Highly Efficient Salient Object Detection with 100K Parameters

Shang-Hua Gao[1], Yong-Qiang Tan[1], Ming-Ming Cheng[1⋆], Chengze Lu[1], Yunpeng Chen[2], Shuicheng Yan[2]

Nankai University[1], Yitu Technology[2]

Abstract. Salient object detection models often demand a considerable amount of computation cost to make precise prediction for each pixel, making them hardly applicable on low-power devices. In this paper, we aim to relieve the contradiction between computation cost and model performance by improving the network efficiency to a higher degree. We propose a flexible convolutional module, namely generalized Oct-Conv (gOctConv), to efficiently utilize both in-stage and cross-stages multi-scale features, while reducing the representation redundancy by a novel dynamic weight decay scheme. The effective dynamic weight decay scheme stably boosts the sparsity of parameters during training, supports learnable number of channels for each scale in gOctConv, allowing 80% of parameters reduce with negligible performance drop. Utilizing gOctConv, we build an extremely light-weighted model, namely CSNet, which achieves comparable performance with $\sim 0.2\%$ parameters (100k) of large models on popular salient object detection benchmarks. The source code will be made publicly available.

Keywords: Salient object detection, Highly efficient

1 Introduction

Salient object detection (SOD) is an important computer vision task with various applications in image retrieval [11], visual tracking [17], and weakly supervised semantic segmentation [19]. While convolutional neural networks (CNNs) based SOD methods have made significant progress, most of these methods focus on improving the state-of-the-art (SOTA) performance, by utilizing both fine details and global semantics [57,72,74,68], attention [2], as well as edge cues [8,61,75] *etc.* Despite the great performance, these models are usually resource-hungry, which are hardly applicable on low-power devices with limited storage/computational capability. How to build an extremely light-weighted SOD model with SOTA performance is an important but less explored area.

The SOD task requires generating accurate prediction scores for every image pixel, thus requires both large scale high level feature representations for correctly locating the salient objects, as well as fine detailed low level representations for precise boundary refinement [8,60,18]. There are two major challenges

⋆ M.M. Cheng is the corresponding author (cmm@nankai.edu.cn).

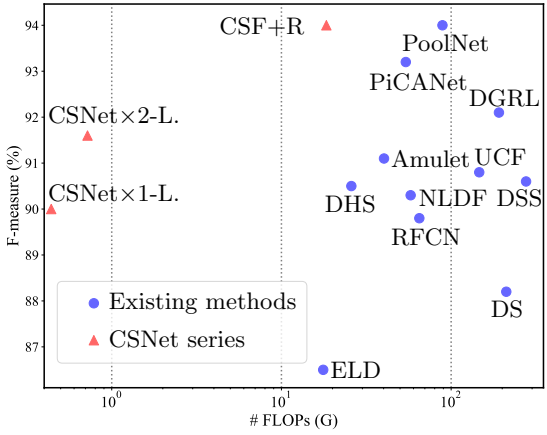

Fig. 1. FLOPs and performance of models on salient object detection task.

towards building an extremely light-weighted SOD models. **Firstly**, serious redundancy could appear when the low frequency nature of high level feature meets the the high output resolution of saliency maps. **Secondly**, SOTA SOD models [37,64,8,39] usually rely on ImageNet pre-trained backbone architectures [13,9] to extract features, which by itself is resource-hungry.

Very recently, the spatial redundancy issue of low frequency features has also been noticed by Chen *et al.* [3] in the context of image and video classification. As a replacement of vanilla convolution, they design an OctConv operation to process feature maps that vary spatially slower at a lower spatial resolution to reduce computational cost. However, directly using OctConv [3] to reduce redundancy issue in the SOD task still faces two major challenges. 1) Only utilizing two scales, *i.e.,* low and high resolutions as in OctConv, is not sufficient for fully reduce redundancy issues in the SOD task, which needs much stronger multi-scale representation ability than classification tasks. 2) The number of channels for each scale in OctConv is manually selected, requiring lots of efforts to re-adjust for saliency model as SOD task requires less category information.

In this paper, we propose a generalized OctConv (gOctConv) for building an extremely light-weighted SOD model, by extending the OctConv in the following aspects: 1). The flexibility to take inputs from arbitrary number of scales, from both in-stage features as well as cross-stages features, allows a much larger range of multi-scale representations. 2). We propose a dynamic weight decay scheme to support learnable number of channels for each scale, allowing 80% of parameters reduce with negligible performance drop.

Benefiting from the flexibility and efficiency of gOctConv, we propose a highly light-weighted model, namely CSNet, that fully explores both in-stage and **C**ross-**S**tages multi-scale features. As a bonus to the extremely low number of parameters, our CSNet could be directly trained from scratch without ImageNet pre-training, avoiding the unnecessary feature representations for dis-

tinguishing between various categories in the recognition task. In summary, we make two major contributions in this paper:

- We propose a flexible convolutional module, namely gOctConv, to efficiently utilize both in-stage and cross-stages multi-scale features for SOD task, while reducing the representation redundancy by a novel dynamic weight decay scheme.
- Utilizing gOctConv, we build an extremely light-weighted SOD model, namely CSNet, which achieves comparable performance with $\sim 0.2\%$ parameters (100k) of SOTA large models on popular SOD benchmarks.

2 Related Works

2.1 Salient Object Detection

Early works [25,55,66,77,4] mainly rely on hand-craft features to detect salient objects. [31,58,38] utilize CNNs to extract more informative features from image patches to improve the quality of saliency maps. Inspired by the fully convolutional networks (FCNs) [43], recent works [7,29,72,60,39,70] formulate the salient object detection as a pixel-level prediction task and predict the saliency map in an end-to-end manner using FCN based models. [18,59,72,74,51] capture both fine details and global semantics from different stages of the backbone network. [45,34,61,75] introduce edge cues to further refine the boundary of saliency maps. [72,76,62] improve the saliency detection from the perspective of network optimization. Despite the impressive performance, all these CNN based methods require ImageNet pre-trained powerful backbone networks as the feature extractor, which usually results in high computational cost.

