MobileSal: Extremely Efficient RGB-D Salient Object Detection

Yu-Huan Wu, Yun Liu, Jun Xu, Jia-Wang Bian, Yu-Chao Gu, and Ming-Ming Cheng

Abstract—The high computational cost of neural networks has prevented recent successes in RGB-D salient object detection (SOD) from benefiting real-world applications. Hence, this paper introduces a novel network, MobileSal, which focuses on efficient RGB-D SOD using mobile networks for deep feature extraction. However, mobile networks are less powerful in feature representation than cumbersome networks. To this end, we observe that the depth information of color images can strengthen the feature representation related to SOD if leveraged properly. Therefore, we propose an implicit depth restoration (IDR) technique to strengthen the mobile networks’ feature representation capability for RGB-D SOD. IDR is only adopted in the training phase and is omitted during testing, so it is computationally free. Besides, we propose compact pyramid refinement (CPR) for efficient multi-level feature aggregation to derive salient objects with clear boundaries. With IDR and CPR incorporated, MobileSal performs favorably against state-of-the-art methods on six challenging RGB-D SOD datasets with much faster speed (450fps for the input size of $320 \times 320$) and fewer parameters (6.5M). The code is released at https://mmcheng.net/mobilesal.

Index Terms—RGB-D Salient Object Detection, Efficiency, Implicit Depth Restoration.

1 INTRODUCTION

Salient object detection (SOD) aims to locate and segment the most eye-catching object(s) in natural images. It is a fundamental problem in image understanding and serves as a preliminary step for many computer vision tasks such as visual tracking [1], content-aware image editing [2], and weakly supervised learning [3]. Current SOD methods are mainly developed for RGB images [4]–[6], which are usually hindered by indistinguishable foreground and background textures. To this end, researchers resort to the easily accessible depth information as an important complement to the RGB counterpart, with promising progress in RGB-D SOD [7]–[14].

While convolutional neural networks (CNNs) have made brilliant achievements on RGB-D SOD [8]–[12], [14], their high accuracy often comes at the expense of high computational costs and large model size. This situation has prevented recent state-of-the-art methods [8], [14] from being applied to real-world applications, especially for those on mobile devices, which are depth accessible, with very limited energy overhead and computational capability. Hence, it is essential to design efficient networks for accurate RGB-D SOD. A naı̈ve solution towards this goal is to adopt lightweight backbones such as MobileNets [16], [17] and ShuffleNets [18], [19] for deep feature extraction, instead of commonly-used cumbersome backbones like VGG [20] and ResNets [21]. The problem is that lightweight networks are usually less powerful than cumbersome networks on feature representation learning, as widely acknowledged by the research community [16]–[19]. This problem would hinder lightweight networks from accurate RGB-D SOD performance.

To overcome this challenge, we note that the depth information of color images, if leveraged properly, can strengthen the feature representation for RGB-D SOD [8], [9]. Unlike some existing studies [11], [12] that leverage the depth information explicitly, in this paper, we propose an implicit depth restoration (IDR) technique to strengthen the feature representation learning of the lightweight backbone network so as to ensure the accuracy of RGB-D SOD in an efficient setting. More importantly, IDR is only adopted in the training phase and is omitted during testing, so it is computationally free during the inference stage. Specifically, we enforce our model to restore the depth map from high-
level backbone features, through which the representation learning of the lightweight backbone becomes more powerful with important supervision on the depth stream. Besides the IDR module, we propose two more components to ensure the high efficiency: i) we conduct RGB and depth information fusion only at the coarsest level, because such a small feature resolution (i.e., 1/32 scale) is essential for reducing computational cost; ii) we propose a compact pyramid refinement (CPR) module to efficiently aggregate multi-scale deep features, for accurate SOD with clear boundaries.

With MobileNetV2 [17] as the backbone network, MobileSal achieves 450fps on a single NVIDIA RTX 2080Ti GPU with the input size of 320 × 320, tens of times faster than existing RGB-D SOD methods [8], [9], [14], [22]. Extensive experiments on six challenging datasets demonstrate that MobileSal also achieves competitive performance compared with state-of-the-art methods (max F-measure of 91.4% and 91.2% on NJU2K [15] and DUTLF [9] datasets, respectively) with fewer parameters (6.5M). Such high efficiency, good accuracy, and small model size would benefit many real-world applications.

In summary, our main contributions include:

- To the best of our knowledge, MobileSal is the first to shed light upon efficient RGB-D SOD by proposing an extremely efficient network with a speed of 450fps.
- To ensure the efficiency of MobileSal on cross-modal fusion, MobileSal fuses RGB and depth information only at the coarsest level and then efficiently aggregates multi-level deep features using a compact pyramid refinement (CPR) module.
- To ensure the accuracy of MobileSal, we propose an implicit depth restoration (IDR) technique to strengthen the less powerful features learned by mobile backbone networks. This technique is also applicable to other segmentation tasks such as RGB-D semantic segmentation.

