact connection order, we propose to impose dense connections for different pRFs so that the output feature of one pRF will serve as the input for all of the following pRFs. Moreover, Hochstein and Ahissar's *reverse hierarchy theory* (RHT) [40] claims that the visual system first generates perception at higher levels and then travels to low levels, which means that visual attention works in a coarse-to-fine way. Hence the proposed HVP module puts large kernel sizes and dilation rates at the beginning to capture the high-level information (with large pRF). Intuitively, the primate visual system in a pre-attentive vision sends the information to interpret the scene at a glance, *i.e.,* only large details. Experimental results show that arranging kernel sizes and dilation rates in the descending order outperforms other orders, which demonstrates our hypothesis about HVP and RHT. Therefore, the proposed HVP module is not only theoretically but also experimentally reasonable.

### *B. Hierarchical Visual Perception Module*

With the principles described above, we continue by elaborating on the proposed HVP module. As shown in Fig. 1, we adopt dilated convolutions to imitate different visual cortex areas that have different pRFs whose size and eccentricities have similar relation to the kernel size and dilation rate of convolution. Here, we use depth-wise separable convolution (DSConv) [33] and point-wise convolution (*i.e.,* the vanilla $1 \times 1$ convolution) as the atomic operations to reduce parameters and computational load. Let $\mathcal{F}^{k\times k}$ be the vanilla convolution with the kernel size of $k \times k$. For example, $\mathcal{F}^{1\times 1}$ is the vanilla $1 \times 1$ convolution. Suppose that $\hat{\mathcal{F}}_d^{k\times k}$ denotes a DSConv with the kernel size of $k \times k$ and the dilation rate of $d$, and we omit the subscript for $d = 1$, *i.e.,* $\hat{\mathcal{F}}_1^{k\times k} = \hat{\mathcal{F}}^{k\times k}$.

Each simulation unit for pRF is composed of a DSConv with the kernel size of $r$ and a DSConv with the dilation rate of $r$, which can be formulated as

$$\mathcal{R}_r(\boldsymbol{X}) = \begin{cases} \mathcal{F}^{1\times 1}(\boldsymbol{X}), & \textit{if } r = 1; \\ \mathcal{F}^{1\times 1}(\hat{\mathcal{F}}_r^{3\times 3}(\mathcal{F}^{1\times 1}(\hat{\mathcal{F}}^{r\times r}(\boldsymbol{X})))), & \textit{if } r > 1, \end{cases} \tag{1}$$

where standard batch normalization [71] and PReLU [72] layers are connected after each convolution layer. Here, we imitate the pRF size with the kernel size of $\hat{\mathcal{F}}^{r\times r}$ and the eccentricity with the dilation rate of $\hat{\mathcal{F}}_r^{3\times 3}$, so that the pRF size and the pRF eccentricity have the same positive functional relation. Note that we use two convolutions of $\hat{\mathcal{F}}^{r\times r}$ and $\hat{\mathcal{F}}_r^{3\times 3}$, not a single convolution of $\hat{\mathcal{F}}_r^{r\times r}$, because $\hat{\mathcal{F}}_r^{r\times r}$ would have large sparse convolution kernels (*e.g.,* $r > 3$) that is suboptimal for network training [73] and inefficient for network inference. Applying Eq. (1) with different values of $r$, we can imitate different areas on the primate visual cortex, *e.g.,* the occipital areas V1 to hV4 whose sizes and eccentricities of pRF gradually increase. In the concept of deep learning, we can learn multi-scale information with various receptive fields in this way.

As discussed in Section III-A, the processing of different visual cortex areas is organized in a hierarchical manner. Instead of using "flat" processing as in the existing computer vision systems [74], [75], we propose to use hierarchical processing. Specifically, we connect the simulation units for different pRFs in a serial way. Besides, connections on the visual cortex are very complicated, and one area is not only connected to one other area. Hence, we apply dense connections [76] for pRF simulation units to mimic the complex connections on the visual cortex so that the output of one pRF will be viewed as the input signals for all of the following pRFs. Formally, the output responses of all preceding pRF units are concatenated to serve as the input for the next unit, *i.e.,*

$$\boldsymbol{X}_i = \mathcal{R}_{r_i}(\mathrm{Concat}(\boldsymbol{X}_0, \boldsymbol{X}_1, \cdots, \boldsymbol{X}_{i-1})), 1 \le i \le N, \tag{2}$$

where $N$ is the number of pRF units, and $\boldsymbol{X}_0$ ($\boldsymbol{X}_i$ with $i = 0$) denotes the input for an HVP module. $\mathrm{Concat}(\cdot)$ represents the concatenation operation. The number of output channels and convolution groups of the first DSConv $\hat{\mathcal{F}}^{r_i\times r_i}(\cdot)$ in $\mathcal{R}_{r_i}$ is equal to the number of channels of $\boldsymbol{X}_0$. Other convolutions in $\mathcal{R}_{r_i}$ have the same number of channels as $\boldsymbol{X}_0$. From the perspective of deep learning, dense connections bring increased depth and more powerful representation space that leads to better performance.

The last problem is how to decide the order of pRFs. We follow the *reverse hierarchy theory* [40] to first generate visual perception at large pRFs and then flow the perception into small pRFs. Therefore, we use a descending order of kernel sizes/eccentricities of convolutions for imitating pRFs. For example, we adopt a sequence of $r$ values of $7, 5, 3, 1$ for each HVP module. Intuitively, human eyes usually see large objects (large pRFs) at first glance and then gradually focalize to fine-grained image details. This is also similar to the principle of hierarchical representations used for image retrieval and scalable indexes [77]. Our experimental results also demonstrate that the descending order performs better than any other order.

### C. Attention and Dropout Mechanism

The attention mechanism has been demonstrated to be effective [10], [12], [17], [60], [61], [78]. Instead of only using spatial attention to adaptively highlight or suppress certain locations as in these methods, we further include channel attention to explore inter-channel dependencies and re-calibrate channel activation. On the other hand, we insert the attention mechanism in the encoder network, while the above previous methods adopt attention in the decoder because their encoder is usually fixed to existing backbone networks.

*a) Channel Attention:* The channel attention mechanism is first introduced in [79]. Let $\boldsymbol{X} \in \mathbb{R}^{C \times H \times W}$ be the input activation, in which $C$, $H$, and $W$ are the number of channels, height, and width, respectively. We first apply global average pooling (GAP) to extract the channel-wise representations, *i.e.,*

$$\boldsymbol{d}_c = GAP(\boldsymbol{X}) = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \boldsymbol{X}_{c,i,j}, \tag{3}$$

in which $\boldsymbol{d}_c$ is the $c$-th value of the feature vector $\boldsymbol{d} \in \mathbb{R}^C$, and $\boldsymbol{X}_{c,i,j}$ is the value of $\boldsymbol{X}$ at coordinates $(c, i, j)$. Then, we employ a simple soft-gating mechanism to calculate the per-channel importance, namely,

$$\hat{\boldsymbol{d}} = \sigma(\mathcal{F}^{1\times 1}(\psi(\mathcal{F}^{1\times 1}(\boldsymbol{d})))), \tag{4}$$

where the inner and outer $1 \times 1$ convolutions have $\frac{C}{r}$ and $C$ output channels, respectively. Here, $r$ denotes the rate for channel reduction. Hence, we have $\hat{\boldsymbol{d}} \in \mathbb{R}^C$. $\psi$ refers to the standard nonlinear activation function [80], and $\sigma$ is the *Sigmoid* soft-gating function. Afterwards, channel activation is re-calibrated in a multiplicative manner, *i.e.,*

$$\widetilde{\boldsymbol{X}} = \hat{\boldsymbol{d}} \otimes \boldsymbol{X}, \tag{5}$$

where $\hat{\boldsymbol{d}}$ is duplicated into the size of $C \times H \times W$, and $\otimes$ indicates element-wise multiplication.