2.2 Light-weighted Models

Currently, most light-weighted models that are initially designed for classification tasks utilize modules such as inverted block [22,21], channel shuffling [73,46], and SE attention module [21,54] to improve network efficiency. Classification tasks [52] predict semantic labels for an image, requiring more global information but fewer details. Thus, light-weighted models [22,46,73,21] designed for classification use aggressive downsampling strategies at earlier stages to save FLOPs, which are not applicable to be the feature extractor for SOD task that requires multi-scale information with both coarse and fine features. Also, SOD task focuses on determine the salient region while classification tasks predicts category information. To improve performance under limited computing budget, the allocation of computational resources, *i.e.,* feature resolution, channels, for saliency models should be reconsidered.

2.3 Network Pruning

Many network pruning methods [32,44,42,16,41,15] have been proposed to prune unimportant filters especially on channel level. [32,14] use the norm criterion to

estimate redundant filters. [44] prunes filters based on statistics information of the next layer. [42] reuses the scaling factor of batch normalization layer as the indicator of filter importance. [15] computes the geometric median of weights to select filters. [41] utilizes generated weights to estimate the performance of remaining filters. Mostly pruning approaches still rely on regularization tricks such as weight decay to introduce sparsity to filters. Our proposed dynamic weight decay stably introduces sparsity for assisting pruning algorithms to prune redundant filters, resulting in learnable channels for each scale in our proposed gOctConv.

3 Light-weighted Network with Generalized OctConv

3.1 Overview of Generalized OctConv

Originally designed to be a replacement of traditional convolution unit, the vanilla OctConv [3] shown in Fig. 2 (a) conducts the convolution operation across high/low scales within a stage. Details about OctConv is shown in supplementary. However, only two-scales within a stage can not introduce enough multi-scale information required for SOD task. The channels for each scale in vanilla OctConv is manually set, requires lots of efforts to re-adjust for saliency model as SOD task requires less category information. Therefore, we propose a generalized OctConv (gOctConv) allows arbitrary number of input reolutions from both in-stage and cross-stages conv features with learnable number of channels as shown in Fig. 2 (b). As a generalized version of vanilla OctConv, gOctConv improves the vanilla OctConv for SOD task in following aspects: 1). Arbitrary numbers of input and output scales is available to support a larger range of multi-scale representation. 2). Expect for in-stage features, the gOctConv can also process cross-stages features with arbitrary scales from the feature extractor. 3). The gOctConv supports learnable channels for each scale through our proposed dynamic weight decay assisting pruning scheme. 4). Cross-scales feature interaction can be turned off to support a large complexity flexibility.

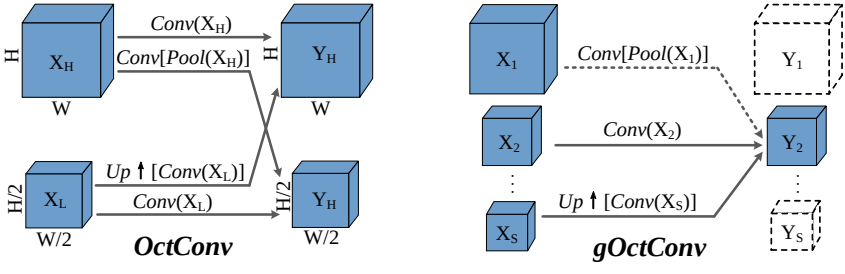

Fig. 2. While originally designed to be a replacement of traditional convolution unit, the OctConv [3] takes two hight/low resolution inputs from the same stage with fixed number of feature channels. Our gOctConv allows arbitrary number of input reolutions from both in-stage and cross-stages conv features with learnable number of channels.

The flexible gOctConv allows many instances under different designing requirements. We will a give detailed introduction of different instances of gOctConvs in following light-weighted model designing.

3.2 Light-weighted Model Composed of gOctConvs

Overview. As shown in Fig. 3, our proposed light-weighted network, consisting of a feature extractor and a cross-stages fusion part, synchronously processes features with multiple scales. The feature extractor is stacked with our proposed in-layer multi-scale block, namely 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. The cross-stages fusion part, composed of gOctConvs, processes features from stages of the feature extractor to get a high-resolution output.

In-layer Multi-scale Block. ILBlock enhances the multi-scale representation of features within a stage. gOctConvs are utilized to introduce multi-scale within ILBlock. Vanilla OctConv requires about 60% FLOPs [3] to achieves the similar performance to standard convolution, which is not enough for our objective of designing a highly light-weighted model. To save computational cost, interact features with different scales in every layer is unnecessary. Therefore, we apply an instance of gOctConv that each input channel corresponds to an output channel with the same resolution through eliminating the cross scale operations. A depthwise operation within each scale in utilized to further save computational cost. This instance of gOctConv only requires about $1/channel$ FLOPs compared with vanilla OctConv as analyzed in supplementary. ILBlock is composed of a vanilla OctConv and two 3×3 gOctConvs as shown in Fig. 3. Vanilla OctConv interacts features with two scales and gOctConvs extract features within each scale. Multi-scale features within a block are separately processed and interacted alternately. Each convolution is followed by the BatchNorm [24] and PRelu [12]. Initially, we roughly double the channels of ILBlocks as the resolution decreases, except for the last two stages that have the same channel number. Unless otherwise stated, the channels for different scales in ILBlocks

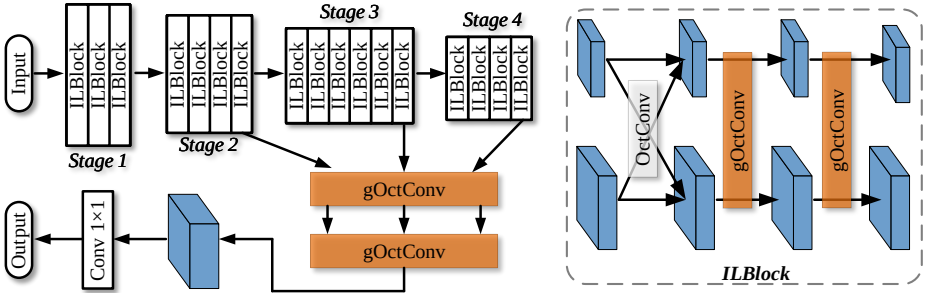

Fig. 3. Illustration of our salient object detection pipeline, which uses gOctConv to extract both in-stage and cross-stages multi-scale features in a highly efficient way.

are set evenly. Learnable channels of OctConvs then are obtained through the scheme as described in Sec. 3.3.