2 RELATED WORK

2.1 Salient Object Detection

Benefited from the rapid development of deep CNNs [20], [21] in recent years, CNNs-based SOD methods for RGB images [5], [6], [23]–[28] have achieved substantial progress compared with conventional methods [29]–[34]. Along this direction, much attention is paid to design various effective strategies to fuse multi-scale features generated by multi-level CNN layers [6], [23], [24], [26]. Some efforts are also spent on exploring the effectiveness of extra boundary information [24], [25], [35] or part-object relationship [36]. Detailed introductions of SOD works can refer to recent popular surveys [37]–[39]. Despite many success stories, RGB SOD is hindered by indistinguishable foreground and background textures, which can be largely alleviated by incorporating the depth information, i.e., RGB-D SOD.

2.2 RGB-D Salient Object Detection

Like early SOD methods, conventional RGB-D SOD works extract hand-crafted features from RGB and depth maps and fuse them together [40]–[45]. Recently, RGB-D SOD has gained more attention and deep-learning-based RGB-D SOD has been developed rapidly [8]–[13], [46]–[50]. Typically, Zhao et al. [8] proposed a contrast-prior-based network, providing strong depth enhancement. Piao et al. [9] proposed to refine depth via depth-induced multi-scale recurrent attention (DMRA). Huang et al. [49] proposed RGB-D fusion via joint cross-modal and unimodal features, providing more comprehensive RGB-D analysis. Zhang et al. [11] provided another perspective of uncertainty RGB-D saliency via conditional variational autoencoders. Chen et al. [51] first introduced 3D CNNs to RGB-D SOD, providing more abundant spatial semantics. Ji et al. [52] introduced a flexible depth calibration module, providing reliable complementary information for saliency models. Zhao et al. [47] proposed a self-supervised learning framework, which only leverages image-level annotations saving large costs from large-scale data annotations. More inspiring related works can refer to the recent survey [53].

In terms of the fusion strategy of RGB and depth information, the diagrams of RGB-D SOD architectures can be broadly divided into late fusion [42], [54], [55], early fusion [11]–[13], [51], and multi-scale fusion [8], [9], [14], [22], [46]–[49]. Late fusion appears at the end of feature extraction and only predicts the result from the fused features [42], [54], [55]. Early fusion directly concatenates the input RGB image and depth map and then derives the saliency map from such RGB-D input using the encoder-decoder network [12], hypercolumn network [11], siamese network [14], or 3D CNN [51]. Multi-scale fusion first extracts RGB and depth features separately and then aggregates RGB-D features at all levels [9], [49], at middle and high levels [56], or at middle levels [46]. Although the early fusion strategy is more efficient, multi-scale fusion is more accurate. To ensure high efficiency, our method only fuses RGB and depth features at the coarsest level in a small resolution. IDR is then applied to strengthen the feature representation learning of mobile networks in a computationally free manner.

2.3 Efficient Backbone Networks

Recent growing interests in mobile vision applications have generated a high demand for efficient CNNs. Mobile devices such as autonomous driving vehicles, robots, and smartphones only have limited computational resources, so traditional cumbersome networks, like VGG [20] and ResNets [21], are unsuitable for these platforms. To this end, some efficient networks are proposed for image classification, such as MobileNets [17], ShuffleNets [19], MnasNet [57], etc. There also emerge some efficient networks for semantic segmentation [58], object detection [59], and ordinary RGB SOD [60], [61]. These efficient networks are with low computational costs and thus flexible for mobile platforms. In this paper, we are the first to shed light upon efficient RGB-D SOD by adopting MobileNetV2 [17] as the backbone for deep feature extraction. Our proposed techniques aim to ensure the SOD accuracy and high efficiency simultaneously in such a lightweight setting.

3 METHODOLOGY

In this section, we first provide an overview of our method in §3.1. Then, we introduce the proposed cross-modal feature fusion scheme in §3.2, implicit depth restoration in §3.3,
compact pyramid refinement in §3.4. Finally, we present the hybrid loss function in §3.5.

3.1 Overview

Fig. 2 depicts the overall architecture of our method. We use RGB and depth streams for separate feature extraction. RGB Stream. We employ MobileNetV2 [17] as the backbone of our method. To adapt it to the SOD task, we remove the global average pooling layer and the last fully-connected layer from the backbone. For the RGB stream, each stage is followed by a convolutional layer with a stride of 2, and thus feature maps are downsampled into half resolution after each stage. For convenience, we denote the output feature maps for five stages as $C_1, C_2, C_3, C_4, C_5$, with strides of $2, 2^2, 2^3, 2^4, 2^5$, respectively.

