Look, Perceive and Segment: Finding the Salient Objects in Images via Two-stream Fixation-Semantic CNNs

Xiaowu Chen[1], Anlin Zheng[1], Jia Li[1,2*], Feng Lu[1,2]
[1]State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University
[2]International Research Institute for Multidisciplinary Science, Beihang University

Abstract

Recently, CNN-based models have achieved remarkable success in image-based salient object detection (SOD). In these models, a key issue is to find a proper network architecture that best fits for the task of SOD. Toward this end, this paper proposes two-stream fixation-semantic CNNs, whose architecture is inspired by the fact that salient objects in complex images can be unambiguously annotated by selecting the pre-segmented semantic objects that receive the highest fixation density in eye-tracking experiments. In the two-stream CNNs, a fixation stream is pre-trained on eye-tracking data whose architecture well fits for the task of fixation prediction, and a semantic stream is pre-trained on images with semantic tags that has a proper architecture for semantic perception. By fusing these two streams into an inception-segmentation module and jointly fine-tuning them on images with manually annotated salient objects, the proposed networks show impressive performance in segmenting salient objects. Experimental results show that our approach outperforms 10 state-of-the-art models (5 deep, 5 non-deep) on 4 datasets.

1. Introduction

Salient object detection (SOD) in images and videos is one of the key steps in many vision tasks like robot navigation [5] and object recognition [34]. For image-based SOD, there are two major tasks that need to be addressed, including popping-out salient objects as a whole and suppressing all probable distractors. Actually, the two tasks are somehow complementary that inherently lead to the trade-off between recall and precision in image-based SOD. Considering that salient objects may be sometimes embedded in cluttered background and share some visual attributes with certain distractors, SOD remains a challenging task especially in such complex scenes.

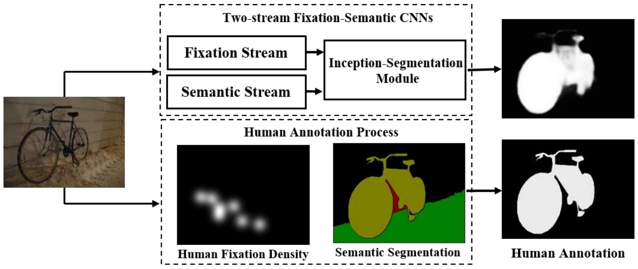

Figure 1. Salient objects can be accurately detected by our two-stream fixation-semantic CNNs, which simulate the process that salient objects are unambiguously annotated by combining human fixation density maps with semantic segmentation results [23].

Towards image-based SOD, hundreds of models have been proposed in the past decade, among which the frameworks gradually evolve from heuristic [1, 4, 33] to learning-based [27, 15, 38]. Compared with the heuristic models, the learning-based models can make better use of complex features and thus often demonstrate better performance. In particular, with the presence of large-scale datasets [28, 44, 20] for image-based SOD, high-dimensional features and complex feature-saliency mapping functions can be directly learned from data by utilizing Convolutional Neural Networks (CNNs). For example, Kuen *et al.* [19] adopted an end-to-end convolution-deconvolution framework to obtain an initial saliency map and then iteratively refined it by recurrent attentional networks. He *et al.* [9] learned heuristic contrast features from two superpixel sequences by using CNNs, and such features were then fused to infer the final saliency map. Lee *et al.* [7] extracted a $26,675$d descriptor for each superpixel and fed it into several cascaded fully-connected (FC) layers so as to identify whether a superpixel is salient or not. With the powerful features and complex mapping functions learned from data, deep models [7, 21, 41, 24, 47, 18] often significantly outperform heuristic models that adopt hand-crafted features (*e.g.*, local/global contrast [26], Gabor response [15] and dissimilarity from image boundary [46]) and classic learning algorithms (*e.g.*, random forest [15], multi-instance learning [42], bootstrap learning [39]).

* Corresponding Author: Jia Li (E-mail: jiali@buaa.edu.cn)

By analyzing the pros and cons of existing deep SOD models, we find that one of the most important issues is to find a proper network architecture that best fits for the SOD task. Toward this end, this paper proposes two-stream fixation-semantic CNNs for image-based SOD (as shown in Fig. 1). The architecture of the proposed networks is mainly inspired by the work of [23], which demonstrates that salient objects can be annotated (and detected) by the human-being (and the classic random forest model) through the fusion of fixation and semantic cues.

In the proposed networks, a fixation stream is pre-trained on eye-tracking data whose architecture is suitable for the task of human fixation estimation. The other stream, denoted as the semantic stream, is pre-trained on image recognition dataset [6] so that it can extract semantic cues from the input visual stimuli. These two streams are then merged into an inception-segmentation module that can detect salient visual content through an inception-like block followed by convolution and deconvolution layers. In this manner, complex salient objects can be detected as a whole, while distractors can be well suppressed (see Fig. 1). Note that the two-stream networks can be directly trained in an end-to-end manner, which avoids the explicit superpixel segmentation process adopted by many existing deep models [21, 20, 9, 7] that often leads to unexpected noise and consumes a large portion of computational resources. Extensive experiments on four benchmark datasets show that the two-stream fixation-semantic networks outperform 10 state-of-the-art deep and non-deep models.

The main contributions of this paper include: 1) We propose novel two-stream fixation-semantic CNNs that can effectively detect salient objects in images; 2) we conduct a comprehensive analysis of state-of-the-art deep SOD models and compare them with the proposed networks, which can be helpful for designing new deep SOD models.

2. Related Work

Hundreds of image-based SOD models have been proposed in the past decade that explore saliency cues such as local/global contrast [17, 4], sparsity and low-rank properties [35, 31, 32] and boundary priors [14, 40]. Recently, Deep Convolutional Neural Networks have been widely used for learning high dimensional representations as well as the complex feature-saliency mapping functions. In many scenarios, such deep models have achieved state-of-the-art performance in salient object detection. Considering that there already exist several comprehensive surveys on non-deep SOD models (*e.g.*, [2, 3]), we only focus on the latest deep models in reviewing related work.