Cross-stages Fusion. To retain a high output resolution, common methods retain high feature resolution on high-level of the feature extractor, inevitablely increase the computational redundancy. Instead, we simply use gOctConvs to fuse multi-scale features from stages of the feature extractor and generate the high-resolution output. As a trade-off between efficiency and performance, features from last three stages are used. A gOctConv 1×1 takes features with different scales from the last conv of each stage as input and conducts a cross-stages 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 detailed in supplementary. Features are then are sent to another gOctConv 1×1 to generate features with the highest resolution. Another standard conv 1×1 outputs the prediction result of saliency map. Learnable channels of gOctConvs in this part are also obtained.

3.3 Learnable Channels for gOctConv

We propose to get learnable channels for each scale in gOctConv by utilizing our proposed dynamic weight decay assisted pruning during training. Dynamic weight decay maintains a stable weights distribution among channels while introducing sparsity, helping pruning algorithms to eliminate redundant channels with negligible performance drop.

Dynamic Weight Decay. The commonly used regularization trick weight decay [27,69] endows CNNs with better generalization performance. Mehta *et al.*[47] shows that weight decay introduces sparsity into CNNs, which helps to 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 [32,44,42,16,41,15]. The common implementation of weight decay is by adding the L2 regularization to the loss function, which can be written as follows:

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

where $\boldsymbol{L_0}$ is the loss for the specific task, $\boldsymbol{w_i}$ is the weight of ith layer, and λ is the weight for weight decay. During back propagation, the weight $\boldsymbol{w_i}$ is updated as

$$\boldsymbol{w_i} \leftarrow \boldsymbol{w_i} - \nabla f_i\left(\boldsymbol{w_i}\right) - \lambda \boldsymbol{w_i}, \tag{2}$$

where $\nabla f_i\left(\boldsymbol{w_i}\right)$ is the gradient to be updated, and $\lambda \boldsymbol{w_i}$ 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 (a) shows that diverse weights cause unstable distribution of outputs

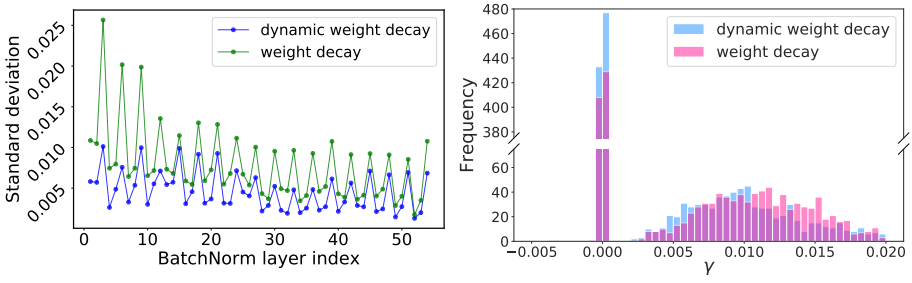

Fig. 4. a) Left: The averaged standard deviation of outputs among channels from BatchNorm layer in models trained with/without dynamic weight decay. b) Right: Distribution of γ in Eqn. (5) of models trained with/without dynamic weight decay.

among channels. Ruan *et al.*[6] reveals 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 [23,6]. We propose to relieve diverse outputs among channels with no extra cost during 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

$$\boldsymbol{w_i} \leftarrow \boldsymbol{w_i} - \nabla f_i\left(\boldsymbol{w_i}\right) - \lambda_d \, \mathrm{S}\left(\boldsymbol{x_i}\right) \boldsymbol{w_i}, \tag{3}$$

where λ_d is the weight of dynamic weight decay, $\boldsymbol{x_i}$ denotes the features calculated by $\boldsymbol{w_i}$, and $\mathrm{S}\left(\boldsymbol{x_i}\right)$ is the metric of the feature, 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) [36] as the metric for a certain channel:

$$\mathrm{S}\left(\boldsymbol{x_i}\right) = \frac{1}{HW} \sum_{h=0}^{H} \sum_{w=0}^{W} \boldsymbol{x_i}_{h,w}, \tag{4}$$

where H and W are the height and width of the feature map $\boldsymbol{x_i}$. The dynamic weight decay with the GAP metric ensures that the weights producing large value features are suppressed, giving a compact and stable weights and outputs distribution as revealed in Fig. 4. Also, the metric can be defined as other forms to suit certain tasks as we will study in our future work. Please refer to Sec. 4.3 for a more detailed interpretation of dynamic weight decay.

Learnable channels with model compression. Now, we incorporate dynamic weight decay with pruning algorithms to remove redundant weights, so as to get learnable channels of each scale in gOctConvs. In this paper, we follow [42] to use

the weight of BatchNorm layer as the indicator of the channel importance. The BatchNorm operation [24] is written as follows:

$$y = \frac{x - \mathrm{E}(x)}{\sqrt{\mathrm{Var}(x) + \epsilon}}\gamma + \beta, \tag{5}$$

where x and y are input and output features, $\mathrm{E}(x)$ and $\mathrm{Var}(x)$ are the mean and variance, respectively, and ϵ is a small factor to avoid zero variance. γ and β are learned factors. We apply the dynamic weight decay to γ during training. Fig. 4 (b) reveals that there is a clear gap between important and redundant weights, and unimportant weights are suppressed to nearly zero ($\boldsymbol{w_i} < 1e-20$). 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 getting learnbale channels of gOctConvs is illustrated in Alg. 1.