*b) Spatial Attention:* Given the re-calibrated features $\widetilde{\boldsymbol{X}}$, we extract the pixel-wise importance based on local responses. Our operation is computationally efficient for the requirement of lightweight SOD. Concretely, we adopt a simple $k \times k$ convolution with a single output channel, and again use soft-gating mechanism (*i.e., Sigmoid*) to compute the spatial multipliers. Mathematically, we have

$$\boldsymbol{v} = \sigma(\mathcal{F}^{k\times k}(\widetilde{\boldsymbol{X}})), \tag{6}$$

where we have $\boldsymbol{v} \in \mathbb{R}^{H\times W}$. Similarly, the spatially re-calibrated activation are formulated as

$$\widehat{\boldsymbol{X}} = \boldsymbol{v} \otimes \widetilde{\boldsymbol{X}}, \tag{7}$$

TABLE I

ENCODER CONFIGURATIONS OF THE PROPOSED LIGHTWEIGHT SOD MODEL. "MODULE", "#M", "#F", "K", AND "S" REPRESENT THE MODULE TYPE, THE NUMBER OF MODULES, THE NUMBER OF CONVOLUTION FILTERS, KERNEL SIZE, AND STRIDE, RESPECTIVELY. "RESATT" REFERS TO THE RESIDUAL ATTENTION IN SECTION III-C.

| Stage | Resolution | Module | #M | #F | K | S |
|---|---|---|---|---|---|---|
| 1 | $224 \times 224$ | Conv | 1 | 16 | 3 | 2 |
| | $112 \times 112$ | Conv & ResAtt | 1 | 16 | 3 | 1 |
| 2 | $112 \times 112$ | DSConv | 1 | 32 | 5 | 2 |
| | $56 \times 56$ | HVP & ResAtt | 1 | 32 | 7-5-3-1 | 1 |
| 3 | $56 \times 56$ | DSConv | 1 | 64 | 5 | 2 |
| | $28 \times 28$ | HVP & ResAtt | 3 | 64 | 7-5-3-1 | 1 |
| 4 | $28 \times 28$ | DSConv | 1 | 128 | 5 | 2 |
| | $14 \times 14$ | HVP & ResAtt | 5 | 128 | 7-5-3-1 | 1 |

where $\boldsymbol{v}$ is duplicated to the size of $C \times H \times W$ before multiplication.

*c) Residual Attention:* As we employ attention mechanism sequentially and iteratively, multiplying by factors within the range of $(0, 1)$ would weaken the activation gradually, leading to vanishing gradients. To this end, we employ residual learning [63] to facilitate gradient propagation. Finally, the output activation become

$$\boldsymbol{Y} = \widehat{\boldsymbol{X}} + \boldsymbol{X}. \tag{8}$$

*d) Dropout:* Overfitting is always a pesky problem in deep learning. The strategy of dropping CNN activation is shown to be useful in increasing the generalization capability and avoiding overfitting [81], [82]. In this paper, we connect a standard dropout layer [81] with a dropout rate of 0.1 before each HVP module in training. Unlike the recent attention-based dropout strategy [82] that drops activation according to the computed saliency map, the attention and dropout in our method are independent for a fair comparison with previous SOD methods. In other words, the exploration of the new dropout strategy is out of the scope of this paper, so we follow previous literature to use the standard dropout layer [81] in this paper. In testing, the dropout layers are directly abandoned.

### D. Network Architecture

With the aforementioned components, we build an encoder-decoder network with lateral connections, namely HVPNet. For the encoder, we stack the proposed HVP modules for fast deep feature extraction in a bottom-up manner. For the decoder, we use a simple method to integrate the high-level semantic features and low-level fine-grained details in a top-down way. The details of our design are introduced as follows.

*a) Encoder Network:* Our encoder consists of 4 stages, and the default configurations for each stage are summarized in Table I. At the $s$-th stage, the input activation $\boldsymbol{F}_{s-1}$ is first downsampled by a (depth-wise separable or vanilla) convolution with stride 2, which is formulated as

$$\boldsymbol{F}_s = \mathrm{Concat}(\mathcal{H}_s(\boldsymbol{F}_{s-1}), \mathrm{MaxPool}_2(\boldsymbol{F}_{s-1})), \tag{9}$$

where $\mathrm{MaxPool}_2$ denotes the max pooling operator with the stride of 2. Standard batch normalization and PReLU follow Eq. (9). $\mathcal{H}_s$ is defined as

$$\mathcal{H}_s(\boldsymbol{F}) = \begin{cases} \mathcal{F}^{3\times3}(\boldsymbol{F}), & if\ s = 1; \\ \hat{\mathcal{F}}^{5\times5}(\mathcal{F}^{1\times1}(\boldsymbol{F})), & if\ s > 1. \end{cases} \tag{10}$$

For the first stage, the input is the color image that only has three channels, so we directly use a vanilla strided convolution.

*b) Decoder Network:* Given the four output features $\{\boldsymbol{F}_s : s = 1, 2, 3, 4\}$ of the encoder network, the goal of the decoder is to integrate features of different scales step by step and gradually resume the spatial resolution. For the fusion of the top features $\boldsymbol{F}_3$ and $\boldsymbol{F}_4$, we first upsample $\boldsymbol{F}_4$ by a factor of 2 to match the resolution of $\boldsymbol{F}_3$, and then a $1 \times 1$ convolution is applied to adjust the number of channels. Afterwards, the feature map is equally split into two feature maps in terms of the channel dimension, with each split refined at a different scale. Formally, we have

$$\boldsymbol{U}_4 = \mathrm{Upsample}(\boldsymbol{F}_4), \tag{11}$$

$$\boldsymbol{Q}_3 = \mathrm{Concat}(\mathcal{F}^{3\times3}(\widetilde{\boldsymbol{U}}_4^1), \mathcal{F}_2^{3\times3}(\widetilde{\boldsymbol{U}}_4^2)), \tag{12}$$

where $\widetilde{\boldsymbol{U}}_4^1$ and $\widetilde{\boldsymbol{U}}_4^2$ denote the two channel splits of $\mathcal{F}^{1\times1}(\boldsymbol{U}_4)$. Finally, features are integrated by a simple element-wise summation, *i.e.,*

$$\boldsymbol{D}_3 = \psi(\mathrm{BN}(\boldsymbol{Q}_3) + \mathrm{BN}(\mathcal{F}^{1\times1}(\boldsymbol{F}_3))), \tag{13}$$

where BN and $\psi$ are standard batch normalization [71] and nonlinear activation function [72]. $\boldsymbol{D}_3$ is passed to the bottom stages. For the feature fusion of the bottom stages, Eq. (11) is simply adapted to $\boldsymbol{U}_s = \mathrm{Upsample}(\boldsymbol{D}_s)$, and all of the remaining operations are preserved. In the end, we obtain the integrated features $\{\boldsymbol{D}_s : s = 1, 2, 3, 4\}$ at different scales ($\boldsymbol{D}_4 = \boldsymbol{F}_4$), which will be re-visited when calculating the final training loss.

