A Highly Efficient Model to Study the Semantics of Salient Object Detection

Ming-Ming Cheng, Shang-Hua Gao, Ali Borji, Yong-Qiang Tan, Zheng Lin, Meng Wang

Abstract—CNN-based salient object detection (SOD) methods achieve impressive performance. However, the way semantic information is encoded in them and whether they are category-agnostic is less explored. One major obstacle in studying these questions is the fact that SOD models are built on top of the ImageNet pre-trained backbones which may cause information leakage and feature redundancy. To remedy this, here we first propose an extremely light-weight holistic model tied to the SOD task that can be freed from classification backbones and trained from scratch, and then employ it to study the semantics of SOD models. With the holistic network and representation redundancy reduction by a novel dynamic weight decay scheme, our model has only 100K parameters, \( \sim 0.2\% \) of parameters of large models, and performs on par with SOTA on popular SOD benchmarks. Using CSNet, we find that a) SOD and classification methods use different mechanisms, b) SOD models are category insensitive, c) ImageNet pre-training is not necessary for SOD training, and d) SOD models require far fewer parameters than the classification models. The source code is publicly available at https://mmcheng.net/sod100k/.

Index Terms—salient object detection, efficient saliency prediction, semantics.

1 INTRODUCTION

Based on the observed human reaction time and signal transmission time along biological pathway [11], [82], cognitive psychology studies suggest that human visual system (HVS) recruits pre-attentive, bottom-up attention mechanisms before recognizing the semantics of the scene [72]. Computer vision community has modeled these findings by using category-independent hand-crafted contrast features [2], [9] to build traditional salient object detection (SOD) models [3]. Many computer vision applications such as image retrieval [5], [8], [23], visual tracking [29], photographic composition [7], [21], image quality assessment [80], content-aware image processing [59], [92], and unsupervised semantic segmentation [18], utilize SOD models based on the hypothesis that salient objects are generic and category-independent.

While SOD methods based on convolutional neural networks (CNNs) have made significant progress, most of them focus on improving the state-of-the-art (SOTA) performance by learning fine local details and global features [75], [90], [95], [96], attention cues [4] as well as edge cues [14], [79], [98]. Existing CNN-based SOD models rely on ImageNet pre-trained backbone architectures [15], [25] with a considerable amount of parameters to extract features. However, ImageNet pre-training inevitably introduces category semantics into SOD models, causing a potential conflict with the conventional assumption that salient regions are category-independent [3], [37], [87]. This potential conflict raises new questions. How much of a role do category semantics play in the bottom-up SOD task? Is ImageNet pre-training inevitable and necessary for SOD training?

There are two principles in designing a SOD model that can be used to answer these questions. First and foremost, a SOD model should be possible to train without relying on ImageNet pre-training. Existing SOD models are built upon classification backbones that contain too many parameters, making them difficult to be trained from scratch. In order to distinguish between thousand of different classification categories, even the light-weight classification backbone models ResNet-18 [25] and MobileNet v2 [34] contains 11M and 4.2M parameters (vs 100K parameters of our entire model). An important motivation behind this work is to verify if those category-oriented features and the corresponding parameters are indispensable for the SOD task. Second, the SOD task requires the model to have high feature resolution and strong multi-scale ability, that are non-essential for classification backbone [77], [95]. Existing works [14], [30], [78] relieve these problems by adding the SOD task related saliency heads to backbones, but inevitably introduce extra parameters.

To achieve the aforementioned requirements, we propose an extremely light-weight model that holistically consider the feature extractor and saliency head. We generalize the OctConv [6], namely gOctConv, with more flexibility and additional the self-adaptive property. A dynamic weight decay scheme is designed to enables self-adaptive learnable number of feature channels in gOctConv. This scheme not only helps analyze the semantic information in SOD models, but also allows \( \sim 80\% \) parameter reduction with a negligible performance drop. Utilizing gOctConv, we propose a highly light-weight holistic Cross-Stage Cross-Scale network, namely CSNet. Benefiting from the holistically design and the dynamic weight decay, CSNet performs on par with SOTA but has only 100k parameters (\( \sim 0.2\% \) parameters of SOTA models).

As a bonus to the extremely low number of parameters, our CSNet can be directly trained from scratch without ImageNet pre-training, providing a unique opportunity for answering questions
Informative features from image patches for improving the quality of salient object detection methods [43], [50], [76] utilize CNNs to extract more hand-crafted features to detect salient objects. Early deep learning methods [9], [37], [73], [87], [102] mainly rely on traditional methods that are category insensitive and the detected salient objects are generic. Further, we observe that SOD models require far fewer parameters than classification models since the feature representation for distinguishing categories is not needed in SOD models. The category insensitive mechanism of our SOD model not only provides an opportunity to improve the efficiency of SOD models, but also mimic the category independent bottom-up human attention mechanism.

In summary, we make two major contributions in this paper:

- We thoroughly analyze the semantic information in CNN-based SOD models and experimentally verify that SOD models require negligible category information (i.e. category insensitive).
- By holistically redesigning feature extraction and SOD prediction, we abandon the wildly used pre-trained CNN backbones, which contains intensive parameters for unnecessary category information. Accompanying the sparsity introduced by dynamic weight decay, we significantly slim the model to \(\sim 0.2\%\) parameters of SOTA with comparable performance.

Our focus in this work is on semantics of SOD whereas the conference version [19] was concerned mostly with building a light-weight SOD model. In Sec. 3, we introduce the dynamic weight decay scheme and the learnable channels for generalized OctConv. We then introduce the light-weight holistic CSNet designed for the analysis of SOD model in Sec. 4. With the CSNet, we analyze the semantics of the SOD model in Sec. 5. In Sec. 6, we do performance evaluation and ablations to show the efficiency of CSNet.

## 2 Related Works

### 2.1 Salient Object Detection

Traditional methods [9], [37], [73], [87], [102] mainly rely on hand-crafted features to detect salient objects. Early deep learning based methods [43], [50], [76] utilize CNNs to extract more informative features from image patches for improving the quality of saliency maps. Inspired by the fully convolutional networks (FCNs) [55], recent methods [13], [41], [51], [78], [93], [95] formulate salient object detection as a pixel-level prediction task and solve it in an end-to-end manner using FCN based models. These methods [30], [66], [77], [95], [96] capture both fine details and global features from different stages of the backbone network. Edge cues are introduced in [45], [57], [79], [98] to refine the saliency detection from the perspective of network optimization. Recently, Wei et al. [81] decomposes the original saliency map into a detail map to better learn edge features, and a body map to avoid the distraction from pixels near edges. Pan et al. [64] integrate the features from adjacent levels to solve the multi-scale issue in the SOD task. Multi-level gate units are used in [100] to balance the contribution from each encoder block and to suppress the activation of the features from non-salient regions. Despite the impressive performance, all of these CNN-based methods require powerful pre-trained ImageNet backbones as the feature extractor, which usually results in high computational cost. Moreover, none of them have studied the semantics behind the CNN-based SOTA SOD models and whether pre-training is indeed necessary.