DRR [8] is an early "deep" model that adopts multiple streams of stacked denoising autoencoders to represent multi-scale boundary priors. Salient regions are detected by measuring the reconstruction residuals that reflect the distinctness between background and salient regions.

SuperCNN [9] first segments an image into superpixels at multiple scales. The color uniqueness and distribution sequences extracted from each superpixel are then fed into CNNs to obtain hierarchical features. Finally, multi-scale saliency maps are inferred from such features, which are then fused together to form the final saliency map.

DHSNet [25] has a cascaded architecture that adopts VGG16 [36] to extract a global descriptor for the input image. After that, the descriptor is reshaped to form an initial saliency map, and hierarchical recurrent CNNs are adopted to progressively refine the details in saliency maps so as to highlight the boundaries of salient objects.

SUNet [18] simultaneously carries out the tasks of fixation prediction and SOD within a unified network. The former half of **SUNet** is initialized by the parameters from VGG16, while two branches in the latter half of **SUNet** are separately used for fixation prediction and SOD. Considering that salient objects are tightly correlated with human fixations, the two branches can make full use of two types of training data by enforcing certain weights sharing in **SUNet**, leading to impressive performance.

RACDNN [19] is a recurrent model which initializes a coarse saliency map via a convolution-deconvolution network. After that, the coarse map is refined by iteratively attending to selected image sub-regions and enhancing the detection results. In this manner, the boundaries of salient objects gradually become more clear.

ELD [7] adopts a two-stream architecture, in which the first stream adopts VGG16 to calculate a high dimensional global descriptor for each image, and the second stream divides images into superpixels and characterizes them with heuristic features like color histograms and Gabor responses. After that, the second stream computes several grid-based distance maps and encodes them into compact superpixel descriptors, which are then combined with the global descriptor extracted by the first stream to determine whether a superpixel is salient via `FC` layers.

MDF [20] first divides images into superpixels at 15 scales. After that, single-stream CNNs are adopted for multi-scale feature extraction from the nested and increasingly larger rectangular windows that correspond to a superpixel, its neighboring regions and the whole image (except the segments). Finally, multi-scale features enter `FC` layers for saliency prediction. A salient map can be thus generated by By repeatedly processing every superpixel.

DCL [21] adopts two streams that take the original image and the segmented superpixels as inputs, respectively. In the first stream, VGG16 is revised by incorporating the idea of dilation convolution [45] to generate a coarse saliency map.

In the second stream, images are divided into superpixels, which are then linked with the features extracted in the first stream so that a fine saliency map can be generated by simultaneously measuring the saliency of all superpixels. Finally, the coarse and fine maps are fused via a convolutional layer with 1×1 kernels.

LEGS [24] first estimates pixel-wise saliency for an image by using CNNs. Meanwhile, various object proposals are extracted and incorporated to obtain a local saliency map. After that, another deep networks with only FC layers are adopted to predict the saliency of each candidate object from the global perspective so that salient objects can be detected as a whole.

MCDL [47] adopts a two-stream architecture that embeds superpixels into different contexts. The first stream handles a superpixel-centered window padded with mean pixel value and outputs a global saliency map, which are then fed into the last layer of the second stream that focuses on a closer superpixel-centered window. Finally, saliency map is generated by fusing local and global features.

RFCN [41] proposes convolutional networks with recurrent mechanism for image-based SOD. Heuristic saliency maps are first computed on superpixels by using a contrast-based framework. Such heuristic maps are then used as prior knowledge that enter the recurrent CNNs along with the original image to obtain the refined saliency map. In this process, the foreground map generated by the network is iteratively delivered back to replace the heuristic saliency map so that the quality of a saliency map can be progressively improved. In other words, such recurrent mechanism provides a new way to balance recall and precision.

DSS [10] proposes to use short connections from deeper outputs to shallower ones. In this manner, information from the deeper side can spread out to the shallower side so that salient objects can be accurately detected and refined.

The main characteristics of these deep models can be found in Table 1, from which we find a trend to develop multi-stream end-to-end networks. On the one hand, the multi-stream architecture can learn representative features from multiple perspectives that can be helpful in separating salient objects and distractors. On the other hand, the end-to-end training process avoids the errors at superpixel boundaries caused by inaccurate segmentation algorithms and often leads to faster prediction process.

Beyond existing multi-stream end-to-end networks, an important issue that needs to be further discussed is how to design a better architecture for image-based SOD. In this work, we propose to address this issue by simulating the ways in which ground-truth annotations of salient objects are generated in eye-tracking experiments. Extensive experiments have demonstrated better performance than the 5 deep models we reviewed in this section.

Table 1. A brief summary of state-of-the-art deep models. **Input** (I: image, S: superpixel), **Feature** (H: heuristic, L: learned), **Type** (S: single-stream/cascaded, M: multi-stream, R: recurrent), **Evaluation** (1: MAE, 2: Max F-Measure, 3: Mean F-Measure, 4: Adaptive F-Measure, 5: Precision-Recall Curve)

Model	Input	Feat.	Type	Eval.	#Train
DRR [8]	I	L	M	5	0
ELD [7]	I+S	H+L	M	1+2+5	9,000
SuperCNN [9]	S	L	M	1+4+5	800
LEGS [24]	I+S	L	S	1+3+5	3,340
MCDL [47]	I+S	L	M	4	8,000
MDF [20]	S	L	S	1+4+5	2,500
DCL [21]	I+S	L	M	1+2+4+5	2,500
RFCN [41]	I+S	H+L	S+R	3+5	10,000
SUNet [18]	I	L	M	1+2+3	*25,000
DHSNet [25]	I	L	S+R	4+5	9,500
RACDNN [19]	I	L	S+R	1+2+5	10,565
DSS [10]	I	L	S	1+2+5	2,500

* 10,000 from MSRA10K [28] with salient object masks and 15,000 from SALICON [16] with fixation density maps.