*c) Deep Supervision & Loss Function:* We employ deep supervision [83] to ease the optimization of the latent units. For each fused feature $\boldsymbol{D}_s,\ s = 1, 2, 3, 4$, we project it to a single-channel feature map via a point-wise convolution. A *Sigmoid* activation function is utilized to get the saliency predictions $\boldsymbol{P}_s$. We use the binary cross entropy (BCE) loss for supervision, which is formulated as

$$\mathcal{L} = \mathrm{BCE}(\boldsymbol{P}_1, \boldsymbol{G}) + \lambda \sum_{s=2}^{4} \mathrm{BCE}(\boldsymbol{P}_s, \boldsymbol{G}), \tag{14}$$

where $\boldsymbol{G}$ refers to the ground-truth, and $\lambda$ is a hyper-parameter that is empirically set to 0.4 as in [75]. Note that $\boldsymbol{P}_1$ is the output saliency map of the proposed HVPNet.

## IV. Experiments

### A. Experimental Configurations

*a) Implementation Details:* We implement the proposed HVPNet using the popular PyTorch library. By default, we train our model using Adam optimizer with the weight decay of $10^{-4}$, and the batch size of 20. Our model and its variants are trained from scratch for 50 epochs for ablation studies. When comparing with the state-of-the-art methods, we pretrain HVPNet on the ImageNet as commonly done in state-of-the-art methods. The learning rate decays with *poly* scheduler, *i.e.,*

$$\mathrm{curr\_lr} = \mathrm{init\_lr} \times \left(1 - \frac{\mathrm{curr\_iter}}{\mathrm{max\_iter}}\right)^{\mathrm{power}}, \tag{15}$$

where $\mathrm{init\_lr} = 5 \times 10^{-4}$ and $\mathrm{power} = 0.9$ are used.

*b) Datasets:* We conduct experiments on six popular datasets, namely ECSSD [85], DUT-O (*i.e.,* DUT-OMRON) [5], DUTS [86], HKU-IS [49], SOD [87], and THUR15K [88]. These six datasets contain 1000, 5168, 15572, 4447, 300, and 6232 pairs of natural images and saliency maps, respectively. The DUTS [86] dataset is divided into 10553 training images and 5019 test images. Following recent works [7], [12], [57], [59], [64], we train models on the DUTS training set and evaluate models on the DUTS test set (DUTS-TE) as well as the other five datasets.

*c) Evaluation Metrics:* For evaluation, we adopt two widely-used metrics, *i.e.,* F-measure and mean absolute error (MAE). F-measure, denoted by $F_\beta$, is based on the precision and recall of the prediction, like

$$F_\beta = \frac{(1 + \beta^2) \times \mathrm{Precision} \times \mathrm{Recall}}{\beta^2 \times \mathrm{Precision} + \mathrm{Recall}}, \tag{16}$$

where $\beta^2$ is set to 0.3 to highlight the precision. $\mathrm{MAE}$ is a pixel-wise average of the absolute prediction error, which can be formulated as

$$\mathrm{MAE}(\boldsymbol{P}, \boldsymbol{G}) = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \left| \boldsymbol{P}_{ij} - \boldsymbol{G}_{ij} \right|, \tag{17}$$

where $\boldsymbol{P}$ is the predicted saliency map, $\boldsymbol{G}$ is the corresponding ground-truth, $H$ is the image height, and $W$ is the image width.

*d) Lightweight Measures:* The lightweight setting is the core consideration of this paper. Here, we elaborate on the lightweight measures. If a model has specified its input dimensions, we will use its default settings for testing. Otherwise, we adopt $336 \times 336$ as its input size to test its speed and compute its number of FLOPs. The CPU speed in this paper is tested on an Intel i7-8700K CPU, and the GPU speed is tested using an NVIDIA TITAN Xp GPU.

### B. Performance Comparison

*a) Comparison with Former SOD Methods:* We compare the proposed HVPNet with 15 state-of-the-art SOD methods. Table II shows the quantitative results. Our method achieves comparable performance with previous state-of-the-art BASNet [28] and EGNet [64], but significantly reduces the parameters and flops. For example, our method achieves 92.5% F-measure on ECSSD, slightly lower than the 93.8% F-measure of EGNet, but we only need 1.1% parameters of EGNet. We also achieve the fastest speed and minimal FLOPs than previous methods. Specifically, we can reach 333.2fps, while previous methods can only reach the best speed of 68fps. The number of FLOPs of HVPNet is only 1.1G. Since the number of FLOPs is related to energy consumption, the small number of FLOPs of HVPNet makes it friendly to mobile applications.

TABLE II
COMPARISON WITH EXISTING SOD METHODS. THE NUMBER OF FLOPS IS COMPUTED USING A 336 × 336 INPUT EXCEPT THAT A METHOD HAS SPECIFIED ITS OWN INPUT DIMENSIONS. WE LABEL THE BEST PERFORMANCE IN EACH COLUMN IN BOLD. HERE, THE MAIN ADVANTAGE OF OUR APPROACH LIES IN THE TRADE-OFF BETWEEN ACCURACY AND EFFICIENCY.