### 2.2 Light-weight Models

Currently, most light-weight models that are initially designed for classification tasks utilize modules such as inverted block [33], [34], channel shuffling [58], [97], and SE attention module [33], [71] to improve network efficiency. Classification tasks [67] predict high-level semantic labels for an image, requiring more global information but fewer details. Thus, light-weight models [33], [34], [58], [97] designed for classification use aggressive downsampling strategies at earlier stages to save multiply-accumulate operations (MACC). These strategies makes them less useful for feature extraction since SOD task requires multi-scale information at both coarse and fine levels. Also, the SOD task focuses on determining the salient region while classification tasks predict category information. To improve saliency prediction performance under limited computing budget, the allocation of computational resources (i.e. resolution, channels) should be reconsidered.

### 2.3 Network Pruning

Many network pruning methods have been proposed to prune less important filters especially on the channel level [28], [44]. To prune filters, redundant filters can be estimated by norm criterion [26], [44], statistical information of the next layer [56], geometric median of weights [27], and reusing the scaling factor of the batch normalization layer [53]. Meta pruning et al. [54] utilizes generated weights to estimate the performance of the remaining filters. Most pruning approaches still rely on regularization tricks such as weight decay to introduce sparsity in filters. Our proposed dynamic weight decay stably introduces sparsity for assisting pruning algorithms in removing redundant filters, resulting in learnable channels for each scale in our proposed gOctConv.

## 3 gOctConv & Learnable Channels

To analyze the semantics of CNN-based SOD models without disturbed by the ImageNet pre-training, we need to build a light-weight SOD model that 1) can be trained from scratch without reliance on ImageNet pre-training; 2) is simple enough to avoid
improves the vanilla OctConv for the SOD task in the following stage conv features with self-adaptive learnable channels as shown in Fig. 3. While originally designed to be a replacement for the traditional convolution unit, the OctConv [6] takes two high/low-resolution inputs from the same stage with a fixed number of feature channels. Our gOctConv allows an arbitrary number of input resolutions from both within-stage and cross-stage conv features with a learnable number of channels.

3.1 Generalized OctConv

Originally designed to be a replacement for the traditional convolution unit, the vanilla OctConv [6] shown in Fig. 3 conducts the convolution operation across low and high scales within a layer. However, only two scales within a stage are not enough to excavate multi-scale information required for the SOD task (see also Tab. 6). The number of channels for each scale in the vanilla OctConv is manually set, which requires a lot of effort to re-adjust for a saliency model. Therefore, we propose a generalized OctConv (gOctConv) that allows incorporating an arbitrary number of input scales from both within-stage and cross-stage conv features with self-adaptive learnable channels as shown in Fig. 3.

As a generalized version of the vanilla OctConv, gOctConv improves the vanilla OctConv for the SOD task in the following aspects. Firstly, instead of being a module with a fixed structure, the flexible gOctConv is a class that allows many instances under different SOD design requirements. For instance, the cross-scales feature interaction can be turned off to support large complexity flexibility. Arbitrary numbers of input and output scales are available to support a larger range of multi-scale representations. Except for within-stage features, the gOctConv can also process cross-stage features with arbitrary scales from the feature extractor. With the high flexible instances of gOctConv, we can design a highly efficient but very simple SOD model. Secondly, the gOctConv supports self-adaptive learnable channels for each scale. This property allows analyzing some properties of a SOD model such as model complexity and multi-scale feature requirements. The implementation details and complexity analysis of gOctConv is provided in the supplementary.

3.2 Learnable Channels

We propose to construct self-adaptive learnable channels for each scale in the gOctConv, by utilizing our proposed dynamic weight decay to assist channel pruning. Dynamic weight decay maintains a stable output feature distribution among channels while introducing sparsity, helping pruning algorithms to eliminate redundant channels at the expense of a negligible performance drop.

Introducing Sparsity with Dynamic Weight Decay: The commonly used regularization trick weight decay [39], [91] endows CNNs with better generalization performance. Mehta et al. [60] show that weight decay introduces sparsity into CNNs, which helps prune unimportant weights. Training with weight decay makes unimportant weights in CNN have values close to zero. Thus, weight decay has been widely used in pruning algorithms to introduce sparsity [27], [28], [44], [53], [54], [56]. The common implementation of weight decay is by adding the L2 regularization to the loss function, which can be written as follows:

$$L = L_0 + \lambda \sum \frac{1}{2} w_i^2,$$

where $L_0$ is the loss for the specific task, $w_i$ is the weight of the i-th layer, and $\lambda$ is the weight for weight decay. During back propagation, the weight $w_i$ is updated as:

$$w_i \leftarrow w_i - \nabla f_i(w_i) - \lambda w_i,$$

where $\nabla f_i(w_i)$ is the gradient to be updated, and $\lambda w_i$ is the decay term, which is only associated with the weight itself. Applying a large decay term enhances sparsity, and meanwhile inevitably enlarges the diversity of weights among channels. Fig. 4 shows that diverse weights cause the unstable distribution of outputs among channels. Ruan et al. [12] reveal that channels with
diverse outputs are more likely to contain noise, leading to biased representation for subsequent filters. Attention mechanisms have been widely used to re-calibrate the diverse outputs with extra blocks and computational cost [12], [35]. We propose to relieve diverse outputs among channels with no extra cost during the inference. We argue that the diverse outputs are mainly caused by the indiscriminate suppression of decay terms to weights. Therefore, we propose to adjust the weight decay based on specific features of certain channels. Specifically, during back propagation, decay terms are dynamically changed according to features of certain channels. The weight update of the proposed dynamic weight decay is written as:

$$w_i \leftarrow w_i - \nabla f_i (w_i) - \lambda_d M(x_i) w_i,$$

where $\lambda_d$ is the weight of dynamic weight decay, $x_i$ denotes the features calculated by $w_i$, and $M(x_i)$ is the feature metric, which can have multiple definitions depending on the task. In this paper, our goal is to stabilize the weight distribution among channels according to features. Thus, we simply use the global average pooling (GAP) [48] as the metric for a certain channel:

$$M(x_i) = \frac{1}{HW} \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} x_i(h, w),$$

where $H$ and $W$ are the height and width of the feature map $x_i$. The dynamic weight decay with the GAP metric ensures that the weights producing large value features are suppressed, giving a compact and stable weight and output distribution as revealed in Fig. 4 and Fig. 5. The metric can also be defined as other forms to suit certain tasks, as we will study in our future work.