3. The Two-stream Fixation-Semantic CNNs

The architecture of the proposed two-stream fixation-semantic CNNs is shown in Fig. 2, which takes $H \times W$ images as the input and probability maps of the same size as the output. The networks consist of a two-stream module for feature extraction and an inception-segmentation module for feature fusion and saliency estimation.

Two-stream Module. As shown in Fig. 2, the proposed networks start with a two-stream module that consists of two separate streams responsible for the tasks of fixation prediction and semantic perception, respectively. These two streams are initialized by two pre-trained CNNs, including the deep fixation prediction networks (deepFixNet) [30] and the VGG16 networks [36]. In initializing the fixation stream, we first remove the last deconvolution layer of deepFixNet and then revise the kernel size of the last CONV layer from 13×13 to 3×3. The number of kernels in the last CONV layer is also revised so as to output a $\lceil \frac{H}{8}, \frac{W}{8} \rceil$ feature map with 256 channels.

In initializing the semantic stream with the VGG16 networks, we remove all the pooling layers after the CONV3_3 layer and adopt the dilated convolution operator in all subsequent CONV layers so as to maintain the resolution while expanding the receptive field [45]. Similarly, the first two FC layers are also converted into CONV layers with 256 kernels of 7×7 and 1×1, respectively. After that, we obtain a $\lceil \frac{H}{8}, \frac{W}{8} \rceil$ feature map with 256 channels. Finally, the two feature maps from the two-streams are fused via an ELTSUM layer at the end of of the two-stream module to obtain a feature map with 256 channels via element-wise summation, in which both fixation and semantic cues are encoded with equal weights.

Inception-segmentation Module. The fixation and semantic features extracted from the two-stream module are further delivered into the inception-segmentation module,

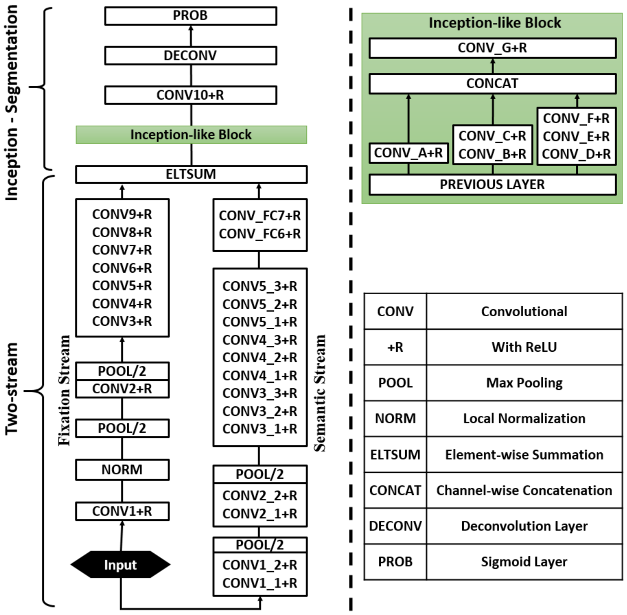

Figure 2. Architecture of the two-stream fixation-semantic CNNs. The bottom half is the two-stream module for extracting fixation and semantic cues, while the top half is the inception-segmentation module for feature fusion and salient prediction.

which contains only a single stream. The design of this module is inspired by the inception networks [37]. As shown in Fig. 2, the inception-segmentation module starts with an inception-like block, in which input features are simultaneously filtered with kernels of different sizes, and the filtered features are then fused in the channel-wise concatenation layer (CONCAT). In this manner, the input data can be processed at multiple scales, leading to a more comprehensive analysis of visual saliency. Finally, a CONV layer with rectified linear unit (ReLU) activation function [29] is added for feature fusion, followed by a DECONV layer and a sigmoid layer cascaded at the end of the inception-segmentation module to output a probability map that represents pixel-wise saliency distribution.

Model Training. The training process of the networks are conducted end-to-end with the cross-entropy loss between estimated and ground-truth saliency maps. The training images are first resized to the resolution of 280 × 280 and flipped horizontally to obtain more training instances. In the optimization process, we set the base learning rate to 5×10^{-8} for the layers initialized with pre-trained networks at the first 40000 iterations and 1×10^{-9} in subsequent iterations. In contrast, the learning rates of the newly added layers are set to 10 times larger. In this manner, the capabilities of deepFixNet and VGG16 in fixation prediction and semantic understanding can be largely preserved, while the newly added layers can be efficiently learned so as to derive a probability map of salient objects from the features extracted from the fixation-semantic streams. The training process is conducted on Caffe [13] with a batch size of 4 images. Moreover, a weight decay of 0.0005 and momentum of 0.9 are used.

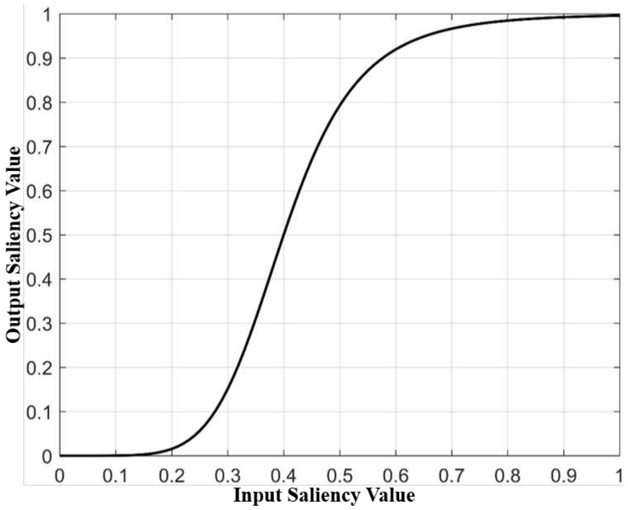

Figure 3. The response of Butterworth high-pass filter with $x_0 = 0.4$ and $M = 3$. This filter can effectively suppress small non-zero responses while preserving the ordering of saliency.