Depth Stream. Similar to the RGB stream, the depth stream also has five stages with the same strides. Since depth maps contain less semantic information than the corresponding RGB images, we build a lightweight depth network with fewer convolutional blocks than the RGB stream. Each stage only has two Inverted Residual Blocks (IRB) [17]. Such a design reduces computational complexity that accords with the goal of efficient RGB-D SOD. In each IRB, we first expand the feature map along the channel dimension by $M$ times via a $1 \times 1$ convolution, followed by a depthwise separable $3 \times 3$ convolution [16] with the same number of input and output channels. Then, the feature channels are squeezed to $1/M$ via another $1 \times 1$ convolution. Here, each convolution is followed by Batch Normalization (BN) [52] and ReLU [63] layers, except for the last $1 \times 1$ convolution that only has a BN layer. The final output of the inverted residual block is the element-wise sum of the initial input and the output generated by the above three sequential convolutions. For the first layer in each stage, the stride of the depthwise separable convolution is set as 2, and the number of hidden feature channels is increased if needed. The output feature maps of five stages of the depth stream are denoted as $D_1, D_2, D_3, D_4, D_5$, the first four of which have 16, 32, 64, 96 channels, respectively. $D_5$ and $C_5$ have the same number of channels and the same stride.

As shown in Fig. 2, with the outputs of the RGB and depth stream, we first fuse the extracted RGB feature $C_5$ and depth feature $D_5$ to generate the RGB-D feature $C_D$. The proposed IDR technique restores the depth map from $C_1, C_2, C_3, C_4, C_D$, which is supervised by the input depth map to strengthen the feature representation learning. For saliency prediction, we design a lightweight decoder using the CPR module as the basic unit. The output of the decoder at the bottom stage is the final predicted saliency map. More details can be seen in the following sections.

3.2 Cross-Modal Fusion of RGB and Depth Features

The depth map reveals spatial cues of color images, which helps distinguish the foreground objects from the background, especially for scenarios with complicated textures. As demonstrated by previous studies [8], [9], [14], [22], [46], [48], proper RGB and depth feature fusion is essential for accurate RGB-D SOD. Our main consideration here is to ensure the high efficiency of our method. Instead of conducting fusion at multiple levels [8], [9], [14], [22], [46], [48], [64], we only fuse RGB and depth features at the coarsest level because the small feature resolution leads to low computational cost.

According to the above analyses, we only fuse the RGB feature map $C_5$ and the depth feature map $D_5$. We design a lightweight Cross-Modal Fusion (CMF) module for this purpose, as shown in Fig. 2. Intuitively, semantic information mainly exists in the RGB image. The depth map conveys the
prior of depth-smooth regions that approximately represent the shapes and structures of complete objects or stuff. Hence, we adopt depth features like a gate to enhance RGB semantic features through multiplication, which can be viewed as a strong regularization. Note that element-wise addition or concatenation can only aggregate two feature maps by treating features equally, which is orthogonal to our goal. Experiments in §4.3 also demonstrate our hypothesis.

Specifically, we first combine the RGB and depth features with an above-mentioned IRB to derive the transited RGB-D feature maps $\mathcal{T}$, which can be formulated as

$$\mathcal{T} = \text{IRB}(C_5 \otimes D_5),$$

where and $\otimes$ is the element-wise multiplication operator. Meanwhile, we apply a global average pooling (GAP) layer to $C_5$ to get a feature vector, followed by two fully-connected layers to compute the RGB attention vector $\mathbf{v}$, like

$$\mathbf{v} = \sigma(FC_2(\text{ReLU}(FC_1(\text{GAP}(C_5))))),$$

in which FC and ReLU denote fully-connected and ReLU layers, respectively. The number of output channels of $FC_1$ and $FC_2$ is the same as the input. $\sigma$ indicates the standard sigmoid function. With $\mathcal{T}$ and $\mathbf{v}$ computed, the multiplication of $\mathbf{v}$, $\mathcal{T}$, and $D_5$ are fed into an IRB, like

$$C_5^D = \text{IRB}(\mathbf{v} \otimes \mathcal{T} \otimes D_5),$$

where $C_5^D$ indicates the output feature map of the CMF module. Note that $\mathbf{v}$ is replicated to the same shape as $\mathcal{T}$ before multiplication. Eq. (3) filters RGB semantic features again by multiplying $D_5$, and the channel attention $\mathbf{v}$ is used to recalibrate the fused features. After the fusion of RGB and depth features, we can derive the backbone features, including the RGB features $C_1, C_2, C_3, C_4$, and the fused RGB-D feature $C_5^D$.

### 3.3 Implicit Depth Restoration

As widely acknowledged [16]–[19], lightweight backbone networks are less powerful in feature representation learning than cumbersome networks. To ensure the accuracy of RGB-D SOD, we consider strengthening the representation learning of mobile networks. We observe that the depth map conveys depth-smooth regions that usually represent objects, object parts, or smooth background, because intuitively, an object or a connected stuff region usually has similar depth. This observation motivates us to use the depth map as an extra supervision source to guide the representation learning, which would help mobile networks restrain the texture changes within objects or connected stuff regions and highlight the difference among them. In this way, the contrast between salient objects and the background will be strengthened too. With this idea, we design an Implicit Depth Restoration (IDR) technique. Here, we use the word “implicit” because IDR is only adopted in the training phase and is omitted during testing, making it computationally free for practical deployment.

