Deep Salient Object Detection with Fuzzy Superpixel Extraction and Controlled Filter Convolution

Yang Liu*, Bo Wu† and Bo Lang‡
State Key Laboratory of Software Development Environment
Beihang University, No.37 Xueyuan Rd., Beijing, China
Email: {*blonster, †wubo, ‡langbo}@nlsde.buaa.edu.cn

Abstract—**Deep salient object detection (DSOD), which leverages the popular deep learning techniques, is a promising new branch of salient object detection (SOD). By training on large-scale public datasets, DSOD methods showed significant performance improvement while avoiding the involvement of manually designed visual features and prior knowledge of specific datasets. This paper proposes a novel superpixel-based DSOD method based on fuzzy superpixel extraction (FSE), a neural network-based differentiable superpixel extraction method, and controlled filter convolution (CFC), a modified convolution operation that accepts two input feature maps and can balance their influences without hand-picked coefficients. Different from other superpixel-based methods, by using FSE, the proposed method is able to include superpixel extraction in the training process, which optimizes the superpixel representations according to the datasets. Then, the CFC layers combine two different parts of the information possessed by the superpixels, which are intrasuperpixel features and intersuperpixel features, to generate a unified feature map. In the experiments conducted on 5 widely used public datasets, the proposed method significantly outperformed state-of-the-art models, which proved its effectiveness and generalization ability.**

I. Introduction

How to model the pattern of human attention when seeing a picture has been a long-standing and challenging topic of computer vision (CV). Since salient objects are supposed to be more important or interesting in images, correctly recognizing these objects, which is usually called salient object detection (SOD), can provide useful information to several more complicated visual tasks, such as object recognition, semantic segmentation and other visual pattern recognition tasks. By using SOD as a preprocessing step, algorithms for these tasks can potentially improve in both accuracy and efficiency.

Based on empirical assumptions or prior knowledge of the neural processes of human brains, researchers have developed a series of models [1], [2], [3], [4], [5], [6], [7] to predict human eye fixation on each pixel. These models work in an unsupervised or semisupervised manner, thus highly depending on their basic assumptions, which are mostly derived from intuitive conclusions or empirical knowledge. Therefore, the above methods are usually very unstable among different datasets, and their performances are relatively low when compared to recent methods.

There are also some researchers [8], [9] that have approached the problem of SOD with supervised learning techniques. These methods introduced the idea of basing models on actual data rather than assumptions, and they achieved impressive results compared to other methods proposed at the same time. Meanwhile, these methods are mostly based on superpixel extraction or other oversegmentation methods, and *their success proved the potential of superpixel-based SOD*.

Superpixel extraction is a branch of image segmentation that strictly localizes the influence of each pixel to reduce time consumption. Different from global segmentation methods such as local variation segmentation [10], superpixel extraction methods tend to overly segment images into many very small regions (i.e., superpixels). While ignoring some pixel-level texture information, superpixels usually maintain most region boundaries, which are especially important for segmentation tasks, including saliency detection. Meanwhile, because there are considerably fewer superpixels than pixels, working on the superpixel-level can greatly reduce the time consumption of an algorithm. Therefore, *superpixel extraction can become an effective component of SOD methods*.

In the last few years, deep learning has drawn considerable attention in the field of machine learning. Because neural networks are capable of describing data distributions by using a large number of sophisticated parameters, they have proven to be extremely effective in many CV tasks. Naturally, deep learning techniques have also been introduced in studies of SOD, thus creating a new branch called deep salient object detection (DSOD). Most DSOD studies [11], [12], [13], [14], [15], [16] have been based on common deep learning techniques widely used in other CV studies, but *they still achieved significantly better performances than traditional methods, which proved that DSOD is very promising in SOD research*.

Different from pixel-level DSOD methods, some researchers have employed superpixel extraction techniques as a part of their studies. SuperCNN [17] leverages superpixelwise convolution and hierarchical contrast features to obtain saliency maps of multiple scales. Then, the multiscale saliency maps are fused with learned weights to yield the final pixel-level saliency map. Tianshui Chen et al. [18] proposed an end-to-end framework called deep image saliency computing (DISC). DISC formulates the DSOD problem as a progressive representation learning problem. It uses superpixels for coarse-level saliency evaluation and then combines coarse-level and fine-level saliency maps with superpixel-based local context information (SLCI). *The success of SuperCNN and DISC proved that superpixel extraction can also be an effective component of DSOD methods.*

However, traditional superpixel extraction methods such as SLIC [19] are mostly nondifferentiable; thus, they cannot be part of the training processes, e.g., SuperCNN and DISC only use superpixel segmentation as input and discard derivatives of superpixels in the backpropagation process.