Heuristic Layers for The Testing Phase. After the training process, the end-to-end networks can be directly used to output a saliency map via the last sigmoid layer. However, the characteristics of sigmoid function often lead to small non-zero responses at non-salient regions. Such small responses may make the saliency map somehow 'noisy'. To suppress such small responses and obtain a more clear saliency map, we cascade a heuristic layer at the end of the networks that become activated only in the testing phase. The heuristic layer, denoted as the Butterworth layer, first normalizes the map to the range of [0,1] and delivers the saliency values into a Butterworth high-pass filter that perfectly rejects small saliency values and has nearly uniform sensitivity for the high saliency values. The Butterworth high-pass filter $\mathcal{B}(x)$ is defined as

$$\mathcal{B}(x) = 1 - \frac{1}{1 + \left(\frac{x}{x_0}\right)^{2M}}, \tag{1}$$

where x_0 is the 'cutoff frequency' and M is the order of filter. In this study, we empirically set $x_0 = 0.4$ and $M = 3$ to prune saliency values smaller than 0.1 (see Fig. 3 for the curve of the Butterworth filter). We can see that saliency values smaller than 0.1 are almost pruned, while the ordering of the rest saliency values stay unchanged since $\mathcal{B}(x)$ is monotonically increasing in $[0, 1]$.

4. Experiments

4.1. Experimental Settings

To validate the effectiveness of the proposed approach, we conduct massive experiments on 4 datasets that are widely used in the literature, including:

1) DUT-OMRON [44] contains $5,168$ complex images with pixel-wise annotations of salient objects. All images are down-sampled to a maximal side length of 400 pixels.

2) PASCAL-S [23] contains 850 natural images that are pre-segmented into objects/regions and free-viewed by 8 subjects in eye-tracking tests for salient object annotation.

3) ECSSD [43] contains $1,000$ images with complex structures and obvious semantically meaningful objects.

4) HKU-IS [20] consists of $4,447$ images. Many images contain multiple disconnected salient objects or salient objects that touch image boundaries.

On these datasets, we compare the two-stream fixation-semantic CNNs (denoted as **FSN**) with 10 models:

- **Deep models**, include: **LEGS** [24], **ELD** [7], **M-CDL** [47], **MDF** [20] and **DCL** [21].
- **Non-deep models**, include **SMD** [32], **DRFI** [15], **RBD** [48], **MST** [40] and **MB+** [46].

All models have source codes on the Internet. As shown in Table 1, most deep models are trained on **MSRA-B** [27] or **MSRA10K** [28, 4]. In this study, we have our **FSN** model trained on the $10,000$ images from **MSRA10K** too. Note that the model **LEGS** also incorporates 340 images from **PASCAL-S** in their training set, which are ignored in the testing stage. Moreover, complex post-processing steps in deep models (*e.g.*, the CRF-based post-processing in **DCL**) are not used for fair comparisons.

In the comparisons, we refer to two evaluation metrics (codes implemented by [22]), including $\mathbf{F}_\beta$ and the Mean Absolute Error (**MAE**). **MAE** reflects the average pixel-wise absolute difference between the estimated and ground-truth saliency maps that are both normalized to $[0, 1]$. Note that we first compute a **MAE** score for each image and then average such scores over a whole dataset. In computing $\mathbf{F}_\beta$, we normalize the estimated saliency maps into $[0, 255]$ and simultaneously binarize all saliency maps from the same dataset by enumerating all probable thresholds in $\{0, \ldots, 255\}$. At each threshold, a Recall is computed as TP/(TP+FN) and a Precision is computed as TP/(TP+FP) for each frame, where TP, FN and FP are the numbers of true positives, false negatives and false positives, respectively. Finally, the $\mathbf{F}_\beta$ is computed as

$$\mathbf{F}_\beta = \frac{(1+\beta^2)\cdot \text{Precision}\cdot\text{Recall}}{\beta^2\cdot\text{Precision}+\text{Recall}}, \qquad (2)$$

where we set $\beta^2 = 0.3$ as in many previous works to emphasize more on Precision. After that, the $\mathbf{F}_\beta^{\max}$ curves can be drawn to show the performance of a model in using different thresholds for binarizing saliency maps, and the maximal $\mathbf{F}_\beta$ over the curve, denoted as $\mathbf{F}_\beta^{\max}$, is used to represent the overall performance of a model. Different from the adaptive $\mathbf{F}_\beta$ that binarizes saliency maps with adaptive thresholds (*e.g.*, twice the mean saliency value as in [1]), $\mathbf{F}_\beta^{\max}$ is less sensitive to reparameterization operations that are frequently used in the post-processing steps, which can lead to fairer comparisons.

4.2. Model Comparisons

The performance of **FSN** and the other 10 models over the four datasets are reported in Table 2. The $\mathbf{F}_\beta^{\max}$ curves of these models are shown in Fig. 4. In addition, some representative examples are shown in Fig. 5.

Comparisons between FSN and other models. From Table 2 and Fig. 4, we can see that **FSN** achieves the best performance over all the four datasets, including the highest $\mathbf{F}_\beta^{\max}$ and the lowest **MAE**. Its maximal improvement against other 10 models varies between 7.1% and 28.0%.