Algorithm 1 Learnable 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 $i \in [1, MaxIteration]$ **do**
2: Feed input X into 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 fine-tune remaining weights.

4 Experiments

4.1 Implementation

Training. The implementation of the proposed method is based on the PyTorch framework. For light-weighted models, we train models using the Adam optimizer [26] with a batch-size of 24 for 300 epochs from scratch. Even with no ImageNet pre-training, the proposed CSNet still achieves comparable performance to large models based on pre-trained backbones [53,13]. The learning rate is set to 1e-4 initially, and divided by 10 at the epochs of 200, and 250. We eliminate redundant weights and fine-tune the model for the last 20 epochs to compress models and get gOctConvs with the learnable channels of different resolutions. We only use the data augmentation of random flip and crop. The weight decay of BatchNorms following gOctConvs is replaced with our proposed dynamic weight decay with the weight of 3 by default 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 [37].

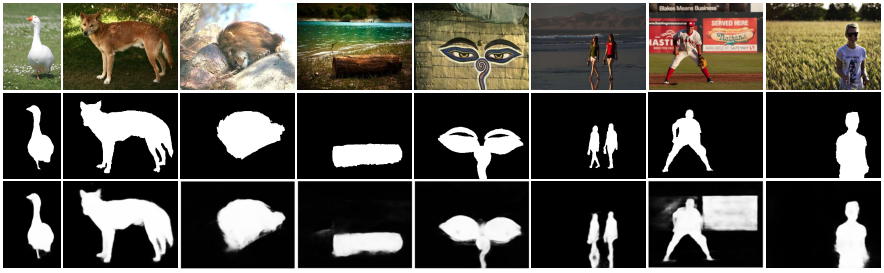

Fig. 5. Visualisation of predicted results on salient object detection. Each row gives the image, GT, and predicted result, respectively.

Datasets. While MSRA 10K [4], MSRA-B [40], DUT-O [66], and HKU-IS [30] datasets are used by earlier methods [10,33,31] for training salient object detectors, these datasets are either too small or lack of diversity. We following common settings of recent methods [74,39,37,60,59,75] to train our models using the DUTS-TR [56] dataset, and evaluate the performance on several commonly used datasets, including ECSSD [65], PASCAL-S [35], DUT-O [66], HKU-IS [30], SOD [49], and DUTS-TE [56]. On ablation studies, the performance on the ECSSD dataset is reported if not mentioned otherwise.

Evaluation metrics. The commonly used evaluation metrics maximum F-measure (F_β) [1] and MAE (M) [5] are used for evaluation. FLOPs of light-weighted models are computed with an image size of 224×224.

4.2 Performance Analysis

In this section, we firstly evaluate the performance of our proposed light-weighted model CSNet with fixed channels. Then, the performance of CSNet with learnable channels using dynamic weight decay is also evaluated. Fig. 5 shows the visualized results of salient object detection using our proposed light-weighted CSNet. Also, we transfer the proposed cross-stages fusion part to commonly used large backbones [13] to verify the cross-stages feature extraction ability.

Performance of CSNet with fixed channels in gOctConv. The extractor model only composed of ILBlocks. With fixed parameters, we adjust the split-ratio of channels for high/low resolution features in gOctConvs of ILBlocks to construct models with different FLOPs, denoted by C_H/C_L. Tab. 1 shows feature extraction models with different split-ratios of high/low resolution features. Extractors achieve an low complexity thanks to the simplified instance of gOctConvs. Benefiting from the in-stage multi-scale representation and the low scale features in ILBlock, the extractor-3/1 achieves performance gain of 0.4% in terms of F-measure with 80% FLOPs over the extractor-1/0. The gOctConvs in cross-stages fusion part enhance the cross-stages multi-scale ability of the network

Method		PARM.	FLOPs	$F_\beta \uparrow$	$M \downarrow$
Extractor	1/0	180K	0.80G	88.2	0.088
	3/1	180K	0.64G	**88.6**	**0.085**
	5/5	180K	0.45G	88.1	0.086
	1/3	180K	0.30G	87.4	0.090
	0/1	180K	0.20G	86.4	0.095
CSNet	1/0	211K	0.91G	90.0	**0.076**
	3/1	211K	0.78G	89.9	0.077
	5/5	211K	0.61G	**90.0**	0.077
	1/3	211K	0.47G	89.2	0.082
	0/1	211K	0.35G	88.2	0.089
CSNet-L	×2	140K	0.72G	**91.6**	**0.066**
	×1	94K	0.43G	90.0	0.075

Table 1. Performance of CSNet with the fixed split-ratio of channels in gOctConvs, and CSNet with learnable channels. CSNet denotes the CSNet with the fixed split-ratio in gOctConvs. Extractor denotes the network only composed of ILBlocks. CSNet-L denotes the model with learnable channels using Alg. 1.