**Self-adaptive Learnable channels:** Now, we incorporate dynamic weight decay with pruning algorithms to remove redundant weights, to construct the self-adaptive learnable channels at each scale in gOctConvs. We follow [53] to use the weight of the BatchNorm layer as the indicator of the channel importance. The BatchNorm operation [36] is written as follows:

$$γ \leftarrow \frac{x - E(x)}{\sqrt{\text{Var}(x) + \epsilon}} \gamma + \beta,$$

where $x$ and $y$ are input and output features, $E(x)$ and $\text{Var}(x)$ are the mean and variance, respectively, and $\epsilon$ is a small factor in avoiding zero variance. $\gamma$ and $\beta$ are learned factors. We apply the dynamic weight decay to $γ$ during training. The output features after the BatchNorm layer and the activation layer are used as the input to compute the metric in Eqn. (4). Fig. 5 reveals a clear gap between important and redundant weights, and unimportant weights are suppressed to nearly zero ($w_i < 1e-20$). Thus, we can easily remove channels whose $γ$ is less than a small threshold. The learnable channels of each resolution features in gOctConv are obtained. The algorithm of constructing learnable channels of gOctConvs is illustrated in Alg. 1.

### Algorithm 1 Learning Channels for gOctConv with Dynamic Weight Decay

**Require:** The initial CSNet in which channels for all scales in gOctConvs are set. Input images $X$ and corresponding label $Y$.

1. for each iteration $i \in [1, \text{MaxIteration}]$ do
2. Feed input $X$ to the network to get the result $\hat{Y}$
3. Compute Loss = criterion($\hat{Y}$, $Y$)
4. Compute metric for each channel using Eqn. (4)
5. Backward with dynamic weight decay using Eqn. (3).
6. end for
7. Eliminate redundant channels to get the learnable channels for each scale in gOctConv.
8. Train for several iterations to finetune remaining weights.

## 4 Holistic Light-Weight Model for Studying the Semantics of SOD Model

### 4.1 Overview

While several CNN-based SOD models [30], [45], [57], [66], [77], [79], [83], [95], [95], [96], [98], [99] with impressive performance and efficiency have been proposed. However, the SOD community usually builds models on top of the ImageNet pre-trained backbones, which limits the design space and inherits a large number of parameters containing category-oriented representations. Even the light-weight classification backbone itself, e.g., ResNet-18 and MobileNet v2, contain 11M and 4.2M parameters respectively. Thanks to the extremely low number of parameters (100K) and the holistic design, our CSNet can be directly trained from scratch without ImageNet pre-training. Such design frees the CSNet from unnecessary category-oriented information contained in ImageNet pre-trained models [62], [88].

To support the analysis of each component in the model, we use different instances of the proposed gOctConv to construct
a simple yet effective SOD model. We holistically design the feature extractor and a cross-stage fusion part following the requirements of SOD task as shown in Fig. 2. It simultaneously processes features within multiple scales. The feature extractor is stacked with our proposed in-layer multi-scale block, namely ILBlocks. The cross-stage fusion part processes features from stages of the feature extractor to obtain a high-resolution output. The ILBlock and cross-stage fusion part, both composed of instances of gOctConv, enhance the within-stage and cross-stage multi-scale representation ability required by the SOD task. The straightforward structure of the CSNet avoids the potential influence of complex modules. Also, benefiting from the self-adaptive property of gOctConv, we can better study the complexity and feature requirements of the SOD model. Therefore, CSNet is a suitable tool for studying the semantics of SOD models.

4.2 In-layer Multi-scale Block

ILBlock enhances the multi-scale representation of features within a stage. gOctConvs are utilized to introduce multi-scale capacity within ILBlock. Borrowing the common definition from the classification models [33], [34], [68], [97], a highly light-weight SOD model should be at least 10× smaller than existing SOD models. The vanilla OctConv requires about 60% MACC [6] to achieve similar performance as in the standard convolution, which is not enough to design a highly light-weight model. To save computational cost, integrating features with different scales in every layer is unnecessary. Therefore, we apply an instance of the gOctConv that eliminates the cross-scale operations while keeps within-scale operations, namely simplified gOctConv. In simplified gOctConv, each input channel corresponds to an output channel with the same resolution, and a depthwise operation within each scale is utilized to further save computational cost. The simplified instance of gOctConv only requires about 1/channel MACC compared to the vanilla OctConv. ILBlock is composed of a vanilla OctConv and two 3 × 3 simplified gOctConvs as shown in Fig. 2. The vanilla OctConv integrates features with two scales and simplified gOctConvs extract features within each scale. Multi-scale features within a block are separately processed and interacted alternately. Each gOctConv is followed by the BatchNorm [36] and PReLU [24].

4.3 Cross-stage Fusion

Common SOD methods retain a high output resolution by preserving high feature resolution at the high-level of the feature extractor, which inevitably increases the computational redundancy. Another solution is to construct complex multi-level aggregation modules to fuse high-level features with semantics and low-level features with details. The value of multi-level aggregation is widely recognized on many tasks, e.g., edge detection [85], object detection [22], classification [70], and the SOD task [30]. However, these works utilize large backbone models. How to efficiently and concisely achieve cross-stage fusion for the SOD task remains challenging. In this works, we aims at designing a simple yet effective holistic model that strongly tied to the SOD task. We also need a simple yet effective multi-level aggregation strategy to analyze the semantics of SOD models. To this end, we simply use the cross-stage instance of gOctConv to fuse multi-scale features from stages of the feature extractor and generate the high-resolution output. A cross-stage gOctConv 1 × 1 takes features with different scales from the last conv of each stage as input and conducts a cross-stage convolution to output features with different scales. To extract multi-scale features at a granular level, each scale of features is processed by a group of parallel convolutions with different dilation rates. Features are then sent to another cross-stage gOctConv 1 × 1 to generate features with the highest resolution. Another standard Conv 1 × 1 layer outputs the prediction result of the saliency map.