In particular, we find that **FSN** outperforms all the other 5 deep models on the four datasets with the maximal improvements over $\mathbf{F}_\beta^{\max}$ range from 7.1% to 16.7%. By inspecting the characteristics of existing deep models shown in Table 1, we find that such improvements may be mainly caused by the two-stream architecture of **FSN** that fuses both fixation and semantic streams without segmenting any superpixels. Actually, the superpixel segmentation will inevitably bring in certain kinds of noise and inherently set a performance upper-bound for the SOD model. Even though the influence of inaccurate segmentation can be alleviated by multi-scale segmentation (*e.g.*, 15 scales in **MDF**), it is still a big challenge to simultaneously select the segments with the best quality and assign the correct saliency scores to them. In contrast, the end-to-end training scheme of fixation-semantic networks avoids the errors introduced by such heuristic pre-processing operations and its performance is solely determined by the data and the network architecture. As a result, pixels within the same semantic category are actually assigned with similar feature descriptors in the semantic stream (although such assignments are not obviously conducted). Meanwhile, the fixation stream inherently assigns each pixel a descriptor that is tightly correlated with the fixation density the pixel may receive. In this manner, the boundaries of salient objects can be still accurately localized without any direct segmentation of superpixels (see the examples in Fig. 5).

One more thing that worth mentioning is that **FSN** takes only 0.12s per image on the Matlab platform with a 3.4GHz CPU (single thread) and a NVIDIA GTX 1080 GPU (LEGS: 1.6s, MDF: 8.0s, MCDL: 2.4s, ELD: 0.73s,

Table 2. Performance of **FSN** and 10 state-of-the-arts on four datasets. Larger $\mathbf{F}_\beta^{\max}$ and smaller **MAE** correspond to better performance. Max ↑ (%) means the maximal relative improvement of **FSN** against other models over the four datasets in $\mathbf{F}_\beta^{\max}$.

	Models	DUT-OMRON		PASCAL-S		ECSSD		HKU-IS		Max ↑ (%)
		$\mathbf{F}_\beta^{\max}$	**MAE**	$\mathbf{F}_\beta^{\max}$	**MAE**	$\mathbf{F}_\beta^{\max}$	**MAE**	$\mathbf{F}_\beta^{\max}$	**MAE**	
Non-Deep	**SMD** [32]	.624	.166	.690	.201	.760	.173	.743	.156	23.1
	DRFI [15]	.664	.150	.694	.201	.782	.170	.777	.145	19.2
	RBD [48]	.630	.144	.659	.197	.716	.171	.723	.142	27.1
	MST [40]	.600	.149	.670	.187	.731	.149	.722	.128	28.0
	MB+ [46]	.624	.168	.680	.193	.739	.171	.728	.150	23.1
Deep	**DCL** [21]	.717	.094	.808	.110	.887	.072	.880	.058	7.10
	LEGS [24]	.670	.204	.759	.155	.827	.118	.767	.192	16.7
	ELD [7]	.700	.092	.771	.126	.866	.079	.839	.073	9.70
	MCDL [47]	.702	.088	.743	.146	.837	.100	.808	.091	11.3
	MDF [20]	.694	.092	.768	.150	.832	.105	.814	.112	10.7
	FSN	**.768**	**.065**	**.827**	**.095**	**.910**	**.053**	**.895**	**.044**	-

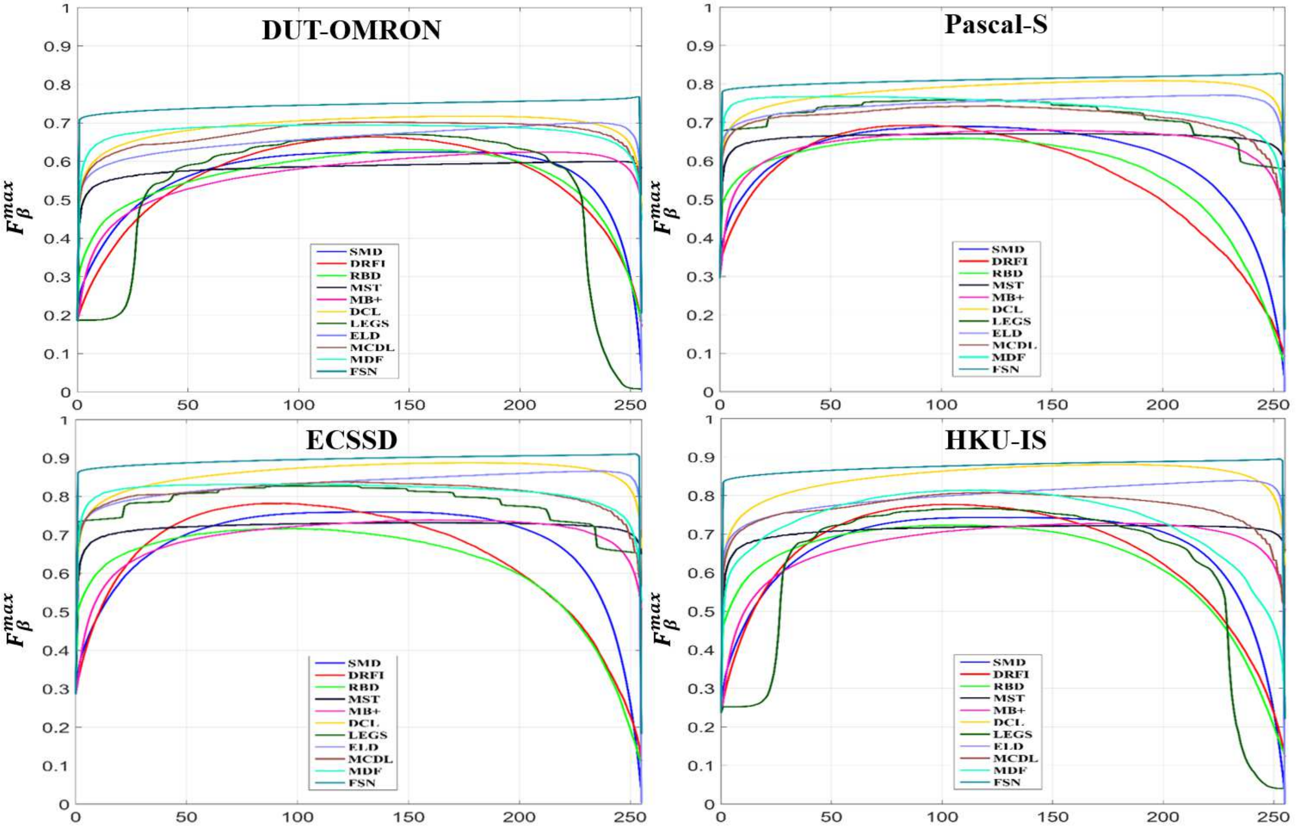

Figure 4. The $\mathbf{F}_\beta^{\max}$ curves of **FSN** and 10 state-of-the-art deep models on four datasets.