while maintaining the high output resolution by utilizing features from different stages. As shown in Tab. 1, the CSNet-5/5 surpasses the extractor-3/1 by 1.4% in terms of F-measure with fewer FLOPs. Even in extreme case that the CSNet-0/1 with only low resolution features in extractor has comparable performance with 44% FLOPs over extractor-1/0 with all high resolution features. However, manually tune the overall split-ratio of feature channels of different resolution may achieves sub-optimal balance between performance and computational cost. To further verify the effectiveness of the cross-stage fusion (CSF) part on large models, we implement this part into commonly used backbone network ResNet [13]. As shown in Tab. 2, the ResNet+CSF achieves similar performance to the ResNet+PoolNet with 53% parameters and 21% FLOPs. Unlike other models such as PoolNet that eliminates downsampling operations to maintain a high feature resolution on high-levels of the backbone, the gOctConvs obtains both high and low resolution features accross different stages of the backbone, getting 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 get the learnable channels for gOctConv as described in Alg. 1, named CSNet-L. The channel for each gOctConv is expanded to enlarge the available space for compression. Models with channels expanded to k times are denoted by CSNet-$\times k$. Tab. 4 shows that our proposed dynamic weight decay assisted pruning scheme can compress the model up to 18% of the original model size with 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. 1, the compressed CSNet×2-L outperforms the CSNet-5/5 by 1.6% with fewer parameters and comparable FLOPs. The CSNet×1-L

Model	Complexity		ECSSD		PASCAL-S		DUT-O		HKU-IS		SOD		DUTS-TE	
	#PARM.	FLOPs	F_β	M	F_β	M	F_β	M	F_β	M	F_β	M	F_β	M
ELD [10]	43.15M	17.63G	.865	.981	.767	.121	.719	.091	.844	.071	.760	.154	-	-
DS [33]	134.27M	211.28G	.882	.122	.765	.176	.745	.120	.865	.080	.784	.190	.777	.090
DCL [31]	-	-	.896	.080	.805	.115	.733	.094	.893	.063	.831	.131	.786	.081
RFCN [58]	19.08M	64.95G	.898	.097	.827	.118	.747	.094	.895	.079	.805	.161	.786	.090
DHS [38]	93.76M	25.82G	.905	.062	.825	.092	-	-	.892	.052	.823	.128	.815	.065
MSR [29]	-	-	.903	.059	.839	.083	.790	.073	.907	.043	.841	.111	.824	.062
DSS [18]	62.23M	276.37G	.906	.064	.821	.101	.760	.074	.900	.050	.834	.125	.813	.065
NLDF [45]	35.48M	57.73G	.903	.065	.822	.098	.753	.079	.902	.048	.837	.123	.816	.065
UCF [72]	29.47M	146.42G	.908	.080	.820	.127	.735	.131	.888	.073	.798	.164	.771	.116
Amulet [71]	33.15M	40.22G	.911	.062	.826	.092	.737	.083	.889	.052	.799	.146	.773	.075
GearNet [20]	-	-	.923	.055	-	-	.790	.068	.934	.034	.853	.117	-	-
PAGR [74]	-	-	.924	.064	.847	.089	.771	.071	.919	.047	-	-	.854	.055
SRM [59]	53.14M	36.82G	.916	.056	.838	.084	.769	.069	.906	.046	.840	.126	.826	.058
DGRL [60]	161.74M	191.28G	.921	.043	.844	.072	.774	.062	.910	.036	.843	.103	.828	.049
PiCANet [39]	47.22M	54.05G	.932	.048	.864	.075	.820	.064	.920	.044	.861	.103	.863	.050
PoolNet [37]	68.26M	88.89G	.940	.042	.863	.075	.830	.055	.934	.032	.867	.100	.886	.040
Light-weighted models designed for other tasks:														
Eff.Net [54]	8.64M	2.62G	.828	.129	.739	.158	.696	.129	.807	.116	.712	.199	.687	.135
Sf.Netv2 [46]	9.54M	4.35G	.870	.092	.781	.127	.720	.100	.853	.078	.779	.163	.743	.096
ENet [50]	0.36M	0.40G	.857	.107	.770	.138	.730	.109	.839	.094	.741	.183	.730	.111
CGNet [63]	0.49M	0.69G	.868	.099	.784	.130	.727	.108	.849	.088	.772	.168	.742	.106
DABNet [28]	0.75M	1.03G	.877	.091	.790	.123	.747	.094	.862	.078	.778	.157	.759	.093
ESPNetv2 [48]	0.79M	0.31G	.889	.081	.795	.119	.760	.088	.872	.069	.780	.157	.765	.089
BiseNet [67]	12.80M	2.50G	.894	.078	.817	.115	.762	.087	.872	.071	.796	.148	.778	.084
Ours:														
CSF+R	36.37M	18.40G	.940	.041	.866	.073	.821	.055	.930	.033	.866	.106	.881	.039
CSNet×1-L	94K	0.43G	.900	.075	.819	.110	.777	.087	.889	.065	.809	.149	.799	.082
CSNet×2-L	140K	0.72G	.916	.066	.835	.102	.792	.080	.899	.059	.825	.137	.819	.074

Table 2. Performance and complexity comparison with state-of-the-art methods. +R denotes using the ImageNet pre-trained ResNet50 [13] backbone network. Unlike previous methods that require the ImageNet pre-trained backbone, our proposed light-weighted CSNet is trained from scratch without ImageNet pre-training.

achieves comparable performance compared with CSNet-5/5 with 45% parameters and 70% FLOPs. Tab. 2 shows that CSNet-L series achieve comparable performance compared with some models with extensive parameters such as SRM [59], and Amulet [72] with $\sim 0.2\%$ parameters. Note that our light-weighted models are trained from scratch while those large models are pre-trained with ImageNet. The performance gap between the proposed light-weighted model and the SOTA models with extensive parameters and FLOPs is only $\sim 2\%$.

Comparison with light-weighted models. To the best of our knowledge, we are the first work that aims to design an extremely light-weighted model for SOD task. Therefore, we port several SOTA light-weighted models designed for other tasks such as classification and semantic segmentation for comparison. All models share the same training configuration with our training strategy, and details are introduced in supplementary. Tab. 2 shows that our proposed models have considerable improvements compared with those light-weighted models.