4.4 Implementation details of CSNet

CSNet consists of a feature extractor and a cross-stage fusion part. As shown in Tab. 2, the feature extractor is stacked with ILBlocks, and is split into 4 stages according to the resolution of feature maps, where each stage has 3, 4, 6, and 4 ILBlocks, respectively. Initially, we set the number of channels of the first stage to 20, and double the channels of ILBlocks as the resolution decreases, except for the last two stages that have the same number of channels. The channel for each gOctConv can be expanded to enlarge the model capability. Models with channels expanded k times are denoted by CSNet-×k. Learnable channels of OctConvs are then obtained with the self-adaptive channel learning scheme. Given an input image with the shape \((H, W)\), the first gOctConv in the first ILBlock takes in the image and outputs features with two resolutions \((H, W)\) and \((H/2, W/2)\). The features with two scales are processed in parallel by the feature extractor. We denote the term “split-ratio” as the ratio of the number of channels among different feature scales in gOctConv. The split-ratio in ILBlocks can be adjusted to construct models with different MACC, denoted by \(C_H/C_L\). Unless otherwise stated, the channels for different scales in ILBlocks are set evenly. For cross-stage fusion, only the output feature of each stage is used. The last ILBlock of each stage merges two streams of different scales to the high-resolution stream. The cross-stage fusion part processes features from stages of the feature extractor to obtain a high-resolution output. As a trade-off between efficiency and performance, features from the last three stages are used. Learnable channels of gOctConvs in this part are also obtained. The detailed configurations of the cross-stage fusion part are shown in Tab. 1.

5 Analysis of SOD Models

In this section, we are going to answer these questions about the semantics of the CNN-based SOD model with our proposed CSNet: 1) Are SOD models sensitive to category information? 2) Which part of the SOD model is most responsible for locating the salient regions? 3) Can the SOD model detect salient regions over unseen categories? 4) Do SOD models have the same complexity compared to classification models? 5) What are the feature requirements of SOD task from the feature extractor? 6) What role does ImageNet pre-training play in the SOD model training?

5.1 Category Sensitivity

Before the emergence of CNNs, SOD methods were regarded as being category agnostic [3], [37], [87]. Researchers have used these category agnostic saliency detection models to support downstream tasks [5], [21], [23], [31], [59], [80], [92], for tasks such as weakly supervised segmentation. With the widespread popularity of CNNs, the community has proposed several new SOD models utilizing powerful ImageNet pre-trained CNN backbones to extract features. Are these models still category insensitive and can they be used as generic features? How much
role category information plays in CNN based saliency models? Can we use saliency as general knowledge to reduce the domain-specific data annotation in tasks like weakly supervised semantic segmentation? These are very important questions but have been less explored. To study the category sensitivity of the SOD model, we analyze the CSNet trained with the same data but over the SOD and classification tasks.

### 5.1.1 Data preparation

Data distribution plays a vital role in representation learning. To remove the influence of different data distribution from different datasets, we train the SOD model and the classification model over the same set of images but use salient object mask labels and category labels as supervision for each task. Since the image-level category label is more accessible than pixel-level annotations for SOD, we annotate the existing SOD dataset with category labels. Specifically, we utilize the commonly used datasets DUTS-TR [74], DUTS-TE [74] and ECSSD [86] as the source dataset, and assign category labels of the ImageNet to images. These SOD datasets have imbalanced category distribution. Thus, they can not be directly used for the analysis of category sensitivity. Analyzing models under the classification metric and SOD metric requires different data distributions. We therefore choose two sub-datasets for the evaluation of classification and SOD tasks, respectively.

**Data for classification:** The DUTS-TR dataset is used as the training dataset. As the DUTS-TE dataset has a highly similar category distribution with the DUTS-TR dataset, we use the DUTS-TE dataset to evaluate the classification task. A classifier trained on imbalanced category data may overfit to some categories, making the classification metric meaningless. To harness this, we choose 10 categories that have a balanced number of images in both DUTS-TR and DUTS-TE datasets. The training set and testing set contain 644 and 150 images, respectively. The distribution of the classification dataset is illustrated in Fig. 6.

**Data for SOD:** Eliminating a certain small category among many categories will have almost no impact on the original SOD dataset. To reinforce the impact of the category on the SOD dataset, we organize images into major categories according to the WordTree of ImageNet [67], and select 12 merged categories with a considerable number of images. Most SOD experimental results presented in this paper are reported on the ECSSD dataset [86]. Thus, we use the ECSSD dataset for the evaluation of SOD tasks. The distribution of images from the selected categories in the SOD dataset is illustrated in Fig. 7. We show in Fig. 8 that even with a biased category distribution, SOD models can still achieve reasonable performance.

### TABLE 1

Architecture for the cross-stage fusion part using four stages in CSNet×1.

<table>
<thead>
<tr>
<th>stage</th>
<th>output feature size</th>
<th>config [op, kernel size, stride]</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>[224×224×10, 112×112×10]</td>
<td>[OctConv 3×3, 1, gOctConv 3×3, 1, gOctConv 3×3, 1] ×1</td>
</tr>
<tr>
<td>stage2</td>
<td>[112×112×20, 56×56×20]</td>
<td>[OctConv 3×3, 2, gOctConv 3×3, 1, gOctConv 3×3, 1] ×1</td>
</tr>
<tr>
<td>stage3</td>
<td>[56×56×40, 28×28×40]</td>
<td>[OctConv 3×3, 2, gOctConv 3×3, 1, gOctConv 3×3, 1] ×1</td>
</tr>
<tr>
<td>stage4</td>
<td>[28×28×40, 14×14×40]</td>
<td>[OctConv 3×3, 2, gOctConv 3×3, 1, gOctConv 3×3, 1] ×1</td>
</tr>
</tbody>
</table>

### TABLE 2

Architecture for the feature extractor in CSNet×1.

<table>
<thead>
<tr>
<th>stage</th>
<th>output feature size</th>
<th>config [op, kernel size, stride]</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage1</td>
<td>[224×224×20, 112×112×40, 56×56×80, 28×28×80]</td>
<td>gOctConv, kernel size 1×1, dilation 1</td>
</tr>
<tr>
<td>stage2</td>
<td>[224×224×(1+1+1+1), 112×112×(2+2+2+2+5), 56×56×(5+5+5+5+6), 28×28×(5+5+5+5+6)]</td>
<td>DilatedConvs, kernel size 3×3, dilations [1, 2, 4, 8, 16] × scales</td>
</tr>
<tr>
<td>stage3</td>
<td>224×224×70</td>
<td>gOctConv, kernel size 1×1, dilation 1</td>
</tr>
<tr>
<td>stage4</td>
<td>224×224×1</td>
<td>StandardConv, kernel size 1×1, dilation 1</td>
</tr>
</tbody>
</table>
The basic idea of transfer learning is to pre-train the model on the source task, and finetune the pre-trained model on the target task to acquire performance gain or ease the convergence [25].