| Methods | #Param | FLOPs | Speed | ECSSD | | DUT-O | | DUTS-TE | | HKU-IS | | SOD | | THUR15K | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (M) | (G) | (FPS) | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ |
| DRFI [6] | - | - | 0.1 | 0.777 | 0.161 | 0.652 | 0.138 | 0.649 | 0.154 | 0.774 | 0.146 | 0.704 | 0.217 | 0.670 | 0.150 |
| DCL [56] | 66.24 | 224.9 | 1.4 | 0.895 | 0.080 | 0.733 | 0.095 | 0.785 | 0.082 | 0.892 | 0.063 | 0.831 | 0.131 | 0.747 | 0.096 |
| DHSNet [15] | 94.04 | 15.8 | 10.0 | 0.903 | 0.062 | - | - | 0.807 | 0.066 | 0.889 | 0.053 | 0.822 | 0.128 | 0.752 | 0.082 |
| RFCN [23] | 134.69 | 102.8 | 0.4 | 0.896 | 0.097 | 0.738 | 0.095 | 0.782 | 0.089 | 0.892 | 0.080 | 0.802 | 0.161 | 0.754 | 0.100 |
| NLDF [25] | 35.49 | 263.9 | 18.5 | 0.902 | 0.066 | 0.753 | 0.080 | 0.806 | 0.065 | 0.902 | 0.048 | 0.837 | 0.123 | 0.762 | 0.080 |
| DSS [62] | 62.23 | 114.6 | 7.0 | 0.915 | 0.056 | 0.774 | 0.066 | 0.827 | 0.056 | 0.913 | 0.041 | 0.842 | 0.122 | 0.770 | 0.074 |
| Amulet [14] | 33.15 | 45.3 | 9.7 | 0.913 | 0.061 | 0.743 | 0.098 | 0.778 | 0.085 | 0.897 | 0.051 | 0.795 | 0.144 | 0.755 | 0.094 |
| UCF [24] | 23.98 | 61.4 | 12.0 | 0.901 | 0.071 | 0.730 | 0.120 | 0.772 | 0.112 | 0.888 | 0.062 | 0.805 | 0.148 | 0.758 | 0.112 |
| SRM [59] | 43.74 | 20.3 | 12.3 | 0.914 | 0.056 | 0.769 | 0.069 | 0.826 | 0.059 | 0.906 | 0.046 | 0.840 | 0.126 | 0.778 | 0.077 |
| PiCANet [12] | 32.85 | 37.1 | 5.6 | 0.923 | 0.049 | 0.766 | 0.068 | 0.837 | 0.054 | 0.916 | 0.042 | 0.836 | 0.102 | 0.783 | 0.083 |
| BRN [7] | 126.35 | 24.1 | 3.6 | 0.919 | 0.043 | 0.774 | 0.062 | 0.827 | 0.050 | 0.910 | 0.036 | 0.843 | 0.103 | 0.769 | 0.076 |
| C2S [9] | 137.03 | 20.5 | 16.7 | 0.907 | 0.057 | 0.759 | 0.072 | 0.811 | 0.062 | 0.898 | 0.046 | 0.819 | 0.122 | 0.775 | 0.083 |
| RAS [61] | 20.13 | 35.6 | 20.4 | 0.916 | 0.058 | 0.785 | 0.063 | 0.831 | 0.059 | 0.913 | 0.045 | 0.847 | 0.123 | 0.772 | 0.075 |
| CPD [84] | 29.23 | 59.5 | 68.0 | 0.930 | 0.044 | 0.794 | 0.057 | 0.861 | **0.043** | 0.924 | 0.033 | 0.848 | 0.113 | 0.795 | 0.068 |
| BASNet [28] | 87.06 | 127.3 | 36.2 | **0.938** | **0.040** | **0.805** | **0.056** | 0.859 | 0.048 | **0.928** | **0.032** | 0.849 | 0.112 | 0.783 | 0.073 |
| EGNet [64] | 108.07 | 270.8 | 12.7 | **0.938** | 0.044 | 0.794 | **0.056** | **0.870** | 0.044 | **0.928** | 0.034 | **0.859** | **0.110** | **0.800** | **0.070** |
| **HVPNet (OURS)** | **1.23** | **1.1** | **333.2** | 0.925 | 0.055 | 0.799 | 0.064 | 0.839 | 0.058 | 0.915 | 0.045 | 0.826 | 0.122 | 0.787 | 0.076 |

TABLE III
COMPARISON BETWEEN THE PROPOSED HVPNET AND EXISTING LIGHTWEIGHT BACKBONE NETWORKS. WE REFORM THESE LIGHTWEIGHT BACKBONE NETWORKS FOR SOD BY VIEWING THEM AS THE ENCODER AND ADDING THE SAME DECODER AS HVPNET TO THEM. THE BEST PERFORMANCE IS HIGHLIGHTED IN **BOLD**.

| Backbone | ECSSD | | DUT-O | | DUTS-TE | | HKU-IS | | SOD | | THUR15K | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ | $F_\beta$ ↑ | MAE ↓ |
| MobileNet [33] | 0.892 | 0.127 | 0.743 | 0.091 | 0.792 | 0.090 | 0.885 | 0.098 | 0.765 | 0.208 | 0.759 | 0.090 |
| MobileNetV2 [34] | 0.903 | 0.066 | 0.760 | 0.072 | 0.804 | 0.067 | 0.896 | 0.053 | 0.807 | 0.137 | 0.768 | 0.082 |
| ShuffleNet [36] | 0.913 | 0.061 | 0.764 | 0.069 | 0.813 | 0.063 | 0.901 | 0.051 | 0.815 | 0.130 | 0.771 | 0.080 |
| ShuffleNetV2 [35] | 0.898 | 0.070 | 0.751 | 0.076 | 0.789 | 0.072 | 0.885 | 0.059 | 0.785 | 0.147 | 0.756 | 0.087 |
| **HVPNet (OURS)** | **0.925** | **0.055** | **0.799** | **0.064** | **0.839** | **0.058** | **0.915** | **0.045** | **0.826** | **0.122** | **0.787** | **0.076** |

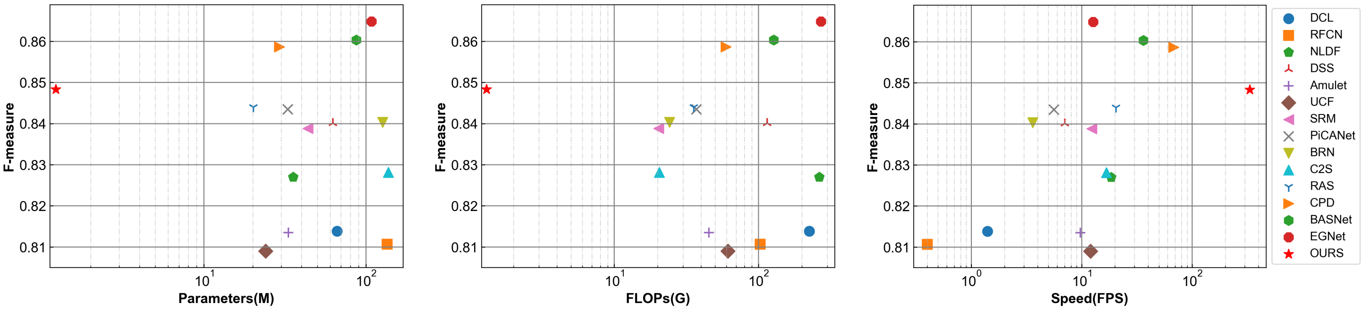

Fig. 2. Illustration of the trade-off among F-measure, the number of parameters, FLOPs, and speed. Here, the F-measure is the average of all six datasets for test. Note that the horizon axis is logarithmic.

To better illustrate the trade-off between the accuracy and efficiency, we plot three figures in Fig. 2, showing the F-measure against the number of parameters, the number of FLOPs, and speed, respectively. Here, we adopt the average F-measure over all six datasets for clarity. In the figures of F-measure *vs.*parameters and F-measure *vs.*FLOPs, HVPNet lies at the top left, which demonstrates its extremely lightweight setting and good accuracy. In the figure of F-measure *vs.*speed, HVPNet lies at the top right, which demonstrates its good trade-off between accuracy and speed. Therefore, we can come to the conclusion that HVPNet achieves a good trade-off among accuracy, the number of parameters, the number of FLOPs, and the speed.