DCL: 0.17s). This may be caused by the revision of VGG16 networks that greatly reduces the amount of parameters.

Comparisons between deep and non-deep models. By comparing the deep models and non-deep models, we also find that the 6 deep models outperform the 5 non-deep models in most cases. Actually, deep and non-deep models now start to focus on different aspects of image-based SOD. For deep models, recall and precision are still the major objectives, while speed and complexity are somehow beyond the main concerns. Instead, computational cost is frequently emphasized by recent non-deep models (*e.g.*, **MST**: ∼40 FPS, **MB+**: ∼60 FPS), even though their recall and precision are often worse than deep models. With these characteristics, deep models become suitable for off-line processing of large-scale image data, while non-deep models can be well utilized in online applications or on

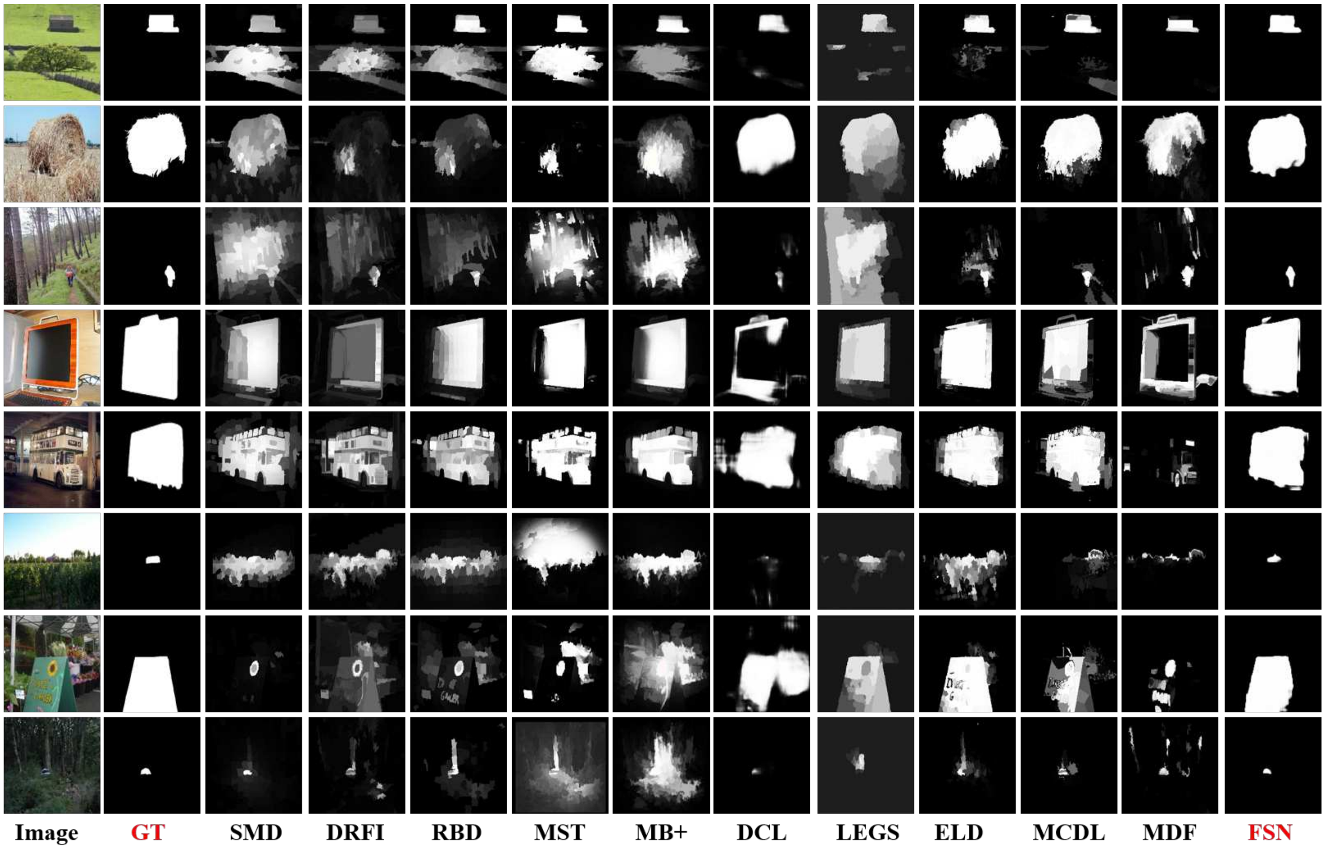

Figure 5. Representative results of **FSN** and the other 10 models. **GT** indicates ground-truth mask of salient objects.

mobile devices that require small models, high processing speed and acceptable recall and precision.

Comparisons between different settings of FSN. To validate the effectiveness of different components of **FSN**, we conduct several experiments on **DUT-OMRON**, the largest one among the four datasets with many complex scenarios, to see the performance variation of **FSN** with different experimental settings.