Transfer learning from ImageNet pretraining to many downstream tasks, e.g., depth estimation, crowd counting, and bounding box regression, has been proved very effective. And transfer learning between the source and target tasks under similar semantic requirements, e.g., both tasks are category-related, is more effective than the transfer learning of two tasks under different semantics. Classification task is a category sensitive task, and the classification accuracy can be used to measure the category sensitivity of models. We have verified this hypothesis by conducting experiments of transferring the semantic segmentation model to the classification model.

CSNet is specifically designed for the SOD task, we modify the cross-stage fusion part to make it suitable for classification. The cross-stage fusion part takes in features from different stages and outputs features with the lowest resolution, as low-resolution features are more suitable for classification [25]. The last Conv 1×1 layer in CSNet is replaced with a global average pooling layer and a fully connected (FC) layer. By doing so, except for the last prediction layer, the remaining parts of CSNet trained for SOD and classification can share parameters. We use the top-1 classification accuracy as the metric for analyzing the category sensitivity of models. We conduct two groups of experiments as follows: (1) Cls-scratch is denoted as the model trained with only category labels from scratch for the classification task. (2) Finetune-SOD is denoted as the model that is pre-trained with SOD annotations on the SOD task and then finetuned on the classification task.

If Finetune-SOD model achieves significantly worse results than the Cls-scratch model, we can conclude that the SOD model is not sensitive to the category information. During finetuning, parts of the Finetune-SOD model are finetuned for the classification task and other parts are fixed with SOD pre-trained weights. By doing so, we can find out which parts of the SOD model are more related to the SOD task and which parts are more general to both classification and SOD tasks. For example, if finetuning stage 1 of the SOD model achieves very limited performance gain than only finetuning the FC layer, it indicates that features in stage 1 are more general instead of very closely related to the SOD task. Stages 1 to 4 and the cross-stage fusion parts are all studied as shown in Tab. 3.

With the transfer learning from SOD to classification, we can answer questions: 1) Whether SOD model is sensitive to category information. 2) The parts of SOD model that are more task-specific and parts that are more general to both classification and SOD tasks.

Experimental results: Tab. 3 shows the top-1 classification accuracy of models transferred from the SOD task to the classification task. The classification model trained from scratch achieves the top-1 acc. of 61.1%, while the SOD model with only the FC layer finetuned achieves the top-1 acc. of 18.1%. The result means that the model trained for SOD requires almost no category information to determine the salient region. To pinpoint which part of the SOD model is more task-specific or general to classification and SOD task, we finetune a part of the SOD model to check the relative classification performance gain. Finetuning the cross-stage fusion part achieves a considerable performance gain of 30.2%, indicating that the feature difference of this part between classification and SOD task is much larger than the other parts. Finetuning from stage 1 to stage 4 achieves increasingly higher performance gain, showing that high-level features are more task-specific while low-level features are general for both tasks.

To give a more direct evidence of the category sensitivity of SOD models, we give the class activation maps [101] (CAM) comparison between the SOD model finetuned on all stages (model with 58.4 acc. on Tab. 3) and the FC layer (model with 18.1 acc. on Tab. 3) for the classification task as shown in Tab. 9. CAM of the model finetuning on all stages focus on objects information. To pinpoint which part of the SOD model are finetuned for the classification task, we denoted as the model that is pre-trained with SOD annotations on the SOD task and then finetuned on the classification task.

Settings: The basic idea of transfer learning is to pre-train the model on the source task, and finetune the pre-trained model on the target task to acquire performance gain or ease the convergence [25].

Transfer learning between musical instrument models. We conduct two groups of experiments as follows: (1) Cls-scratch is denoted as the model trained with only category labels from scratch for the classification task. (2) Finetune-SOD is denoted as the model that is pre-trained with SOD annotations on the SOD task and then finetuned on the classification task.
TABLE 3
The top-1 acc. of classification task using models transferred from the SOD task. s1 to s4 fuse refer to stage1 to stage4 and the fusion part as shown in Fig. 2, respectively. ✓ indicates that parameters in a stage are finetuned.

<table>
<thead>
<tr>
<th>Setups</th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
<th>s4</th>
<th>fuse</th>
<th>top1 acc.</th>
<th>gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Finetune-SOD ✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>18.1</td>
<td>-</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>21.5</td>
<td>3.4</td>
<td>30.9</td>
<td>12.8</td>
<td>32.9</td>
<td>14.8</td>
<td>36.2</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>48.3</td>
<td>30.2</td>
<td>58.4</td>
<td>40.3</td>
<td>61.1</td>
<td>43.0</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 9. Class activation maps comparison between the SOD model finetuned on all stages (ALL) and the FC layer (FC) for the classification task. CAM of the model finetuning on all stages focus on objects required by the classification, while CAM of the model finetuning on only the FC layer has no specific category-related focus.

5.1.3 SOD over Unseen Categories

Settings: We now study the category sensitivity of SOD models from the perspective of SOD metric. Generalizing to unseen objects or categories is one of the key features of SOD models, because this feature provides the basis of many down-stream vision tasks such as image retrieval [23], visual tracking [29], photographic composition [21], and image quality assessment [80]. We propose to test the performance of SOD models on images with unseen categories to verify the category sensitivity of SOD models. Given a SOD model trained without a certain category, if the model can still detect the salient object on the image with that category, it can be regarded as not sensitive to the category information. Specifically, based on the data for SOD, we have a series of models where each of them is trained by the data with images of one category removed. Each model is evaluated on images from all categories, respectively. We compare the relative performance changes between those models and the baseline model trained with images from all categories. Removing training images inevitably causes a performance drop. If the largest performance drop does not occur on the category that not exists on the training set, the SOD model can be regarded as category insensitivity.