In Fig. 3, we display some qualitative comparison with state-of-the-art SOD methods. To clearly show the difference between the predicted saliency maps of various methods, we calculate the similarity of each predicted saliency map to the corresponding ground-truth saliency map. Here, we adopt three similarity metrics, including Pearson's Correlation Coefficient (PCC or CC), Similarity (or histogram intersection, denoted as SIM), and SSIM [89]. Please refer to the survey paper [90] of saliency metrics for more details about PCC/CC and SIM, while SSIM [89] is a well-known metric for structural similarity measurement. From Fig. 3, we can observe that in spite of the extremely lightweight setting, HVPNet outperforms previous methods in strange objects, (lines 1-2), confusing scenarios (lines 3-4), thin objects (line 5), complex background (lines 6-7), indistinguishable boundaries (line 8), and large objects (lines 9-10). This further demonstrates the superiority of HVPNet.

TABLE IV
ABLATION STUDIES. "PA", "SE", "DC", "CA", "SA", "DP", AND "IP" REFER TO PARALLEL CONNECTION, SERIES CONNECTION, DENSE CONNECTION, CHANNEL-WISE ATTENTION, SPATIAL ATTENTION, DROPOUT, AND IMAGENET PRETRAINING, RESPECTIVELY.

| No. | Component | | | | | | | ECSSD | | DUT-O | | DUTS-TE | | HKU-IS | | SOD | | THUR15K | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PA | SE | DC | CA | SA | DP | IP | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ |
| 1 | ✓ | | | | | | | 0.906 | 0.065 | 0.769 | 0.073 | 0.799 | 0.071 | 0.892 | 0.056 | 0.798 | 0.138 | 0.759 | 0.087 |
| 2 | | ✓ | | | | | | 0.893 | 0.075 | 0.750 | 0.080 | 0.777 | 0.080 | 0.879 | 0.064 | 0.792 | 0.152 | 0.749 | 0.091 |
| 3 | | ✓ | ✓ | | | | | 0.909 | 0.062 | 0.772 | 0.072 | 0.807 | 0.068 | 0.899 | 0.053 | 0.798 | 0.138 | 0.766 | 0.085 |
| 4 | | ✓ | ✓ | ✓ | | | | 0.911 | 0.062 | 0.776 | 0.071 | 0.807 | 0.069 | 0.898 | 0.053 | 0.811 | 0.135 | 0.765 | 0.084 |
| 5 | | ✓ | ✓ | | ✓ | | | 0.908 | 0.062 | 0.768 | 0.073 | 0.804 | 0.069 | 0.895 | 0.054 | 0.809 | 0.136 | 0.765 | 0.085 |
| 6 | | ✓ | ✓ | ✓ | ✓ | | | 0.912 | 0.062 | 0.772 | 0.071 | 0.808 | 0.068 | 0.898 | 0.053 | 0.801 | 0.138 | 0.766 | 0.084 |
| 7 | | ✓ | ✓ | ✓ | ✓ | ✓ | | 0.910 | 0.065 | 0.781 | 0.070 | 0.814 | 0.068 | 0.900 | 0.054 | 0.809 | 0.141 | 0.769 | 0.083 |
| 8 | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 0.925 | 0.055 | 0.799 | 0.064 | 0.839 | 0.058 | 0.915 | 0.045 | 0.826 | 0.122 | 0.787 | 0.076 |

TABLE V
ABLATION STUDIES. "KERNEL ORDER" MEANS THE ORDER OF $r$ IN EQ. (1) IN EACH HVP MODULE. WE USE "#MODULES" TO DEPICT THE NUMBERS OF HVP MODULES FOR FOUR STAGES OF THE ENCODER. THE DEFAULT CONFIGURATION IS WITH A SEQUENCE OF $r$ VALUES OF $7, 5, 3, 1$ AND THE NUMBERS OF MODULES OF $1, 1, 3, 5$.

| | Config. | ECSSD | | DUT-O | | DUTS-TE | | HKU-IS | | SOD | | THUR15K | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ | $F_\beta \uparrow$ | MAE $\downarrow$ |
| Default Config. | | 0.910 | 0.065 | 0.781 | 0.070 | 0.814 | 0.068 | 0.900 | 0.054 | 0.809 | 0.141 | 0.769 | 0.083 |
| kernel order | $1, 3, 5, 7$ | 0.901 | 0.070 | 0.769 | 0.078 | 0.796 | 0.075 | 0.891 | 0.059 | 0.808 | 0.138 | 0.761 | 0.089 |
| | $1, 5, 3, 7$ | 0.900 | 0.070 | 0.768 | 0.077 | 0.790 | 0.077 | 0.889 | 0.060 | 0.795 | 0.145 | 0.762 | 0.089 |
| | $3, 7, 1, 5$ | 0.902 | 0.069 | 0.765 | 0.079 | 0.794 | 0.076 | 0.890 | 0.059 | 0.795 | 0.146 | 0.760 | 0.089 |
| | $5, 1, 7, 3$ | 0.905 | 0.068 | 0.774 | 0.076 | 0.801 | 0.073 | 0.892 | 0.057 | 0.803 | 0.137 | 0.762 | 0.088 |
| dilation rates | $9, 7, 5, 3, 1$ | 0.911 | 0.066 | 0.778 | 0.074 | 0.807 | 0.071 | 0.899 | 0.055 | 0.814 | 0.141 | 0.766 | 0.087 |
| | $5, 3, 1$ | 0.910 | 0.065 | 0.772 | 0.075 | 0.800 | 0.073 | 0.894 | 0.056 | 0.809 | 0.141 | 0.764 | 0.088 |
| #modules | $1, 1, 2, 2$ | 0.908 | 0.065 | 0.773 | 0.074 | 0.800 | 0.072 | 0.897 | 0.055 | 0.798 | 0.141 | 0.765 | 0.087 |
| | $1, 1, 3, 8$ | 0.906 | 0.064 | 0.772 | 0.076 | 0.796 | 0.073 | 0.893 | 0.057 | 0.801 | 0.141 | 0.765 | 0.089 |
| #filters | $\times 0.75$ | 0.906 | 0.067 | 0.768 | 0.077 | 0.796 | 0.076 | 0.894 | 0.058 | 0.797 | 0.143 | 0.763 | 0.089 |
| | $\times 1.25$ | 0.915 | 0.062 | 0.786 | 0.072 | 0.816 | 0.068 | 0.902 | 0.053 | 0.822 | 0.137 | 0.773 | 0.083 |

*b) Comparison with Lightweight Backbones:* Although so far there is no lightweight SOD model, there exist lightweight backbone networks designed for efficient image classification. Here, we add our lightweight decoder to four lightweight backbones for SOD, including MoblieNet [33], MobileNetV2 [34], ShuffleNet [36], and ShuffleNetV2 [35]. We adopt the same training settings as HVPNet to train these baselines. Table III demonstrates the evaluation results. We can observe that HVPNet achieves better results than directly applying lightweight backbones for SOD. This demonstrates that lightweight SOD is a worth studying and promising research field. This also proves that the proposed method is nontrivial.