4.3 Dynamic Weight Decay

In this section, we verify the effectiveness of our proposed dynamic weight decay. We apply different degrees of standard weight decay to achieve the trade-off

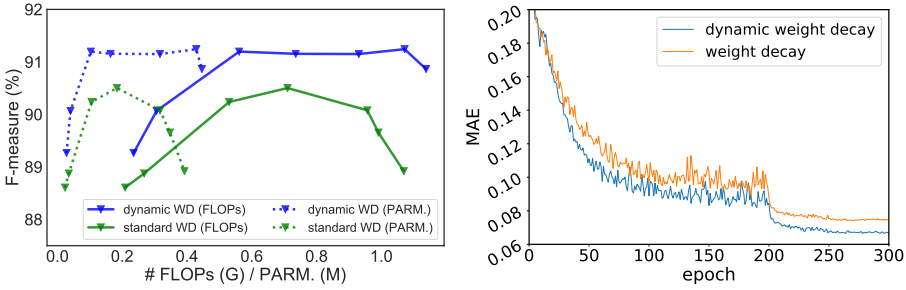

Fig. 6. a) Left: Performance and complexity of compressed model using dynamic/standard weight decay under different λ as shown in Eqn. (1). b) Right: The test MAE of models with/without dynamic weight decay.

Pruning Filters [32]					Geometric-Median [15]				
	PARM.	FLOPs	F_β	M		PARM.	FLOPs	F_β	M
Standard	227K	0.69G	88.7	0.080	Standard	227K	0.70G	88.7	0.083
Dynamic	226K	0.69G	89.4	0.078	Dynamic	226K	0.68G	89.6	0.082

Table 3. Integrating dynamic weight decay into pruning methods. Standard/Dynamic denote the standard/dynamic weight decay, respectively.

between model performance and sparsity, while keeping the weights for dynamic weight decay unchanged. We insert our proposed dynamic weight decay to the weights of BatchNorm layers while using the standard weight decay on remaining weights for a fair comparison. Fig. 6 (a) shows the performance and complexity of the compressed model using dynamic/standard weight decay under different λ in Eqn. (1). The λ_d in Eqn. (3) for dynamic weight decay on BatchNorm is set to 3 by default. Models trained with dynamic weight decay have better performance under the same complexity. Also, the performance of dynamic weight decay based models is less sensitive to the model complexity. We eliminate redundant channels according to the absolute value of γ in Eqn. (5) as described in Sec. 3.3. Fig. 4 (b) shows the distribution of γ for models trained with/without dynamic weight decay. By suppressing weights according to features, dynamic weight decay enforces the model with more sparsity. Fig. 4 (a) reveals the average standard deviation of outputs among channels from the BatchNorm layer of models trained with/without dynamic weight decay. Features of dynamic weight decay based model are more stabilized due to the stable weight distribution. Fig. 6 (b) shows the testing MAE of each epoch with/without dynamic weight decay. Training with dynamic weight decay brings better performance in terms of MAE and faster convergence. The dynamic weight decay generalizes well on other tasks as shown in supplementary.

Width	Prune	×1	×1.2	×1.5	×1.8	×2.0
Parms	N	211K	298K	455K	645K	788K
	Y	94K	109K	118K	134K	140K
Ratio		55%	63%	74%	79%	82%
FLOPs	N	0.61G	0.82G	1.17G	1.58G	1.87G
	Y	0.43G	0.52G	0.63G	0.71G	0.72G
Ratio		30%	37%	46%	55%	61%
F_β	N	90.1	90.7	91.1	91.2	91.5
	Y	90.0	90.7	91.2	91.3	91.6

Table 4. The compression ratio of CSNet with different initial channel widths. The pruning rate is defined as the ratio of model complexity between pruned parts and complete CSNet.

Cooperating with pruning methods. By default, we use the pruning method in [42] to eliminate 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 use pruning methods that focus on identify unnecessarily weights. Therefore, we integrate the dynamic weight decay into several pruning methods as shown in Tab. 3. All configurations remain the same except for replacing the standard weight decay to our proposed dynamic weight decay. Pruning methods [32,15] equipped with dynamic weight decay achieve better performance under the similar parameters.

4.4 CSNet with Learned Channels in gOctConv

Pruning rate & Channel width. An initial training space with a large channel width is required for learning more useful features. To enlarge the available training space, we linearly expand the channel number of gOctConvs. A pruning rate is defined as the ratio of model complexity between pruned parts and complete CSNet. Tab. 4 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 rises, the complexity of pruned models only has a limited increment. The quality of the pruned model is dependant on the available training space. With a large enough training space, results are closing to the optimal. Also, benefited from the stable distribution introduced by dynamic weight decay, compressed models have similar or even better performance compared with the initial model.

Visualization of channels of gOctConvs. We visualize the learned channel number of gOctConvs in Fig. 7. It can be seen that as the network goes deeper, the feature extraction network shows a trend of utilizing more low resolution features. Within the same stage, high resolution features are urged in the middle of the stage. Also, the model trained with dynamic weight decay has a stabler

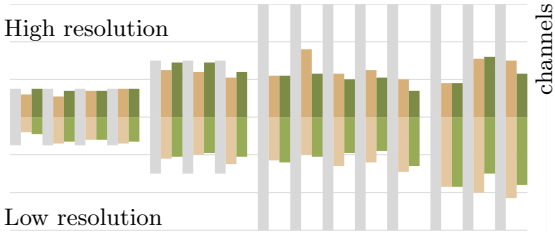

Fig. 7. Visualization of the feature extractor of CSNet. Gray is CSNet with fixed channel, Yellow and Green are the CSNet-L trained with standard/dynamic weight decay, respectively.

channel number variation among different layers. Deeper layers contain more redundant channels compared with shallow layers.

4.5 Run-Time

CSNet is designed to be light-weighted and high efficient on SOD task. We compare the run-time of our proposed CSNet with existing models from Tab. 2 as shown in Tab. 5. The run-time is tested with an image of 224 × 224 on a single core of i7-8700K CPU. Our proposed CSNet has more than x10 acceleration compared with large-weight models. With similar speed, CSNet achieves up to 6% gain in F-measure compared with those models designed for other tasks. However, there is still a gap between FLOPs and run-time, as current deep learning frameworks are not optimized for vanilla and our proposed gOctConvs yet.