Experimental results: We now test SOD models on images from unseen categories. Fig. 8 shows the relative performance changes of F-measure after removing a certain category for training. All results are the average of three runs. Removing a category during training does not significantly affect the test performance on that category, which indicates that SOD model is not sensitive to category information. For instance, removing the category ‘vertebrate’ during training causes the largest test performance drop of category ‘geological formation’, instead of that category itself. As shown in Fig. 7, the number of images from each category in the SOD dataset is not evenly distributed. Reducing the number of available training images has a significant impact on performance. For example, removing the category ‘vertebrate’ causes the largest performance drop in the ECSSD dataset, since this category has the largest number of training images. In contrast, removing categories such as ‘street sign’ and ‘plant’ have a very limited impact on the performance, since they have a relatively small number of training images. Therefore, we assume that instead of requiring category information, the SOD model needs to be trained with more images with various scenes to improve the performance.

5.2 Model Complexity

Settings: The complexity of models for different tasks is not necessarily the same. Current CNN-based SOD models are mostly built on top of the classification backbone models such as VGGNet [69] and ResNet [25]. Effective as these backbone models are on the classification, their large model complexity may not be necessary for the SOD task. Unlike the classification task that needs to learn category related features, the SOD task has almost no requirement for category information. We propose to analyze the complexity of models for the SOD task and the classification task from the perspective of model compression. As our proposed gOctConv is capable of eliminating redundant parameters with the help of the effective dynamic weight decay scheme, we can have a clear insight regarding the complexity of models for each task.

Experimental results: We use the dataset for classification metric in Sec. 5.1.2 as the training set. Since the training set contains only 644 images, a standard configuration of 300 epochs training is not enough for the convergence of a compact model. We therefore increase the number of the training epochs to 3000. Tab. 4 shows the model complexity for different tasks based on the CSNet × 1.5. The SOD model requires 36% parameters of the original model, while the classification model requires 65% of parameters, indicating that SOD models require fewer parameters as they need almost no category information. With this observation, we believe a more compact SOD model can be designed specifically for the SOD task.

5.3 Feature Requirements from the Extractor

Using features from different stages of the extractor: As a pixel-level prediction task, SOD should generate a high resolution output map. Therefore, it is a common choice for existing SOD models [49], [51], [94] to utilize features from different stages of the feature extractor. Utilizing more features from earlier stages results in higher resolution prediction maps, while introducing more computational complexity. Here we study whether using more features from earlier stages can improve the SOD performance. As shown in Fig. 2, the cross-stage fusion part takes in
features from different stages of the extractor. We now perform the ablation study using features from different numbers of stages. To minimize the impact of the initial number of channels, we use the CSNet×2-L model with learned channels in gOctConvs. As shown in Tab. 5, using more stages from the feature extractor results in better performance and causes larger model complexity. CSNet×2-L using stages 1 to 4 achieves a 0.2% gain (91.8 vs 91.6) with 37k (177k vs 141k) more parameters than CSNet×2-L using stages 2 to 4. Therefore, using more features from earlier stages does benefit the quality of SOD. In our work, as a trade-off between efficiency and performance, we choose to use the last three stages as the input of the cross-stage fusion part if not otherwise stated.

**Visualization of feature scale requirement:** Since the feature extractor of our CSNet is composed of gOctConvs, we can study the feature scale requirement of the feature extractor with the self-adaptive channel learning property of gOctConvs. We visualize the learned number of channels of gOctConvs in Fig. 10. It can be seen that as the network goes deeper, the feature extractor shows a trend of utilizing more low-resolution features. Within the same stage, high-resolution features are urgently used in the middle of the stage. Also, the model trained with dynamic weight decay has a more stable number of channel variation among different layers. Deeper layers contain more redundant channels compared with shallower ones.

### 5.4 ImageNet Pre-training

ImageNet pretraining has been proved to be very effective for many down-stream tasks, e.g., depth estimation, crowd counting, and bounding box regression. Utilizing ImageNet pretrained feature extractor has been a default configuration in CNN-based SOD models due to its effectiveness. We compare the SOD model convergence speed with/without ImageNet pre-training on both light-weight and large models, shown in Fig. 11. For large model CSF+ResNet, the ImageNet pre-training helps the model converge in fewer epochs. As shown in Sec. 5.1.2, the early stage of the model contains more general low-level features. Large SOTA SOD models need ImageNet pre-training because the universal low-level features in the early stage of the pre-trained ImageNet models may help the convergence of large models that contain a lot of trainable parameters. For light-weight model CSNet, the convergence speed between the model with ImageNet pre-training and model trained from scratch shows almost no difference. Light-weight models are too small to benefit from the convergence acceleration of ImageNet pre-training. We pre-train the extractor of the CSNet on ImageNet to see if ImageNet pre-training can further benefit the performance of the SOD task. As shown in Tab. 7, the ImageNet pre-trained CSNet×1.5-L has similar performance compared with the model trained from scratch. Therefore, the effect of ImageNet pre-training is limited for the light-weight SOD models. ImageNet pre-training, however, still benefits large model convergence.

6 Performance Analysis and Ablation

#### 6.1 Implementation

**Training:** Our method is implemented in PyTorch. We train the light-weight models using the Adam optimizer [38] with a batch-size of 24 for 300 epochs from scratch. Even with no ImageNet pre-training, the proposed CSNet still performs on par with large models based on pre-trained backbones [25], [69]. The learning rate is initially set to 1e-4, and is divided by 10 at the epochs of 200, and 250. We eliminate redundant weights and finetune the model for the last 20 epochs to compress models and get gOctConvs with the learnable channels of different resolutions. We only utilize random flip and crop for data augmentation. The weight decay of BatchNorms following gOctConvs is replaced with our proposed dynamic weight decay with the default weight of 3, while the weight decay for other weights is set to 5e-3 by default. For large models based on the pre-trained backbones, we train our models following the implementation of [49].

**Datasets:** While MSRA 10K [9], MSRA-B [52], DUT-O [87], and HKU-IS [42] datasets are used by earlier methods [20], [43], [46] for training salient object detectors, these datasets are either too small or lack diversity. We follow common settings of recent methods [49], [51], [77], [78], [96], [98] to train our models using the DUTS-TR [74] dataset, and evaluate the performance on several commonly used datasets, including ECSSD [86], PASCAL-S [47], DUT-O [87], HKU-IS [42], SOD [63], and DUTS-TE [74]. On ablation studies, the performance on the ECSSD dataset is reported.
Fig. 12. Visual qualitative comparison between our proposed CSNet and existing SOTA models. The last column shows a failure case of CSNet, which we assume is caused by the limited representation ability of the extremely small model parameters.

if not mentioned otherwise. For the analysis of the semantics of SOD models, we use our two proposed datasets as described in Sec. 5.1.1.