## C. Ablation Study

*a) Effectiveness of Each Module Component:* Table IV verifies the effectiveness of each component of the proposed HVPNet. Our efforts start with the design of a parallel version of the HVP module. We find the densely-connected series version of the HVP module can outperform parallel connection, which verifies that processing visual conception in a hierarchical manner is more effective. Then, we incorporate the spatial and channel attention mechanisms into our encoder. The results verify that including spatial and channel attention simultaneously is beneficial to hierarchical visual perception learning. We also investigate the impact of different training strategies, *i.e.,* dropout and ImageNet [91] pretraining. We find both strategies can improve the generalization ability of our model in a majority of the experimental settings.

*b) Configurations of HVPNet:* Table V demonstrates the ablation results of various network configurations. Firstly, setting kernel sizes in the descending order for each HVP module consistently outperforms the other variants, implying the correctness of the *reverse hierarchy theory* [40]. Secondly, enhancing model capacity, *i.e.,* incorporating another pRF or increasing the number of filters, can slightly improve performance, but leads to inefficiency, which is opposite to the purpose of our design. Considering the trade-off between accuracy and efficiency, we adopt the default settings as in Table I.

## D. Evaluation for Eye Fixation Prediction

Another task that is highly related to SOD is eye fixation prediction. Unlike SOD that requires to segment the whole salient objects from an image, eye fixation prediction only aims at finding eye fixation points without the requirement for the segmentation of objects. In some studies, eye fixation prediction is also called saliency prediction. Here, we call it eye fixation prediction to distinguish it from SOD. To demonstrate the superiority of the proposed HVPNet, we also evaluate it for eye fixation prediction on the well-known SALICON 2017 benchmark [97]. SALICON 2017 contains 10000 training images and 5000 validation images with ground-truth annotations. The test set with 5000 images is released without ground-truth because it is an online competition. All images have the same resolution of 480×640. We train HVPNet for 10 epochs with a batch size of 8 and the standard loss function in [94], [95]. Other training settings are kept the same as SOD. For the evaluation metrics, we adopt four standard metrics in eye fixation prediction, including $\mathrm{NSS}$, $\mathrm{CC}$, $\mathrm{AUC}$, and $\mathrm{sAUC}$, using the code provided by the SALICON 2017 benchmark [97]. Please refer to the survey paper [90] for more details about these metrics.

TABLE VI
EVALUATION FOR EYE FIXATION PREDICTION ON THE SALICON 2017 BENCHMARK. THE NUMBER OF FLOPS IS COMPUTED USING A 480×640 INPUT IMAGE. WE HIGHLIGHT THE BEST PERFORMANCE IN EACH COLUMN IN BOLD. THE PROPOSED HVPNET ACHIEVES SIMILAR RESULTS TO THE BEST PERFORMANCE WITH AN EXTREMELY LIGHTWEIGHT SETTING.

| Method | Backbone | Year | #Param | FLOPs | VALIDATION SET | | | | TEST SET | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | (M) | (G) | NSS ↑ | CC ↑ | AUC ↑ | sAUC ↑ | NSS ↑ | CC ↑ | AUC ↑ | sAUC ↑ |
| MLNet [92] | ResNet50 | 2016 | 15.42 | 123.2 | 1.422 | 0.584 | 0.769 | 0.697 | 1.453 | 0.583 | 0.764 | 0.687 |
| SalGAN [93] | ResNet50 | 2017 | 31.78 | 94.1 | 1.635 | 0.796 | 0.846 | 0.716 | 1.662 | 0.798 | 0.847 | 0.700 |
| SAM [94] | ResNet50 | 2018 | 70.09 | 343.6 | 1.966 | 0.900 | **0.866** | 0.758 | 1.990 | 0.899 | **0.865** | **0.741** |
| EML-NET [95] | ResNet50 | 2018 | 23.54 | 25.3 | **2.002** | 0.879 | 0.861 | 0.757 | **2.018** | 0.874 | 0.858 | 0.740 |
| DINet [96] | ResNet50 | 2019 | 27.03 | 156.7 | 1.957 | **0.907** | 0.864 | **0.759** | 1.972 | **0.907** | 0.863 | **0.741** |
| **HVPNet (OURS)** | - | - | **1.23** | **3.0** | 1.981 | 0.873 | 0.865 | 0.757 | 2.003 | 0.869 | 0.863 | 0.740 |