We continue by introducing how to use $C_1, C_2, C_3, C_4, C_5^D$ for the above auxiliary supervision. As shown in Fig. 3 (a), the pipeline of IDR is simple, i.e., just concatenating multi-level feature maps and then fusing them. Specifically, we first apply a $1 \times 1$ convolution to squeeze $C_1, C_2, C_3, C_4, C_5^D$ to the same number of channels, i.e., 256. Then, the resulting feature maps are resized to the same size as $C_4$, followed by the concatenation of them. A $1 \times 1$ convolution changes the concatenated feature map from 1280 channels to 256 channels for saving computational cost. Next, four sequential IRBs are followed to fuse multi-level features so that we can obtain powerful multi-scale features. At last, a simple $1 \times 1$ convolution converts the fused feature map to a single channel. With a standard sigmoid function and bilinear upsampling, we can obtain the restored depth map with the same size as the input. The training loss of IDR adopts the well-known SSIM metric [65] to measure the structural similarity between the restored depth map $D_r$ and input one $D_g$, which can be written as

$$\mathcal{L}_{IDR} = 1 - \text{SSIM}(D_r, D_g),$$

where SSIM uses the default setting. Note that the above operations are omitted during testing to make IDR free.

### 3.4 Compact Pyramid Refinement

It is widely accepted that high-level features in the backbone network contain semantic abstract features, while low-level features convey fine-grained details. For accurate SOD, it is essential to fully utilize both high-level and low-level features. There exists a lot of literature on this topic [8], [9], [13], [14], [22], but existing methods usually design cumbersome decoders without consideration of efficiency. Here, our decoder should not only fuse multi-level features effectively but also be efficient as much as possible.

The proposed decoder uses the Compact Pyramid Refinement (CPR) module as the basic unit. For efficiency, CPR
uses $1 \times 1$ and depthwise separable convolutions [16] instead of vanilla convolutions in previous methods [12–14], [22]. Since multi-level features exhibit multi-scale representations with the high level corresponding to the coarse scale and vice versa, multi-scale learning would be necessary for multi-level feature fusion. Hence, CPR adopts a lightweight multi-scale learning strategy to enhance such fusion. Suppose that the input of a CPR module is $\mathcal{X}$. As shown in Fig. 3 (b), CPR first applies a $1 \times 1$ convolution to expand the number of channels by $M$ times. Then, three $3 \times 3$ depthwise separable convolutions with dilation rates of $1, 2, 3$ are connected parallel for multi-scale fusion. This can be formulated as

$$\begin{align*}
\mathcal{X}_1 &= \text{Conv}_{1\times 1}(\mathcal{X}), \\
\mathcal{X}^{d_1}_2 &= \text{Conv}_{3\times 3}^{d_1}(\mathcal{X}_1), \\
\mathcal{X}^{d_2}_2 &= \text{Conv}_{3\times 3}^{d_2}(\mathcal{X}_1), \\
\mathcal{X}^{d_3}_2 &= \text{Conv}_{3\times 3}^{d_3}(\mathcal{X}_1), \\
\mathcal{X}_2 &= \text{ReLU}(\text{BN}(\mathcal{X}^{d_1}_2 + \mathcal{X}^{d_2}_2 + \mathcal{X}^{d_3}_2)),
\end{align*}$$

(5)

where $d_1, d_2$, and $d_3$ are dilation rates, i.e., $1, 2, 3$ here, respectively. BN is the abbreviation of batch normalization [62]. A $1 \times 1$ convolution is used to squeeze channels to the same number as the input, i.e.,

$$\mathcal{X}_3 = \text{Conv}_{1\times 1}(\mathcal{X}_2) + \mathcal{X},$$

(6)

which uses a residual connection for better optimization. The attention mechanism in Eq. (2) is applied to $\mathcal{X}$ to calculate an attention vector $\mathbf{v'}$, so that we have

$$\mathbf{y} = \mathbf{v'} \odot \text{Conv}_{1\times 1}(\mathcal{X}_3).$$

(7)

Eq. (7) uses global contextual information to recalibrate the fused features.

As shown in Fig. 2, at each decoder stage, two feature maps from the top decoder and the corresponding encoder stage first reduce their numbers of channels to half using a $1 \times 1$ convolution separately. The results are then concatenated, followed by a CPR module for feature fusion. In this way, our lightweight decoder aggregates multi-level features from top to bottom.

3.5 Hybrid Loss Function

At each decoder stage, we predict the saliency map by sequentially adding a $1 \times 1$ convolution with a single channel, a sigmoid function, and bilinear upsampling to the output of the CPR module, as shown in Fig. 2. Hence, we can derive predicted saliency maps $P_i (i = 1, 2, \cdots, 5)$ for five stages, respectively. Let the ground-truth saliency map be $G$. The loss of each side-output can be computed as

$$L^i_{sal} = \text{BCE}(P_i, G) + \text{Dice}(P_i, G),$$

(8)

BCE denotes binary cross-entropy loss function:

$$\text{BCE}(P_i, G) = G \cdot \log P_i + (1 - G) \cdot \log (1 - P_i),$$

(9)

where “$\cdot$” indicates the dot-product operation. Dice represents the Dice loss [66]:

$$\text{Dice}(P_i, G) = 1 - \frac{2 \cdot G \cdot P_i}{||G|| + ||P_i||},$$

(10)

where $|| \cdot ||$ denotes the $\ell_1$ norm. With deep supervision and IDR, the training loss can be formulated as

$$L = \sum_{i=1}^{5} L^i_{sal} + \lambda \cdot L_{IDR},$$

(11)

where $\lambda$ is a balance weight. In the testing phase, $P_1$ is the final predicted saliency map.