We consider this a problem because while the whole network is being optimized according to a specific dataset, superpixel extraction cannot gain from the training process and could potentially become a bottleneck in the performance. Therefore, we propose a novel differentiable superpixel extraction method implemented by neural network modules, i.e., fuzzy superpixel extraction (FSE). Fuzzy superpixel extraction networks (FSENs) are basically CNNs with the dimensions of their outputs being fixed to $O \times N \times M$, where O is the upper bound of the number of potential owner superpixels with respect to each pixel, and N and M are the height and width of the input image, respectively. Their outputs represent values of the membership function of each superpixel at each pixel. Sec.III explains FSEN and the constant O. FSENs are CNNs, which means they can be embedded into any network.

The other problem in superpixel-based methods is how to balance intrasuperpixel information and intersuperpixel information, more specifically, information of pixels within a superpixel and information of the boundary around a superpixel. Unlike pixels that share a fixed shape in an image, each superpixel has a unique shape that could contain information useful to segmentation tasks; thus, treating superpixels as mere collections of pixels, such as averaging color and features among pixels, may lose important information.

As stated before, SuperCNN [17] directly conducts convolutions on superpixels, which almost completely ignores intersuperpixel information. DISC [18] uses SLCI to combine the two types of information, but SLCI is basically hand designed and heavily dependent on assumptions. Therefore, we propose using a neural network-based approach, called controlled filter convolution (CFC), with its own learnable parameters to reduce the influence of priors and assumptions.

CFC is inspired by dynamic filter convolution (DFC) [20] and edge-conditioned convolution (ECC) [21]. DFC and ECC are based on the idea of dynamically changing the convolution kernels according to the current input. The idea is based to a great extent on convolution-based neural networks, but since DFC and ECC are both designed to process graph data, which are highly unstructured and unordered, they are too complicated for image processing. Therefore, we propose CFC as a task-specific simplified method. Different from traditional convolution layers, each CFC layer simultaneously accepts two input feature maps with different channels, and the output is not simply a weighted sum of the two inputs but generated in a slightly complicated fashion. Specifically, when using one of the input feature maps as the input of a traditional convolution layer, the filters are functions of the other input feature map, i.e., the filters are *CONTROLLED* by one of the input feature maps. In our experiments, superpixelwise features and superpixel shape description features are the two inputs to CFC, so that intrasuperpixel information and intersuperpixel information can participate in the following process in a balanced manner.

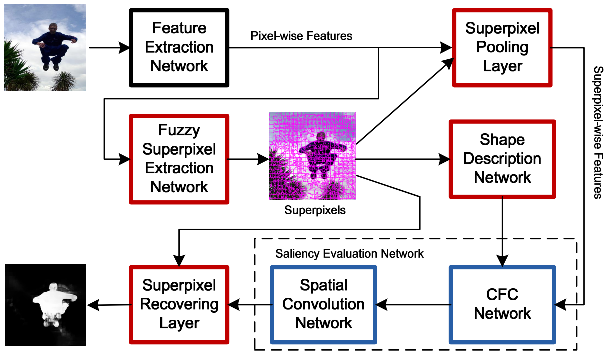

Feature Extraction Network
Pixel-wise Features
Superpixel Pooling Layer
Superpixel-wise Features
Fuzzy Superpixel Extraction Network
Superpixels
Shape Description Network
Saliency Evaluation Network
Superpixel Recovering Layer
Spatial Convolution Network
CFC Network

Fig. 1. The salient object detection network using fuzzy superpixel extraction and controlled filter convolution. The black box marks the feature extraction network, the red boxes mark the superpixel handling modules, and the blue boxes mark the components of the saliency evaluation network.

Fig. 1 illustrates our working network. This network is a complete end-to-end solution for the pixelwise saliency detection task, which is different from DISC in that it needs to use two different networks simultaneously. Apparently, while both take advantage of superpixels, a single network is much easier to train.

The main contributions of this paper lie in three parts:

- Fuzzy superpixel extraction (FSE), a neural network-based differentiable superpixel extraction method that can participate in the training process and be optimized according to the datasets;
- Controlled filter convolution (CFC), a convolution-based network layer model that simultaneously accepts two different input feature maps and is capable of balancing their influences without hand-picked coefficients; and
- a novel DSOD network consisting of FSE and CFC.

The remainder of this paper is structured as follows. Sec. II will introduce the task of deep salient object detection and the proposed network to clarify some important concepts and give the readers an overall understanding of our method. Then, fuzzy superpixel extraction and controlled filter convolution are presented in detail in Sec. III and Sec. IV, respectively. To evaluate the performance, a series of experiments have been conducted, and their results will be presented and discussed in Sec. V. Finally, Sec. VI concludes our work.