(1) Without fixation stream. The VGG16 network has been used in many deep SOD models [7, 21], while the usage of fixation stream is relatively new. To validate that the fixation stream is useful, we remove the fixation stream and re-train the whole networks. In this case, the $\mathbf{F}_{\beta}^{\max}$ score decreases from 0.768 to 0.753, and the **MAE** score increases from 0.065 to 0.080. This indicates that the fixation stream can facilitate the detection of salient objects.

(2) Null fixation stream. To further validate the influence of fixation stream, we set its output feature maps to all zeros and re-test the two-stream **FSN**. In this case, the $\mathbf{F}_{\beta}^{\max}$ score drops sharply (see Table 3), implying that the fixation stream can provide useful cues to detect salient objects.

(3) Randomly initialized fixation stream. Another concern is that the performance gain of **FSN** may come from the additional eye-tracking data used to pre-train the fixation branch. Therefore, we only keep the architecture of the fixation stream and re-initialize all its parameters with random values. Surprisingly, as shown in Table 3, the performance scores only slightly decrease, indicating that it is the architecture, other than the additional training data, that makes real contribution to the SOD task.

Table 3. Performance of **FSN** on **DUT-OMRON** with new settings

Settings	$\mathbf{F}_{\beta}^{\max}$	**MAE**
Without fixation stream	.753	.080
Null fixation stream	.723	.072
Randomly initialized fixation stream	.759	.074
Without inception-like block	.752	.068
Without Butterworth	.773	.101
Default Setting	.768	.065

(4) Without inception-like block. In this experiment, we remove the inception-like block and retrain the networks. In this case, salient objects are simply detected and segmented via `CONV` and `DECONV` layers. As shown in Table 3, the $\mathbf{F}_{\beta}^{\max}$ score decreases from 0.768 to 0.752. This may be caused by the fact that the inception-like block starts with

three parallel branches with different CONV layers, making it capable to explore the saliency cues from multiple scales.

(5) Without Butterworth layer. As shown in Table 3, the $\mathbf{F}_\beta^{\max}$ score slightly increases (0.65%) when the Butterworth layer is discarded, while there exists a sharp increase (55.4%) of **MAE**. This may be caused by the fact that the normalization and high-pass filtering operations in the Butterworth layer have little impact on the ordering of saliency. In this manner, the $\mathbf{F}_\beta^{\max}$ obtained by enumerating all probable thresholds stay almost unchanged. In contrast, the **MAE** metric focuses on the magnitude of saliency and thus become very sensitive to such re-parameterization operations. Actually, we observe similar performance variations by applying Butterworth filter to the results of other models. For example, after using the same Butterworth filter the $\mathbf{F}_\beta^{\max}$ scores of LEGS and MDF stay almost unchanged, while their **MAE** scores decrease to 0.129 and 0.087, respectively. These results further validate the effectiveness of selecting $\mathbf{F}_\beta^{\max}$ as the main evaluation metric other than the adaptive $\mathbf{F}_\beta$ and mean $\mathbf{F}_\beta$ as in many previous works.

Drawback of deep models. As shown in Table 2, the $\mathbf{F}_\beta^{\max}$ scores of the six deep models reach above 0.8 on at least one dataset. In particular, on **ECSSD**, a dataset that used to be widely recognized as very challenging for containing scenes with complex structures, existing deep models significantly outperform non-deep ones in $\mathbf{F}_\beta^{\max}$ since they can learn more effective representations. However, such successful cases do not mean that deep models already capture all the essential characteristics of salient objects in all scenes.

To validate this point, we test the six deep models over the psychological patterns, in which salient objects are very simple and can be easily detected by the human-being and the fixation prediction algorithms proposed decades ago (*e.g.*, [12, 11]). As shown in Fig. 6, however, the six deep models often fail to detect such simple psychological patterns, which may be caused by the fact that existing deep models rely heavily on the features learned from natural images, in which salient objects often have obvious semantic meanings. However, such features may not always work well in processing the simple psychological patterns without obvious semantic attributes. In these cases, simple features like local/global contrasts, responses of orientated Gabor filters and spatial color distributions may work, which may imply that incorporating such heuristic features may be a way for designing better deep models.

5. Conclusion

This paper proposes two-stream fixation-semantic CNNs for image-based salient object detection. When the fixation stream corresponds to human visual attention, the semantic stream extracts features for high level visual perception. These two streams are then fused into the inception-segmentation module in which salient objects can be efficiently and accurately segmented. Experimental results show that the proposed fixation-semantic networks outperform 5 deep and 5 non-deep models on four datasets, which further validates the feasibility of designing network architecture by simulating the ways that ground-truth data are generated by the human-being.

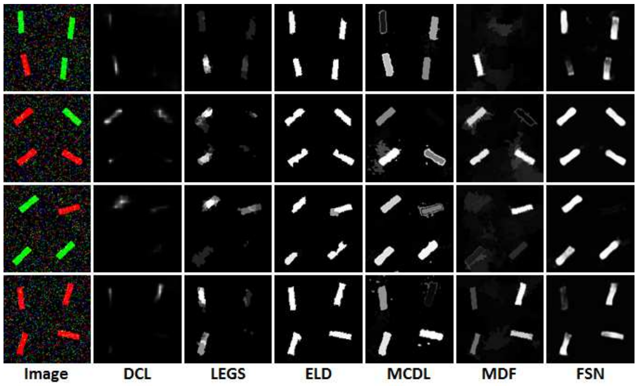

Figure 6. The six deep models fail in many cases when detecting salient psychological patterns.

Despite the impressive success of the proposed networks over the natural images from existing datasets, the failure cases on simple psychological patterns indicate that there is still a long way to go before the perfect detection of salient objects in various types of scenarios. In the future work, we will try to incorporate the heuristic features like local/global contrast into our model and re-design its architecture to simulate the saccade shift processes of the human-being in eye-tracking experiments so as to detect salient objects beyond natural images.