Method	FLOPs	Run-time	Method	FLOPs	Run-time
PiCANet [38]	54.06G	2850.2ms	PoolNet [37]	88.89G	997.3ms
ENet [50]	0.40G	89.9ms	ESPNetv2 [48]	0.31G	186.3ms
CSNet×1	0.61G	135.9ms	CSNet×1-L	0.43G	95.3ms

Table 5. Run-time using 224 × 224 input on a single core of i7-8700K CPU.

5 Conclusion

In this paper, we propose the generalized OctConv with more flexibility to efficiently utilize both in-stage and cross-stages multi-scale features, while reducing the representation redundancy by a novel dynamic weight decay scheme. The dynamic weight decay scheme maintains a stable weights distribution among channels and stably boosts the sparsity of parameters during training. Dynamic weight decay supports learnable number of channels for each scale in gOctConvs, allowing 80% of parameters reduce with negligible performance drop. Utilizing

different instances of gOctConvs, we build an extremely light-weighted model, namely CSNet, which achieves comparable performance with $\sim 0.2\%$ parameters (100k) of large models on popular salient object detection benchmarks. The source code will be made publicly available.

References

1. Achanta, R., Hemami, S., Estrada, F., Süsstrunk, S.: Frequency-tuned salient region detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1597–1604 (2009)
2. Chen, S., Tan, X., Wang, B., Hu, X.: Reverse attention for salient object detection. In: European Conference on Computer Vision (2018)
3. Chen, Y., Fan, H., Xu, B., Yan, Z., Kalantidis, Y., Rohrbach, M., Yan, S., Feng, J.: Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution. In: IEEE International Conference on Computer Vision (ICCV) (2019)
4. Cheng, M.M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.M.: Global contrast based salient region detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(3), 569–582 (2015)
5. Cheng, M.M., Warrell, J., Lin, W.Y., Zheng, S., Vineet, V., Crook, N.: Efficient salient region detection with soft image abstraction. In: IEEE International Conference on Computer Vision (ICCV). pp. 1529–1536 (2013)
6. Dongsheng, R., Jun, W., Nenggan, Z.: Linear context transform block. arXiv preprint arXiv:1909.03834 (2019)
7. Fan, D.P., Cheng, M.M., Liu, J.J., Gao, S.H., Hou, Q., Borji, A.: Salient objects in clutter: Bringing salient object detection to the foreground. In: European Conference on Computer Vision (ECCV) (September 2018)
8. Feng, M., Lu, H., Ding, E.: Attentive feedback network for boundary-aware salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
9. Gao, S.H., Cheng, M.M., Zhao, K., Zhang, X.Y., Yang, M.H., Torr, P.: Res2net: A new multi-scale backbone architecture. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
10. Gayoung, L., Yu-Wing, T., Junmo, K.: Deep saliency with encoded low level distance map and high level features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
11. He, J., Feng, J., Liu, X., Tao, C., Chang, S.F.: Mobile product search with bag of hash bits and boundary reranking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
12. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: IEEE International Conference on Computer Vision (ICCV). pp. 1026–1034 (2015)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
14. He, Y., Kang, G., Dong, X., Fu, Y., Yang, Y.: Soft filter pruning for accelerating deep convolutional neural networks. In: International Joint Conference on Artificial Intelligence (IJCAI) (2018)

15. He, Y., Liu, P., Wang, Z., Hu, Z., Yang, Y.: Filter pruning via geometric median for deep convolutional neural networks acceleration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4340–4349 (2019)
16. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: IEEE International Conference on Computer Vision (ICCV). pp. 1389–1397 (2017)
17. Hong, S., You, T., Kwak, S., Han, B.: Online tracking by learning discriminative saliency map with convolutional neural network. In: International Conference on Machine Learning (ICML) (2015)
18. Hou, Q., Cheng, M.M., Hu, X., Borji, A., Tu, Z., Torr, P.: Deeply supervised salient object detection with short connections. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(4), 815–828 (2019). https://doi.org/10.1109/TPAMI.2018.2815688
19. Hou, Q., Jiang, P.T., Wei, Y., Cheng, M.M.: Self-erasing network for integral object attention. In: NeurIPS (2018)
20. Hou, Q., Liu, J., Cheng, M.M., Borji, A., Torr, P.H.: Three birds one stone: a unified framework for salient object segmentation, edge detection and skeleton extraction. arXiv preprint arXiv:1803.09860 (2018)
21. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. arXiv preprint arXiv:1905.02244 (2019)
22. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
23. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
24. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (ICML) (2015)
25. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(11), 1254–1259 (1998)
26. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2014)
27. Krogh, A., Hertz, J.A.: A simple weight decay can improve generalization. In: Advances in Neural Information Processing Systems (NIPS). pp. 950–957 (1992)
28. Li, G., Kim, J.: Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. In: British Machine Vision Conference (BMVC) (2019)
29. Li, G., Xie, Y., Lin, L., Yu, Y.: Instance-level salient object segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
30. Li, G., Yu, Y.: Visual saliency based on multiscale deep features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
31. Li, G., Yu, Y.: Deep contrast learning for salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
32. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: International Conference on Learning Representations (ICLR) (2016)
33. Li, X., Zhao, L., Wei, L., Yang, M.H., Wu, F., Zhuang, Y., Ling, H., Wang, J.: Deepsaliency: Multi-task deep neural network model for salient object detection. IEEE Transactions on Image Processing **25**(8), 3919 – 3930 (Aug 2016). https://doi.org/10.1109/TIP.2016.2579306