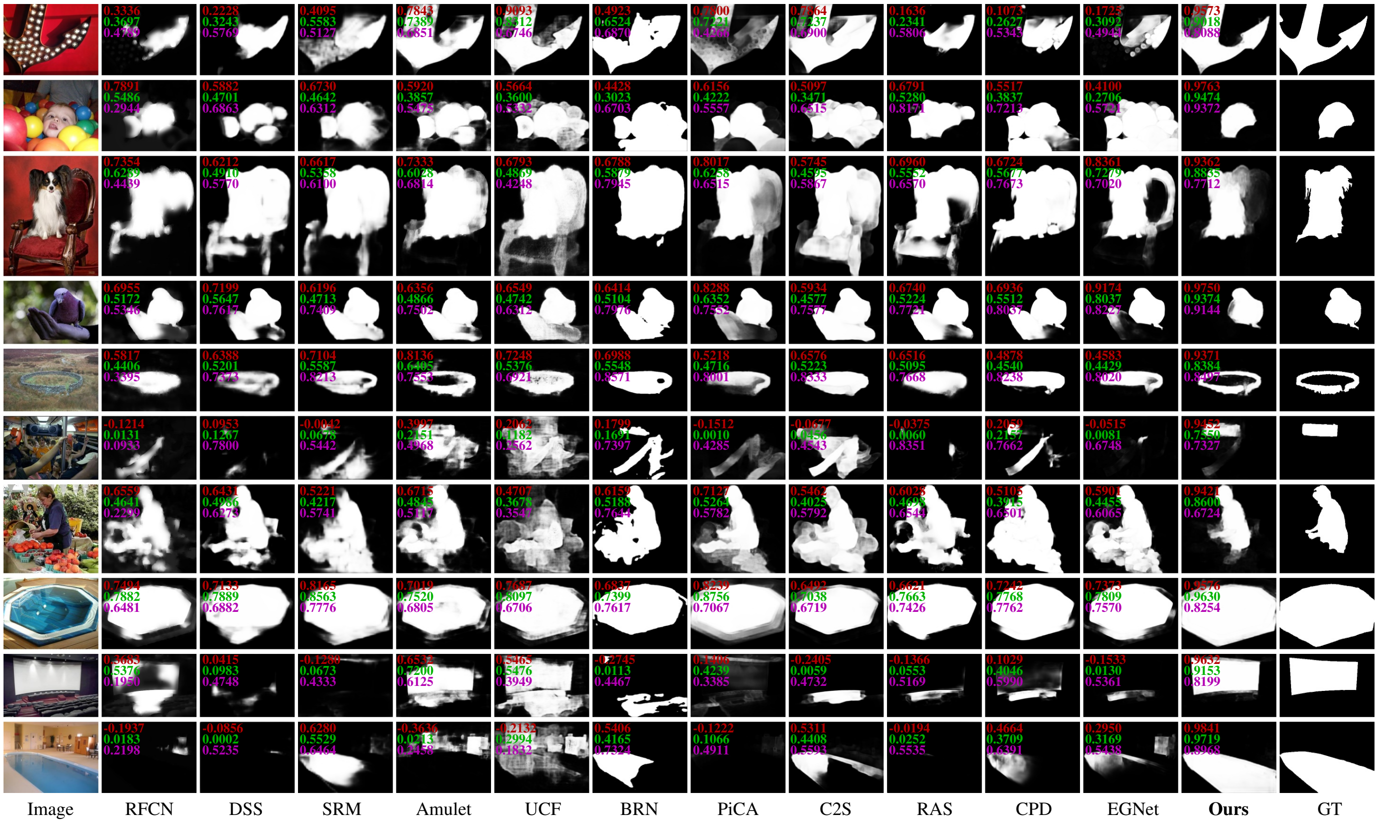

Fig. 3. Qualitative comparison with state-of-the-art SOD methods. The **red**, **green**, and **pink** numbers denote the PCC, SIM, and SSIM values between each predicted saliency map and the corresponding ground-truth (GT), respectively.

The evaluation results are summarized in Table VI. We compare the proposed HVPNet with recent state-of-the-art eye fixation prediction methods, including MLNet [92], SalGAN [93], SAM [94], EML-NET [95], and DINet [96]. We can find that HVPNet achieves similar results to the best performance in terms of all evaluation metrics but with an extremely lightweight setting, *i.e.,* significantly fewer parameters and FLOPs. Therefore, we can come to the conclusion that HVPNet achieves a good trade-off between effectiveness and efficiency for both SOD and eye fixation prediction, making it possible to be applied in practical applications.

## V. CONCLUSION

Instead of only focusing on model accuracy, in this paper, we explore a new direction for SOD, *i.e.,* lightweight SOD, which aims at achieving a good trade-off among accuracy, efficiency, the number of parameters, and the number of FLOPs. Along this path, we present a novel HVP module to imitate the primate visual cortex for hierarchical visual perception learning. Building on the proposed HVP module, the proposed HVPNet can achieve comparable accuracy with state-of-the-art SOD models while maintaining much faster speed, much fewer parameters, and FLOPs. To the best of our knowledge, this is the first attempt in SOD towards accuracy-efficiency trade-off and lightweight models. We demonstrate that directly applying lightweight backbones [33]–[36] for SOD leads to suboptimal performance, which suggests lightweight SOD is worth studying and should be set up as a new research direction. Through this study, we expect to arouse the research for lightweight SOD that has the potential to promote more practical SOD systems.

## REFERENCES

[1] U. Rutishauser, D. Walther, C. Koch, and P. Perona, "Is bottom-up attention useful for object recognition?" in *IEEE Conf. Comput. Vis.*

*Pattern Recog.*, vol. 2, 2004, pp. 37–44.

[2] W. Zhou, H. Li, and Q. Tian, "Recent advance in content-based image retrieval: A literature survey," *arXiv preprint arXiv:1706.06064*, 2017.

[3] P. Li, D. Wang, L. Wang, and H. Lu, "Deep visual tracking: Review and experimental comparison," *Pattern Recognition*, vol. 76, pp. 323–338, 2018.

[4] L. Marchesotti, C. Cifarelli, and G. Csurka, "A framework for visual saliency detection with applications to image thumbnailing," in *Int. Conf. Comput. Vis.*, 2009, pp. 2232–2239.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] W. Wang, J. Shen, X. Dong, and A. Borji, "Salient object detection driven by fixation prediction," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 1711–1720.

[12] 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.

[13] M. A. Islam, M. Kalash, and N. D. Bruce, "Revisiting salient object detection: Simultaneous detection, ranking, and subitizing of multiple salient objects," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7142–7150.

[14] 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, pp. 202–211.

[15] N. Liu and J. Han, "DHSNet: Deep hierarchical saliency network for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 678–686.

[16] S. He, J. Jiao, X. Zhang, G. Han, and R. W. Lau, "Delving into salient object subitizing and detection," in *Int. Conf. Comput. Vis.*, 2017, pp. 1059–1067.

[17] W. Wang, S. Zhao, J. Shen, S. C. Hoi, and A. Borji, "Salient object detection with pyramid attention and salient edges," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 1448–1457.

[18] 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, pp. 8150–8159.

[19] H. Li, G. Li, B. Yang, G. Chen, L. Lin, and Y. Yu, "Depthwise nonlocal module for fast salient object detection using a single thread," *IEEE Transactions on Cybernetics*, 2020.

[20] S. Chen, B. Wang, X. Tan, and X. Hu, "Embedding attention and residual network for accurate salient object detection," *IEEE Trans. Cybernetics*, vol. 50, no. 5, pp. 2050–2062, 2020.

[21] F. Guo, W. Wang, J. Shen, L. Shao, J. Yang, D. Tao, and Y. Y. Tang, "Video saliency detection using object proposals," *IEEE Transactions on Cybernetics*, vol. 48, no. 11, pp. 3159–3170, 2017.

[22] Y. Zhou, S. Huo, W. Xiang, C. Hou, and S.-Y. Kung, "Semi-supervised salient object detection using a linear feedback control system model," *IEEE Transactions on Cybernetics*, vol. 49, no. 4, pp. 1173–1185, 2018.

[23] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, "Saliency detection with recurrent fully convolutional networks," in *Eur. Conf. Comput. Vis.*, 2016, pp. 825–841.

[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] 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, pp. 6609–6617.

[26] P. Hu, B. Shuai, J. Liu, and G. Wang, "Deep level sets for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2300–2309.

[27] N. D. Bruce, C. Catton, and S. Janjic, "A deeper look at saliency: Feature contrast, semantics, and beyond," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 516–524.

[28] 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, pp. 7479–7489.

[29] M. Feng, H. Lu, and E. Ding, "Attentive feedback network for boundary-aware salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 1623–1632.

[30] S. Wang, S. Yang, M. Wang, and L. Jiao, "New contour cue-based hybrid sparse learning for salient object detection," *IEEE Transactions on Cybernetics*, 2019.

[31] K. Yan, X. Wang, J. Kim, and D. Feng, "A new aggregation of DNN sparse and dense labeling for saliency detection," *IEEE Transactions on Cybernetics*, 2020.

[32] H. Li, G. Li, and Y. Yu, "ROSA: Robust salient object detection against adversarial attacks," *IEEE Transactions on Cybernetics*, 2019.

[33] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[34] 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.

[35] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient cnn architecture design," in *Eur. Conf. Comput. Vis.*, 2018, pp. 116–131.

[36] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 6848–6856.

[37] I. González-Díaz, V. Buso, and J. Benois-Pineau, "Perceptual modeling in the problem of active object recognition in visual scenes," *Pattern Recogn.*, vol. 56, pp. 129–141, 2016.