Acknowledgement. This work was partially supported by National Natural Science Foundation of China (61672072, 61532003, 61325011 and 61421003), and Fundamental Research Funds for the Central Universities.

References

[1] R. Achanta, S. Hemami, F. Estrada, and S. Süsstrunk. Frequency-tuned salient region detection. In *CVPR*, 2009.

[2] A. Borji, M.-M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. *IEEE TIP*, 24(12):5706–5722, 2015.

[3] A. Borji, D. N. Sihite, and L. Itti. Salient object detection: A benchmark. In *ECCV*, 2012.

[4] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu. Global contrast based salient region detection. *IEEE TPAMI*, 37(3):569–582, 2015.

[5] C. Craye, D. Filliat, and J.-F. Goudou. Environment exploration for object-based visual saliency learning. In *ICRA*, 2016.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] L. Gayoung, T. Yu-Wing, and K. Junmo. Deep saliency with encoded low level distance map and high level features. In *CVPR*, 2016.

[8] J. Han, D. Zhang, X. Hu, L. Guo, J. Ren, and F. Wu. Background prior-based salient object detection via deep reconstruction residual. *IEEE TCSVT*, 25(8):1309–1321, 2015.

[9] S. He, R. Lau, W. Liu, Z. Huang, and Q. Yang. Supercnn: A superpixelwise convolutional neural network for salient object detection. *IJCV*, 115(3):330–344, 2015.

[10] Q. Hou, M.-M. Cheng, X.-W. Hu, A. Borji, Z. Tu, and P. Torr. Deeply supervised salient object detection with short connection. In *CVPR*, 2017.

[11] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *CVPR*, 2007.

[12] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE TPAMI*, 20(11):1254–1259, 1998.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[14] B. Jiang, L. Zhang, H. Lu, and M. Yang. Saliency detection via absorbing markov chain. In *ICCV*, 2013.

[15] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li. Salient object detection: A discriminative regional feature integration approach. In *CVPR*, 2013.

[16] M. Jiang, S. Huang, J. Duan, and Q. Zhao. Salicon: Saliency in context. In *CVPR*, 2015.

[17] D. A. Klein and S. Frintrop. Center-surround divergence of feature statistics for salient object detection. In *ICCV*, 2011.

[18] S. S. S. Kruthiventi, V. Gudisa, J. H. Dholakiya, and R. V. Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *CVPR*, 2016.

[19] J. Kuen, Z. Wang, and G. Wang. Recurrent attentional networks for saliency detection. In *CVPR*, 2016.

[20] G. Li and Y. Yu. Visual saliency based on multiscale deep features. In *CVPR*, 2015.

[21] G. Li and Y. Yu. Deep contrast learning for salient object detection. In *CVPR*, 2016.

[22] X. Li, Y. Li, C. Shen, A. Dick, and A. V. D. Hengel. Contextual hypergraph modeling for salient object detection. In *ICCV*, 2013.

[23] Y. Li, X. Hou, and C. Koch. The secrets of salient object segmentation. In *CVPR*, 2014.

[24] X. R. Lijun Wang, Huchuan Lu and M.-H. Yang. Deep networks for saliency detection via local estimation and global search. In *CVPR*, 2015.

[25] N. Liu and J. Han. DHSNet: Deep hierarchical saliency network for salient object detection. In *CVPR*, 2016.

[26] T. Liu, J. Sun, N.-N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. In *CVPR*, 2007.

[27] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *IEEE TPAMI*, 33(2):353–367, 2011.

[28] MSRA10K. http://mmcheng.net/gsal/.

[29] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[30] J. Pan, E. Sayrol, X. Giro-i Nieto, K. McGuinness, and N. E. O'Connor. Shallow and deep convolutional networks for saliency prediction. In *CVPR*, 2016.

[31] H. Peng, B. Li, R. Ji, W. Hu, W. Xiong, and C. Lang. Salient object detection via low-rank and structured sparse matrix decomposition. In *AAAI*, 2013.

[32] H. Peng, B. Li, H. Ling, W. Hu, W. Xiong, and S. J. Maybank. Salient object detection via structured matrix decomposition. *IEEE TPAMI*, 39(4):818–832, 2017.

[33] F. Perazzi, P. Krahenbuhl, Y. Pritch, and A. Hornung. Saliency filters: Contrast based filtering for salient region detection. In *CVPR*, 2012.

[34] Z. Ren, S. Gao, L. T. Chia, and I. W. H. Tsang. Region-based saliency detection and its application in object recognition. *IEEE TCSVT*, 24(5):769–779, 2014.

[35] X. Shen and Y. Wu. A unified approach to salient object detection via low rank matrix recovery. In *CVPR*, 2012.

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arxiv:1409.1556*, 2014.

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[38] Y. Tian, J. Li, S. Yu, and T. Huang. Learning complementary saliency priors for foreground object segmentation in complex scenes. *IJCV*, 111(2):153–170, 2015.

[39] N. Tong, H. Lu, and M. Yang. Salient object detection via bootstrap learning. In *CVPR*, 2015.

[40] W.-C. Tu, S. He, Q. Yang, and S.-Y. Chien. Real-time salient object detection with a minimum spanning tree. In *CVPR*, 2016.

[41] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan. Saliency detection with recurrent fully convolutional networks. In *ECCV*, 2016.

[42] Q. Wang, Y. Yuan, P. Yan, and X. Li. Saliency detection by multiple-instance learning. *IEEE Transactions on Cybernetics*, 43(2):660–672, 2013.

[43] Q. Yan, L. Xu, J. Shi, and J. Jia. Hierarchical saliency detection. In *CVPR*, 2013.

[44] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, 2013.

[45] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[46] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Měch. Minimum barrier salient object detection at 80 fps. In *ICCV*, 2015.

[47] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015.

[48] W. Zhu, S. Liang, Y. Wei, and J. Sun. Saliency optimization from robust background detection. In *CVPR*, 2014.