4 Experiments

We first provide the experimental setup in §4.1. Then, we compare with state-of-the-art RGB-D SOD methods in §4.2 and conduct comprehensive ablation studies in §4.3. We also discuss the applications of IDR in §4.4.

4.1 Experimental Setup

Implementation details. We implement our network in PyTorch [70] and Jittor [71]. If not specified, we use MobileNetV2 [17] as our backbone. The $M$ values in the depth stream, CPR, and IDR are set to 4, 4, and 6, respectively. We resize both RGB and depth images into $320 \times 320$. We use horizontal flipping and random cropping as the default data augmentation for the ablation study. After freezing the designs and parameters, we apply multi-scale training, i.e., each image is resized into [256, 288, 320] in training, but we keep the size of test images unchanged. We use a single RTX 2080Ti GPU for training and testing. The initial learning rate $lr$ is 0.0001, and the batch size is 10. We train our network for 60 epochs. The poly learning rate policy is applied, so that the actual learning rate for each epoch $cur_{epoch}$ is $(1 - \frac{cur_{epoch}}{60})^{power} \times lr$, where $power$ is 0.9. The Adam optimizer [72] is used for optimizing our networks, and the momentum, weight decay, $\beta_1$, and $\beta_2$ are set as 0.9, 0.0001, 0.9, and 0.99, respectively.

Datasets. We conduct experiments on seven widely-used datasets, including NJU2K [15], DUTLF-D [9], NLPR [44], STERE [73], SSD [74], and SIP [10]. They contain 1985, 1200,
Table 1
Quantitative results on six challenging datasets. The best, second, and third best results are highlighted in red, blue and bold, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>DESM</th>
<th>LHM</th>
<th>ACSD</th>
<th>DCMC</th>
<th>CTMF</th>
<th>PCF</th>
<th>TANet</th>
<th>CFFP</th>
<th>DMRA</th>
<th>D3Net</th>
<th>JLDCF</th>
<th>S2MA</th>
<th>UCNet</th>
<th>DANet</th>
<th>BiAnet</th>
<th>MobileSal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed (FPS)</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>13</td>
<td>74</td>
<td>65</td>
<td>43.2</td>
<td>43.2</td>
<td>33.3</td>
<td>27.9</td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
</tr>
<tr>
<td>Fβmax</td>
<td>0.76</td>
<td>0.70</td>
<td>0.70</td>
<td>0.79</td>
<td>0.87</td>
<td>0.88</td>
<td>0.85</td>
<td>0.91</td>
<td>0.91</td>
<td>0.89</td>
<td>0.86</td>
<td>0.85</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
</tr>
<tr>
<td>MAE</td>
<td>0.286</td>
<td>0.204</td>
<td>0.20</td>
<td>0.171</td>
<td>0.085</td>
<td>0.059</td>
<td>0.060</td>
<td>0.053</td>
<td>0.051</td>
<td>0.047</td>
<td>0.041</td>
<td>0.054</td>
<td>0.043</td>
<td>0.047</td>
<td>0.044</td>
<td>0.041</td>
</tr>
<tr>
<td>RMSE</td>
<td>0.671</td>
<td>0.515</td>
<td>0.708</td>
<td>0.686</td>
<td>0.849</td>
<td>0.877</td>
<td>0.878</td>
<td>0.876</td>
<td>0.866</td>
<td>0.900</td>
<td>0.902</td>
<td>0.894</td>
<td>0.897</td>
<td>0.904</td>
<td>0.905</td>
<td></td>
</tr>
<tr>
<td>PSNR</td>
<td>0.807</td>
<td>0.738</td>
<td>0.814</td>
<td>0.805</td>
<td>0.913</td>
<td>0.924</td>
<td>0.923</td>
<td>0.923</td>
<td>0.923</td>
<td>0.944</td>
<td>0.940</td>
<td>0.930</td>
<td>0.936</td>
<td>0.936</td>
<td>0.941</td>
<td>0.942</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.915</td>
<td>0.892</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
<td>0.902</td>
</tr>
<tr>
<td>No.</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rank</td>
<td>15</td>
<td>15</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2
CPU inference time of different methods. Default input size of each method is applied to test CPU inference time.