34. Li, X., Yang, F., Cheng, H., Liu, W., Shen, D.: Contour knowledge transfer for salient object detection. In: European Conference on Computer Vision (ECCV). pp. 355–370 (2018)
35. Li, Y., Hou, X., Koch, C., Rehg, J.M., Yuille, A.L.: The secrets of salient object segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2014)
36. Lin, M., Chen, Q., Yan, S.: Network in network. In: International Conference on Learning Representations (ICLR) (2013)
37. Liu, J.J., Hou, Q., Cheng, M.M., Feng, J., Jiang, J.: A simple pooling-based design for real-time salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
38. Liu, N., Han, J.: Dhsnet: Deep hierarchical saliency network for salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
39. Liu, N., Han, J., Yang, M.H.: Picanet: Learning pixel-wise contextual attention for saliency detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
40. Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., Shum, H.Y.: Learning to detect a salient object. IEEE Trans Pattern Anal Mach Intell **33**(2), 353–367 (2011)
41. Liu, Z., Mu, H., Zhang, X., Guo, Z., Yang, X., Cheng, T.K.T., Sun, J.: Metapruning: Meta learning for automatic neural network channel pruning. In: IEEE International Conference on Computer Vision (ICCV) (2019)
42. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: IEEE International Conference on Computer Vision (ICCV). pp. 2736–2744 (2017)
43. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3431–3440 (2015)
44. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: IEEE International Conference on Computer Vision (ICCV). pp. 5058–5066 (2017)
45. Luo, Z., Mishra, A., Achkar, A., Eichel, J., Li, S., Jodoin, P.M.: Non-local deep features for salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
46. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: European Conference on Computer Vision (ECCV). pp. 116–131 (2018)
47. Mehta, D., Kim, K.I., Theobalt, C.: On implicit filter level sparsity in convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 520–528 (2019)
48. Mehta, S., Rastegari, M., Shapiro, L., Hajishirzi, H.: Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9190–9200 (2019)
49. Movahedi, V., Elder, J.H.: Design and perceptual validation of performance measures for salient object segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW). pp. 49–56 (June 2010)
50. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: A deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147 (2016)

51. Piao, Y., Ji, W., Li, J., Zhang, M., Lu, H.: Depth-induced multi-scale recurrent attention network for saliency detection. In: IEEE International Conference on Computer Vision (ICCV) (October 2019)
52. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)
53. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (ICLR) (2014)
54. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 (2019)
55. Wang, J., Jiang, H., Yuan, Z., Cheng, M.M., Hu, X., Zheng, N.: Salient object detection: A discriminative regional feature integration approach. International Journal of Computer Vision **123**(2), 251–268 (2017). https://doi.org/10.1007/s11263-016-0977-3
56. Wang, L., Lu, H., Wang, Y., Feng, M., Wang, D., Yin, B., Ruan, X.: Learning to detect salient objects with image-level supervision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
57. Wang, L., Lu, H., Xiang, R., Yang, M.H.: Deep networks for saliency detection via local estimation and global search. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
58. Wang, L., Wang, L., Lu, H., Zhang, P., Ruan, X.: Saliency detection with recurrent fully convolutional networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) European Conference on Computer Vision (ECCV). pp. 825–841 (2016)
59. Wang, T., Borji, A., Zhang, L., Zhang, P., Lu, H.: A stagewise refinement model for detecting salient objects in images. In: IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
60. Wang, T., Zhang, L., Wang, S., Lu, H., Yang, G., Ruan, X., Borji, A.: Detect globally, refine locally: A novel approach to saliency detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
61. Wang, W., Zhao, S., Shen, J., Hoi, S.C.H., Borji, A.: Salient object detection with pyramid attention and salient edges. In: The IEEE Conference on Computer Vision and Pattern Recognition (2019)
62. Wu, R., Feng, M., Guan, W., Wang, D., Lu, H., Ding, E.: A mutual learning method for salient object detection with intertwined multi-supervision. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
63. Wu, T., Tang, S., Zhang, R., Zhang, Y.: Cgnet: A light-weight context guided network for semantic segmentation. arXiv preprint arXiv:1811.08201 (2018)
64. Wu, Z., Su, L., Huang, Q.: Cascaded partial decoder for fast and accurate salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
65. Yan, Q., Xu, L., Shi, J., Jia, J.: Hierarchical saliency detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2013)
66. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3166–3173 (2013)
67. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: European Conference on Computer Vision (ECCV). pp. 325–341 (2018)

68. Zeng, Y., Zhang, P., Zhang, J., Lin, Z., Lu, H.: Towards high-resolution salient object detection. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
69. Zhang, G., Wang, C., Xu, B., Grosse, R.: Three mechanisms of weight decay regularization. In: International Conference on Learning Representations (ICLR) (2019)
70. Zhang, L., Dai, J., Lu, H., He, Y., Wang, G.: A bi-directional message passing model for salient object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
71. Zhang, P., Wang, D., Lu, H., Wang, H., Ruan, X.: Amulet: Aggregating multi-level convolutional features for salient object detection. In: IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
72. Zhang, P., Wang, D., Lu, H., Wang, H., Yin, B.: Learning uncertain convolutional features for accurate saliency detection. In: IEEE International Conference on Computer Vision (ICCV). pp. 212–221. IEEE (2017)
73. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6848–6856 (2018)
74. Zhang, X., Wang, T., Qi, J., Lu, H., Wang, G.: Progressive attention guided recurrent network for salient object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
75. Zhao, J.X., Liu, J.J., Fan, D.P., Cao, Y., Yang, J., Cheng, M.M.: Egnet: Edge guidance network for salient object detection. In: IEEE International Conference on Computer Vision (ICCV) (October 2019)
76. Zhao, K., Gao, S.H., Wang, W., Cheng, M.M.: Optimizing the f-measure for threshold-free salient object detection. In: IEEE International Conference on Computer Vision (ICCV) (October 2019)
77. Zhu, W., Liang, S., Wei, Y., Sun, J.: Saliency optimization from robust background detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2814–2821 (2014)