Evaluation metrics: The commonly used evaluation metrics maximum F-measure ($F_\beta$) [1] and MAE ($M$) [10] are used for evaluation. MACC of lightweight models is computed with an image size of $224 \times 224$.

6.2 Performance Analysis

In this section, we first evaluate the performance of our proposed light-weight model CSNet with fixed channels. Then, the performance of CSNet with learnable channels using dynamic weight decay is measured. We show that ImageNet pre-training is not inevitable for CNN-based SOD models. Fig. 12 shows the qualitative results of salient object detection using our proposed light-weight CSNet. Also, we transfer the proposed cross-stage fusion part to commonly used large backbones [25] to verify the cross-stage feature extraction ability.

Performance of CSNet with fixed channels in gOctConv: The extractor model is only composed of ILBlocks. As shown in Tab. 6, when replacing the gOctConvs in ILBlocks with Vanilla OctConvs, the extractor has $\times 8$ and $\times 7$ of original size in terms of parameters and MACC, while the performance gain is very limited. The large model complexity gap shows the efficiency of the simplified instance of gOctConv in the ILBlock. Tab. 6 shows feature extraction models with different split-ratios of high/low-resolution features. Extractors achieve a low complexity thanks to the simplified instance of gOctConvs. Benefiting from the within-stage multi-scale representation and the low-resolution features in ILBlock, the extractor-3/1 achieves a performance gain of 0.4% in F-measure with 80% MACC over the extractor-1/0. The gOctConvs in the cross-stage fusion part enhance the cross-stage multi-scale ability of the network while maintaining the high output resolution by utilizing features from different stages. As shown in Tab. 6, the CSNet-5/5 surpasses the extractor-3/1 by 1.4% in F-measure with fewer MACC. Even in the extreme case, the CSNet-0/1 with only low-resolution features in extractor performs on par with the extractor-1/0 that has all high-resolution features, while only requires 44% MACC of the extractor-1/0. However, manually tuning the overall split-ratio of feature channels of different resolutions may achieve a sub-optimal balance between performance and computational cost. To further verify the effectiveness of the cross-stage fusion (CSF) part on large models, we add this part into the commonly used backbone network ResNet [25] and Res2Net [15]. Tab. 7 shows that the ResNet+CSF achieves similar performance to the ResNet+PoolNet with 53% parameters and 21% MACC. Unlike other models (e.g., PoolNet) that eliminate downsampling operations to maintain a high feature resolution at high-levels of the backbone, the gOctConvs obtain both high and low resolution features across different stages of the backbone, achieving a high-resolution output while saving a large amount of computational cost.

Performance of CSNet with learnable channels in gOctConv: We further train the model with our proposed dynamic weight decay and obtain the learnable channels in gOctConv as described in Alg. 1. The obtained models are named CSNet-L. Tab. 11 shows that our proposed dynamic weight decay assisted pruning scheme can compress the model up to 18% of the original model size.
with a negligible performance drop. Compared with manually tuned split-ratio of feature resolution, the learnable channels of gOctConvs obtained by model compression achieves much better efficiency. As shown in Tab. 6, the compressed CSNet × 2-L outperforms the CSNet-5/5 by 1.6% with fewer parameters and comparable MACC. The CSNet × 1-L performs on par with CSNet-5/5 with about 45% parameters and about 70% MACC. Tab. 7 shows that CSNet-L series achieves comparable performance compared with some models with extensive parameters such as SRM [77], and Amulet [95] with ~ 0.2% parameters. Note that our light-weight models are trained from scratch while those large models are pre-trained with ImageNet. The performance gap between the proposed light-weight model and the SOTA models with extensive parameters and MACC is only ~ 2%. Utilizing new techniques, e.g., representative batch normalization [16] and receptive fields searching [17], will further close the gap with large models.

Comparison with light-weight models: To the best of our knowledge, we are the first to design an extremely light-weight model for the SOD task. For a more exhaustive analysis, we adopt several SOTA light-weight models, designed for other tasks such as classification and semantic segmentation, for salient object detection. All models share the same training configuration as in our training strategy. When transferring classification models to detection. All models share the same training configuration as in such as classification and semantic segmentation, for salient object detection. Therefore, we are the first to design an extremely light-weight model for the SOD task. We compare the run-time of our proposed CSNet with existing models from Tab. 7. As shown in Tab. 8. The run-time is tested on a single core of i7-8700K CPU using 224 × 224 images. Our proposed CSNet is x10 faster compared with the models designed for other tasks. However, there is still a gap between MACC and run-time, as current deep learning frameworks are not optimized for vanilla and our proposed gOctConvs yet.

**6.3 Ablations**

**Dynamic weight decay:** In this section, we assess the effectiveness of our proposed dynamic weight decay. We apply different degrees of weights to standard weight decay to balance model performance and sparsity, while keeping the weights for dynamic weight decay unchanged. We plug in our proposed dynamic weight decay into the weights of the BatchNorm layers while using the standard weight decay on remaining weights for a fair comparison. Fig. 13 shows the performance and complexity of our compressed model using dynamic/standard weight decay under different λ as shown in Eqn. (1). Applying different degrees of weight decay results in a trade-off between model performance and sparsity.

Fig. 14. The test MAE of models w/o dynamic weight decay.

**Fig. 13. Performance and complexity of our compressed model using dynamic/standard weight decay under different λ as shown in Eqn. (1). Applying different degrees of weight decay results in a trade-off between model performance and sparsity.**

**Table 6** Performance of CSNet with the fixed split-ratio of channels in gOctConvs, and CSNet with learnable channels. CSNet: model with the fixed split-ratio in gOctConvs. Extractor: the network composed of only ILBlocks. Vanilla: the Extractor made of only vanilla OctConvs. CSNet-L: the model with learnable channels using Alg. 1.