<table>
<thead>
<tr>
<th>Method</th>
<th>Ours</th>
<th>JLDCF</th>
<th>UCNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Size</td>
<td>320×320</td>
<td>320×320</td>
<td>352×352</td>
</tr>
<tr>
<td>Inference Time (ms)</td>
<td>43 (1×)</td>
<td>7246 (150×)</td>
<td>784 (18×)</td>
</tr>
<tr>
<td>Method</td>
<td>D3Net</td>
<td>S2MA</td>
<td>DMRA</td>
</tr>
<tr>
<td>Input Size</td>
<td>224×224</td>
<td>256×256</td>
<td>256×256</td>
</tr>
<tr>
<td>Inference Time (ms)</td>
<td>677 (15×)</td>
<td>3049 (70×)</td>
<td>2381 (55×)</td>
</tr>
</tbody>
</table>

Table 3
Quantitative comparisons of restored depth maps by IDR with different scales of input depth maps. The ground-truth depth (GT) recovered from different resolutions using different interpolation methods are compared.

<table>
<thead>
<tr>
<th>Setting</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>1/16</td>
<td>1/8</td>
<td>1/32</td>
<td>1/8</td>
</tr>
<tr>
<td>Data Type</td>
<td>IDR</td>
<td>GT</td>
<td>Bilinear</td>
<td>GT</td>
</tr>
<tr>
<td>Nearest</td>
<td>Bilinear</td>
<td>Bilinear</td>
<td>Bilinear</td>
<td></td>
</tr>
<tr>
<td>PSNR</td>
<td>22.86</td>
<td>30.17</td>
<td>22.35</td>
<td>24.27</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.8687</td>
<td>0.9194</td>
<td>0.8445</td>
<td>0.8170</td>
</tr>
</tbody>
</table>

Table 4
Ablation study for the RGB-D fusion and IDR branch. Note that the variants of No. 6 and 12 (in bold) fuse RGB and depth features only at the coarsest feature level.

<table>
<thead>
<tr>
<th>No.</th>
<th>Features to be fused</th>
<th>IDR</th>
<th>Fβmax</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>0.899</td>
<td>0.052</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>0.894</td>
<td>0.050</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>0.897</td>
<td>0.047</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>0.902</td>
<td>0.048</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>0.906</td>
<td>0.046</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>0.906</td>
<td>0.045</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>0.895</td>
<td>0.047</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>0.892</td>
<td>0.049</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td>0.896</td>
<td>0.048</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td>0.895</td>
<td>0.048</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td>0.898</td>
<td>0.048</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td>0.896</td>
<td>0.047</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td>0.887</td>
<td>0.052</td>
</tr>
</tbody>
</table>

100, 1000, 80, 927 images, respectively. Following [8], [11], [12], [22], we use 1500 images of NJU2K [15] and 700 images of NLPR [44] for training, and the other 485 images of NJU2K [15] and 300 images of NLPR [44] for testing. Except for DUTLF-D [9], other datasets are directly used for testing. On the DUTLF-D [9] dataset, we follow [9], [12] to use 800 images for training and the other 400 images for testing.

Evaluation metrics. Following recent works [7], [8], [11], [68], we adopt two widely-used metrics for evaluation. The first is F-measure Fβ, where β is set as 0.3 to emphasize the significance of the precision, as suggested by [8], [9], [12], [75]. We compute the maximum Fβ as Fβmax under different thresholds. Higher Fβmax indicates better performance. The second is the Mean Absolute Error (MAE), which is the lower, the better. We also report scores of recently proposed S-measure Sβ [76] and maximum E-measure Eβmax [77] under different thresholds for reference. We follow the official paper to compute Sβ and Eβmax. We calculate the overall rank of each method on each dataset based on the above four metrics. Besides, we report the number of parameters and running time of each method for efficiency analysis.
We first compare our method with 15 recent state-of-the-art methods. While others may fail in such scenarios, our method runs at 450fps and only has 6.5M parameters.

Table 1 shows the results. The IDR branch substantially improves the performance in most cases (No. 3-6), and only MAE (fusing low-level features) degrades (No. 1, 2). This validates the efficacy of the combination of our proposed fusion strategy and the IDR branch. We also compare the used fusion strategy with the early fusion strategy, which concatenates the input RGB image and depth map at the input stage. Although the latter strategy is more efficient, our initial fusion strategy significantly outperforms it (Table 5). Hence, To ensure accuracy and efficiency, we fuse RGB and depth features at the coarsest level.

**4.3 Ablation Study**

We evaluate each proposed component on the test split of the NJU2K [15] dataset. $F^\max_\beta$ and MAE are applied as the primary metrics. The results are analyzed below.

**Different RGB-D fusions.** Table 4 shows the results for the RGB-D fusion at different stages. When trained with the IDR branch, fusing RGB and depth features at the coarsest level results in the best performance (No. 6). Fusing RGB and depth features in the last three levels results in the best performance when trained without the IDR branch (No. 11). The IDR branch substantially improves the performance in most cases (No. 3-6), and only MAE (fusing low-level features) degrades (No. 1, 2). This validates the efficacy of the combination of our proposed fusion strategy and the IDR branch. We also compare the used fusion strategy with the early fusion strategy, which concatenates the input RGB image and depth map at the input stage. Although the latter strategy is more efficient, our initial fusion strategy significantly outperforms it (Table 5). Hence, To ensure accuracy and efficiency, we fuse RGB and depth features at the coarsest level.