[38] D. J. Felleman and D. E. Van, "Distributed hierarchical processing in the primate cerebral cortex," *Cerebral Cortex*, vol. 1, no. 1, pp. 1–47, 1991.

[39] T. Serre, "Hierarchical models of the visual system," *Encyclopedia of Computational Neuroscience*, pp. 1309–1318, 2015.

[40] S. Hochstein and M. Ahissar, "View from the top: Hierarchies and reverse hierarchies in the visual system," *Neuron*, vol. 36, no. 5, pp. 791–804, 2002.

[41] J. J. DiCarlo, D. Zoccolan, and N. C. Rust, "How does the brain solve visual object recognition?" *Neuron*, vol. 73, no. 3, pp. 415–434, 2012.

[42] B. A. Wandell and J. Winawer, "Computational neuroimaging and population receptive fields," *Trends in Cognitive Sciences*, vol. 19, no. 6, pp. 349–357, 2015.

[43] S. Liu, D. Huang, and Y. Wang, "Receptive field block net for accurate and fast object detection," in *Eur. Conf. Comput. Vis.*, 2018, pp. 385–400.

[44] L. Wolf, S. Bileschi, and E. Meyers, "Perception strategies in hierarchical vision systems," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2006, pp. 2153–2160.

[45] N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A. J. Rodriguez-Sanchez, and L. Wiskott, "Deep hierarchies in the primate visual cortex: What can we learn for computer vision?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1847–1871, 2012.

[46] Q. Wang, Y. Yuan, P. Yan, and X. Li, "Saliency detection by multiple-instance learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 660–672, 2013.

[47] Q. Wang, Y. Yuan, and P. Yan, "Visual saliency by selective contrast," *IEEE Trans. Circ. Syst. Video Technol.*, vol. 23, no. 7, pp. 1150–1155, 2012.

[48] G. Zhu, Q. Wang, Y. Yuan, and P. Yan, "Learning saliency by MRF and differential threshold," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2032–2043, 2013.

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

[50] Y. Liu, M.-M. Cheng, X. Zhang, G.-Y. Nie, and M. Wang, "DNA: Deeply-supervised nonlinear aggregation for salient object detection," *arXiv preprint arXiv:1903.12476*, 2019.

[51] Y. Qiu, Y. Liu, and J. Xu, "MiniSeg: An extremely minimum network for efficient COVID-19 segmentation," *arXiv preprint arXiv:2004.09750*, 2020.

[52] Y. Qiu, Y. Liu, H. Yang, and J. Xu, "A simple saliency detection approach via automatic top-down feature fusion," *Neurocomputing*, vol. 388, pp. 124–134, 2020.

[53] Y. Qiu, Y. Liu, X. Ma, L. Liu, H. Gao, and J. Xu, "Revisiting multi-level feature fusion: A simple yet effective network for salient object detection," in *IEEE Int. Conf. Image Process.*, 2019, pp. 4010–4014.

[54] 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.

[55] Y. Liu, M.-M. Cheng, D.-P. Fan, L. Zhang, J. Bian, and D. Tao, "Semantic edge detection with diverse deep supervision," *arXiv preprint arXiv:1804.02864*, 2018.

[56] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 478–487.

[57] Y. Zeng, H. Lu, L. Zhang, M. Feng, and A. Borji, "Learning to promote saliency detectors," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 1644–1653.

[58] X. Chen, A. Zheng, J. Li, and F. Lu, "Look, perceive and segment: Finding the salient objects in images via two-stream fixation-semantic CNNs," in *Int. Conf. Comput. Vis.*, 2017, pp. 1050–1058.

[59] 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.

[60] T. Zhao and X. Wu, "Pyramid feature attention network for saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 3085–3094.

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

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

[63] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 770–778.

[64] J.-X. Zhao, J. Liu, D.-P. Fan, Y. Cao, J. Yang, and M.-M. Cheng, "EGNet: Edge guidance network for salient object detection," in *Int. Conf. Comput. Vis.*, 2019, pp. 8779–8788.

[65] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 4820–4828.

[66] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-Net: Imagenet classification using binary convolutional neural networks," in *Eur. Conf. Comput. Vis.*, 2016, pp. 525–542.

[67] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Int. Conf. Comput. Vis.*, 2017, pp. 2736–2744.

[68] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 11 264–11 272.

[69] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *Science*, vol. 331, no. 6022, pp. 1279–1285, 2011.

[70] J. Benois-Pineau and P. Le Callet, *Visual Content Indexing and Retrieval with Psycho-Visual Models*. Springer, 2017.

[71] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[72] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[73] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[74] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018.

[75] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2881–2890.

[76] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 4700–4708.

[77] C. Morand, J. Benois-Pineau, J.-P. Domenger, J. Zepeda, E. Kijak, and C. Guillemot, "Scalable object-based video retrieval in HD video databases," *Signal Processing: Image Communication*, vol. 25, no. 6, pp. 450–465, 2010.

[78] Y. Yuan, Z. Xiong, and Q. Wang, "VSSA-NET: Vertical spatial sequence attention network for traffic sign detection," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3423–3434, 2019.

[79] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7132–7141.

[80] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[81] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[82] A. M. Obeso, J. Benois-Pineau, M. S. G. Vázquez, and A. A. R. Acosta, "Dropping activations in convolutional neural networks with visual attention maps," in *Int. Conf. Content-Based Multimedia Indexing*. IEEE, 2019, pp. 1–4.

[83] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *AISTATS*, 2015, pp. 562–570.

[84] 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, pp. 3907–3916.

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

[86] 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.

[87] V. Movahedi and J. H. Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2010, pp. 49–56.

[88] M.-M. Cheng, N. J. Mitra, X. Huang, and S.-M. Hu, "SalientShape: Group saliency in image collections," *The Visual Computer*, vol. 30, no. 4, pp. 443–453, 2014.

[89] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[90] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 3, pp. 740–757, 2018.

[91] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 248–255.

[92] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "A deep multi-level network for saliency prediction," in *Int. Conf. Pattern Recog.* IEEE, 2016, pp. 3488–3493.

[93] J. Pan, E. Sayrol, X. G.-i. Nieto, C. C. Ferrer, J. Torres, K. McGuinness, and N. E. OConnor, "SalGAN: Visual saliency prediction with adversarial networks," in *CVPR Scene Understanding Workshop (SUNw)*, 2017.

[94] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting human eye fixations via an LSTM-based saliency attentive model," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5142–5154, 2018.

[95] S. Jia and N. D. Bruce, "EML-NET: An expandable multi-layer network for saliency prediction," *Image and Vision Computing*, p. 103887, 2020.

[96] S. Yang, G. Lin, Q. Jiang, and W. Lin, "A dilated inception network for visual saliency prediction," *IEEE Trans. Multimedia*, vol. 22, no. 8, pp. 2163–2176, 2019.

[97] M. Jiang, S. Huang, J. Duan, and Q. Zhao, "SALICON: Saliency in context," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1072–1080.