<table>
<thead>
<tr>
<th>Method</th>
<th>PARM.</th>
<th>MACC</th>
<th>$F_β$</th>
<th>$M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla</td>
<td>5/5</td>
<td>1457K</td>
<td>3.31G</td>
<td>88.4</td>
</tr>
<tr>
<td>Extractor</td>
<td>1/0</td>
<td>180K</td>
<td>0.80G</td>
<td>88.2</td>
</tr>
<tr>
<td></td>
<td>3/1</td>
<td>180K</td>
<td>0.64G</td>
<td>88.6</td>
</tr>
<tr>
<td></td>
<td>5/5</td>
<td>180K</td>
<td>0.45G</td>
<td>88.1</td>
</tr>
<tr>
<td></td>
<td>1/3</td>
<td>180K</td>
<td>0.30G</td>
<td>87.4</td>
</tr>
<tr>
<td></td>
<td>0/1</td>
<td>180K</td>
<td>0.20G</td>
<td>86.4</td>
</tr>
<tr>
<td>CSNet-L</td>
<td>×2</td>
<td>141K</td>
<td>0.72G</td>
<td>91.6</td>
</tr>
<tr>
<td></td>
<td>×1</td>
<td>94K</td>
<td>0.43G</td>
<td>90.0</td>
</tr>
</tbody>
</table>

**Fig. 14. The test MAE of models w/o dynamic weight decay.**
Can get a unique number of channel for each layer. As shown in Fig. 5, there is a clear gap between the large weights and the weights close to zero. Using arbitrary thresholds within this gap would almost have no difference to the final performance of models.

**Fixed pruning ratio/threshold:** We follow [53] to use the weight of the BatchNorm layer as the indicator of the channel importance. We modify the pruning method in [53] by using a fixed threshold to eliminate channels instead of using an fixed pruning ratio. Tab. 10 shows that pruning with a fixed threshold achieves better performance with fewer parameters than using a fixed pruning ratio. The reason behind this result is that different layers require a different number of channels. Therefore, pruning with a threshold can get a unique number of channel for each layer. As shown in Fig. 5, there is a clear gap between the large weights and the weights close to zero. Using arbitrary thresholds within this gap would almost have no difference to the final performance of models.

**Integrating dynamic weight decay into pruning methods:** By default, we use the pruning method in [53] to eliminate the redundant weights. Since our proposed dynamic weight decay focuses on introducing sparsity while maintaining a stable and compact distribution of weights among channels, it is orthogonal to commonly used pruning methods that focus on identifying unnecessary weights. Thus, we integrate the dynamic weight decay into several pruning methods as shown in Tab. 9. All configurations remain the same except for replacing the standard weight decay with our proposed dynamic weight decay. Pruning
The compression ratio of the CSNet with different initial channel widths. The pruning rate is defined as the ratio of model complexity between pruned parts and the complete CSNet.

<table>
<thead>
<tr>
<th>Width</th>
<th>Prune</th>
<th>×1</th>
<th>×1.2</th>
<th>×1.5</th>
<th>×1.8</th>
<th>×2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parms</td>
<td>N</td>
<td>211K</td>
<td>298K</td>
<td>455K</td>
<td>645K</td>
<td>788K</td>
</tr>
<tr>
<td></td>
<td>Y</td>
<td>94K</td>
<td>109K</td>
<td>118K</td>
<td>134K</td>
<td>141K</td>
</tr>
<tr>
<td>Ratio</td>
<td>N</td>
<td>55%</td>
<td>63%</td>
<td>74%</td>
<td>79%</td>
<td>82%</td>
</tr>
<tr>
<td></td>
<td>Y</td>
<td>30%</td>
<td>37%</td>
<td>46%</td>
<td>55%</td>
<td>61%</td>
</tr>
<tr>
<td>F Parsons</td>
<td>N</td>
<td>90.0</td>
<td>90.7</td>
<td>91.1</td>
<td>91.2</td>
<td>91.5</td>
</tr>
<tr>
<td></td>
<td>Y</td>
<td>90.0</td>
<td>90.7</td>
<td>91.2</td>
<td>91.3</td>
<td>91.6</td>
</tr>
</tbody>
</table>

The pruning rate is defined as the ratio of model complexity between pruned parts and the complete CSNet.

Pruning rate & Channel width: An initial model with a large channel width is required for learning more useful features. We linearly expand the number of channel of gOctConvs to enlarge the initial model capacity. A pruning rate is defined as the ratio of model complexity between pruned parts and the complete CSNet. Tab. 11 shows the pruning rate of CSNet with different initial channel widths. The split-ratio of gOctConvs for the initial model is set to 5/5. Larger initial width results in better performance as expected. As the initial width increases, the complexity of pruned models only has a small increment. The quality of the pruned model is dependant on the initial model size. Also, benefiting from the stable distribution introduced by dynamic weight decay, compressed models have similar or even better performance than the initial model.

7 Conclusion and Discussion

In this paper, we propose an extremely light-weight holistic model strongly tied to the SOD task, by abandoning the classification backbone and reducing the representation redundancy with a novel dynamic weight decay scheme. The dynamic weight decay scheme maintains a stable weights distribution across channels and stably boosts the sparsity of parameters during training, allowing 80% reduction in parameters with a negligible performance drop. Our proposed CSNet achieves comparable performance with ~ 0.2% parameters (100k) of large models on popular salient object detection benchmarks. Based on our proposed CSNet, we reveal several properties of the CNN-based SOD model including 1) SOD models are category insensitive and the detected salient objects are generic and category-independent, 2) ImageNet pre-training is not necessary for SOD training, and 3) SOD models require fewer parameters compared with classification models.

Our two major contributions: analyzing the semantics of SOD model and the extremely light-weight holistic SOD model are interdependent of each other. The intentional design of our holistic CSNet make it possible to analyze the semantics of the SOD model. CSNet is trained from scratch, and therefore be free from the potential influence of ImageNet pre-trained backbones, forming the basis of analyzing the category dependency of SOD models. Also, the self-adaptive property and the simple yet effective structure of CSNet benefit the analysis of SOD model complexity and feature requirements. From another perspective, the analysis of SOD model support the designing principle of CSNet. Our analysis proves that category information is not needed by the SOD model, therefore we can abandon the ImageNet pretrained backbone to reduce a great deal of redundancy for the SOD task. And we can design the holistic model specifically for the SOD task instead of adding extra modules on classification backbones to make up for the different between classification backbones and the SOD task.

Future research should focus on analyzing SOD models from other perspectives in particular with more diversity in SOD model structures, and building even more efficient models in terms of speed and accuracy. To facilitate follow-up works we share our code at https://mmcheng.net/sod100k/.

Acknowledgement. This research was supported by the Major Project for New Generation of AI under Grant No. 2018AAA0100400, NSFC (61620106008), S&T innovation project from Chinese Ministry of Education, the Fundamental Research Funds for the Central Universities (Nankai University, 63210309), and Tianjin Natural Science Foundation (18ZXZNGX00110).

References


R. Desimone and J. Duncan. Neural mechanisms of selective visual attention.