**Saved time for RGB-D fusion.** In MobileSal, we conduct RGB and depth information fusion only at the coarsest level. Conducting RGB-D feature fusion of all levels will result in 260fps for our method. Hence, we will save 42% $(1 - \frac{260}{450})$ time. In other words, fusing only the coarsest will speed up our method by 73% $(\frac{450}{260} - 1)$.

**Depth restoration quality.** We explore the restored depth quality of IDR with the widely-used PSNR and SSIM [65] metrics. The scale of the restored depth map in IDR is 1/16 of the input depth map. For comparison, we evaluate the quality of the nearest and bilinear interpolation of the input depth map of 1/8 scale. As the IDR branch receives depth features of the 1/32 scale, we also report the quality of the bilinear interpolation of depth GT with the 1/32 scale. Results are shown in Table 3. We observe that the restored depth map is closer to the input depth map than the upsampled 1/32 GT. We also conduct visual comparisons of the above settings in Fig. 5. One can see that the restored depth maps from IDR keep good restoration quality and are with less noise than the upsampled 1/8 GT.

**Loss selection in IDR.** As introduced in §3.3, instead of using the trivial $L_1/L_2$ loss, the SSIM metric [65] is chosen as the loss of IDR. We validate this design via training our MobileSal with $L_1/L_2/SSIM$ loss. We find that IDR with the $L_1/L_2$ loss can enhance MobileSal by 0.5%/0.7% in terms of the $F^\max_\beta$, and is 0.5%/0.3% lower than IDR with the SSIM loss. This is because SSIM loss provides the structural similarity rather than the simple point-to-point error computed by the $L_1/L_2$ loss. Based on the above discussion, we choose the SSIM loss as the supervision of IDR.

**The $\lambda$ coefficient in the loss function.** $\lambda$ decides the loss weight of the depth restoration loss, as described in Eq. (11). We conduct experiments on our method with different $\lambda$ settings. The results are shown in Table 6. The IDR branch
brings substantial improvement to the robustness of our method with different $\lambda$. Since the third column achieves the best performance, we adopt $\lambda = 0.3$ as the default setting for training our network.

**Depth information and the CMF module.** The results in Table 7 demonstrate the effects of depth information and the CMF module. Results with depth input are trained with the IDR branch. The provision of depth maps without the CMF module only uses an element-wise multiplication for RGB-D fusion. The results show that the depth information is very helpful for RGB-D SOD even with a very simple operation. More specifically, we also observe substantial improvement with the CMF module.

**Operation for initial RGB-D fusion in the CMF module.** As formulated in Eq. (1), an element-wise multiplication is applied to the initial RGB-D fusion. To validate the effectiveness of the selected operation, we compare this design with the widely-used element-wise addition. The results are presented in Table 8. Element-wise multiplication largely outperforms element-wise addition and concatenation by 0.9% and 0.6% in terms of the maximum F-measure, suggesting the superiority of element-wise multiplication in the initial RGB-D fusion of the CMF module.

**Final fusion strategy of the CMF module.** As described in Eq. (3), both the RGB attention vector $v$ and RGB-D feature $T$ join the fusion with depth feature $D_5$. We validate the effectiveness of the above design via removing $v$ or $T$. Experimental results are shown in Table 9. We observe a large performance degradation after removing $T$. This is because in the feature fusion of Eq. (3), only $T$ contains spatial-wise RGB features. Besides, $v$ is an attention vector obtained by RGB features, and $D_5$ is the pure depth feature. Removing $v$ also affects the performance a bit because it provides the channel-wise RGB attention in the RGB-D fusion.

**Compact pyramid refinement.** Table 10 shows the results for CPR, where different dilation strategies are used. We test the default setting (No. 1), single convolution with different dilation rates (No. 2 - 4), and convolutions with sparse combinations of dilation rates (No. 5, 6). The default setting with compact dilation rates (1, 2, 3) significantly outperforms other settings, demonstrating the efficacy of CPR.

**Experimental setup.** We select two recent state-of-the-art representatives [78], [79] as the baselines. We use the official code provided by the authors to implement our ideas. Following [78], [79], we conduct our experiments on the NYUDv2 dataset [80], which consists of 1449 RGB images with corresponding depth maps and pixel-level semantic labels containing 40 semantic classes. This dataset has 795 and 654 images for training and testing, respectively. The training and testing settings follow the official papers [78], [79]. Similar to MobileSal, for [79], RGB features re-calibrated by the depth features for each stage are fed into the IDR branch. However, for [78], different from MobileSal, the output features of the first four stages are the input of the IDR branch because the features of the last stage in [78] are directly used for the prediction of semantic segmentation.

**Evaluation metrics.** Following [78], [79], mean IoU (mIoU), is used as the primary evaluation metric. We also

**Table 6**

<table>
<thead>
<tr>
<th>No.</th>
<th>RGB</th>
<th>CPR</th>
<th>Depth</th>
<th>CMF</th>
<th>$F_{max}$</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.896</td>
<td>0.047</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.902</td>
<td>0.046</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.906</td>
<td>0.045</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.902</td>
<td>0.044</td>
</tr>
</tbody>
</table>

**Table 7**

<table>
<thead>
<tr>
<th>No.</th>
<th>RGB</th>
<th>CPR</th>
<th>Depth</th>
<th>CMF</th>
<th>$F_{max}$</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.896</td>
<td>0.047</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.902</td>
<td>0.046</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.906</td>
<td>0.045</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.902</td>
<td>0.044</td>
</tr>
</tbody>
</table>

**Table 8**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Multiplication</th>
<th>Addition</th>
<th>Concatenation</th>
<th>$F_{max}$</th>
<th>MAE</th>
<th>$F_{max}$</th>
<th>MAE</th>
<th>$F_{max}$</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>0.906</td>
<td>0.045</td>
<td>0.897</td>
<td>0.048</td>
<td>0.900</td>
<td>0.046</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 9**

<table>
<thead>
<tr>
<th>No.</th>
<th>Depth</th>
<th>CMF</th>
<th>$v$</th>
<th>$T$</th>
<th>$F_{max}$</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.894</td>
<td>0.048</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.895</td>
<td>0.049</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.902</td>
<td>0.046</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.906</td>
<td>0.045</td>
</tr>
</tbody>
</table>

**Table 10**

<table>
<thead>
<tr>
<th>No.</th>
<th>D Rates</th>
<th>Depth</th>
<th>CPR</th>
<th>Depth</th>
<th>CMF</th>
<th>$F_{max}$</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1, 2, 3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>1, 3, 6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Analysis of the fusion of CMF in Eq. (3).** No. 1: trivial RGB-D fusion via element-wise multiplication; No. 2-3: variants of removing $v$ or $T$ in Eq. (3); No. 4: the default design.
With the computationally-free IDR incorporated, we observe this project is supported by the National Key Research and Development Program of China (Grant No. 2018AAA0100400) and NSFC (NO. 61922046).

### 5 Conclusion

We propose a new method, MobileSal, which aims at efficient RGB-D SOD. Unlike other accurate RGB-D SOD methods, we are the first to shed light upon efficient RGB-D SOD by proposing an extremely efficient network MobileSal with a speed of 450fps. With less powerful features provided by the mobile backbone network, we propose the implicit depth restoration (IDR) technique to strengthen the less powerful features learned by mobile backbone networks. We perform ablation study for the proposed techniques in MobileSal and demonstrate their effectiveness. We conduct experimental comparisons with state-of-the-art methods on six popular benchmarks. The results show that MobileSal performs favorably against state-of-the-art methods, with fewer parameters and much faster speed. In terms of CPU inference time, our method is 15–150× faster. Our method can serve as a strong baseline for future efficient RGB-D SOD research.

### 6 Acknowledgment

This project is supported by the National Key Research and Development Program of China (Grant No. 2018AAA0100400) and NSFC (NO. 61922046).

### References


Yu-Huan Wu is currently a Ph.D. candidate with College of Computer Science at Nankai University, supervised by Prof. Ming-Ming Cheng. He received his bachelor’s degree from Xidian University in 2018. His research interests include computer vision and machine learning.

Yun Liu received his bachelor’s and doctoral degrees from Nankai University in 2016 and 2020, respectively. Currently, he works as a postdoctoral scholar with Prof. Luc Van Gool at ETH Zurich. His research interests include computer vision and machine learning.

Jun Xu received his B.Sc. and M.Sc. degrees from School of Mathematics Science, Nankai University, Tianjin, China, in 2011 and 2014, respectively, and the Ph.D. degree from the Department of Computing, Hong Kong Polytechnic University, in 2018. He worked as a Research Scientist at IAI, Abu Dhabi, UAE. He is currently a Lecturer with School of Statistics and Data Science, Nankai University. More information can be found at https://csjunxu.github.io/.

Jia-Wang Bian Jia-Wang Bian received the B.Eng. degree from Nankai University, where he was advised by Prof. M.-M. Cheng. He is currently pursuing the Ph.D. degree with The University of Adelaide. He was a Research Assistant with the Singapore University of Technology and Design (SUTD). He also did a Trainee Engineer Job with the Advanced Digital Sciences Center (ADSC), Huawei Technologies Co., Ltd., and Tusimple. He is also an Associated Ph.D. Researcher with the Australian Centre for Robotic Vision (ACRV). He is advised by Prof. I. Reid and Prof. C. Shen. His research interests include computer vision and robotics.

Yu-Chao Gu received his bachelor’s degree from the Beijing University of Chemical Technology in 2019. He is currently pursuing the master’s degree with the College of Computer Science, Nankai University. His research interests include efficient deep-learning and computer vision.

Ming-Ming Cheng received his PhD degree from Tsinghua University in 2012. Then he did two years research fellow with Prof. Philip Torr in Oxford. He is now a professor at Nankai University, leading the Media Computing Lab. His research interests include computer graphics, computer vision, and image processing. He received research awards, including ACM China Rising Star Award, IBM Global SUR Award, and CCF-Intel Young Faculty Researcher Program. He is on the editorial boards of IEEE TPAMI and IEEE TIP.