II. Deep Salient Object Detection Framework

Experimentally, the task of SOD is to assign a saliency score to each pixel in an input image and to make the score as close to a groundtruth reference value as possible, i.e., to find a function $f : \mathbb{R}^{C\times|X|} \to \mathbb{R}^{|X|}$, which minimizes $\|f(I) - G_t\|$, where C is the number of color channels, X is the set of coordinates within the image, I is the input image, and G_t is the groundtruth saliency map.

Different from hand-designed algorithms, deep salient object detection (DSOD) methods use deep learning techniques to model function f. In contrast to traditional algorithms that are basically based on assumptions, DSOD methods are capable of constructing f according to the target dataset. Therefore, by training the networks on large datasets, DSOD methods significantly outperform the traditional methods. In this paper, the proposed network is a single end-to-end network illustrated in Fig. 1, which consists of three major parts: a) a **feature extraction network** (FEN) that provides the following processes with a pixelwise feature map; b) superpixel handling modules, including a **fuzzy superpixel extraction network** (FSEN), a **superpixel pooling layer** (SPL), a **superpixel recovering layer** (SRL) and a **shape description network** (SDN); and c) the **saliency evaluation network** (SEN), which determines the saliency score of each superpixel. If the whole network is seen as a single function f, each module of the network can also be considered as a function:

$$\begin{aligned} FEN &: \mathbb{R}^{C\times|X|} \rightarrow \mathbb{R}^{D_F\times|X|} && (1)\\ FSEN &: \mathbb{R}^{D_F\times|X|} \rightarrow \mathbb{R}^{O\times|X|} && (2)\\ SDN &: \mathbb{R}^{O\times|X|} \rightarrow \mathbb{R}^{D_S\times K} && (3)\\ SPL &: \mathbb{R}^{D_F\times|X|} \times \mathbb{R}^{O\times|X|} \rightarrow \mathbb{R}^{D_F\times K} && (4)\\ SEN &: \mathbb{R}^{D_F\times K} \times \mathbb{R}^{D_S\times K} \rightarrow \mathbb{R}^{K} && (5)\\ SRL &: \mathbb{R}^{K} \times \mathbb{R}^{O\times|X|} \rightarrow \mathbb{R}^{|X|}, && (6) \end{aligned}$$

where D_F is the dimension of pixelwise features given by the FEN, D_S is the dimension of the shape description features given by the SDN, and K is the number of superpixels.

Thus, the workflow can be described as follows:

1) $F_p = FEN(I)$, where F_p is the pixelwise feature map;
2) $S = FSEN(F_p)$, where S is the superpixel representation of the original image;
3) $F_{ssd} = SDN(S)$, where F_{ssd} is the superpixel shape description feature map;
4) $F_s = SPL(F_p, S)$, where F_s is the superpixelwise feature map;
5) $Y_s = SEN(F_s, F_{ssd})$, where Y_s is the superpixelwise saliency map; and
6) $Y = SRL(Y_s, S)$, where Y is the final saliency map.

First, FEN generates a pixelwise feature map F_p according to the input image I. Then, FSEN, SDN and SPL reduce the feature map from pixel-level F_p to superpixel-level F_s and provide superpixel-specific feature F_{ssd}. Then, SEN can generate a saliency map Y_s according to F_s and F_{ssd}. However, since SEN works at the superpixel level, Y_s is superpixelwise, which is not the intended output of DSOD. Therefore, an SRL is needed in the final stage to recover the pixelwise information with the help of superpixel representation S.

During training, there should be a **loss function** attached to the end of the network to compare output Y with ground truth G_t. In our experiments, the loss function was chosen as a *smoothed mean absolute error* used in fast R-CNN [22]. For convenience, the output of the loss function, i.e., the error or loss, is noted as E in the remainder of the paper.

As stated above, the feature map given by FEN is shared by the FSEN and the SEN, thus forming the foundation of all the following processes. This feature map heavily affects the performance of the entire network, especially the capability of generalization. Therefore, due to the limited sizes of the SOD datasets, in the experiments discussed in Sec. V, we did not train FEN along with the other components but chose to use the first two convolution layers of a VGG16 network [23] pretrained on ImageNet[1].

The superpixel representation is generated by the FSEN according to the pixelwise feature map given by the FEN, and serves as input to the SDN, SPL and SRL to help in generating the superpixel shape description features, average pooling pixelwise features within the superpixel boundaries and recovering the pixelwise saliency values from the superpixelwise saliency map. It involves most of the major components within the network, which makes it an extremely important part of our method. Therefore, the extraction and use of fuzzy superpixels are explained in detail in Sec. III.

Because the input of the SEN consists of a superpixelwise feature map and a superpixel shape description feature map, and these two feature maps cannot be compared numerically, it could prove difficult to balance their influence by handpicked coefficients. To solve this problem, we propose the use of CFC instead of conventional convolution layers in this part of the network. CFC guarantees the equality between the two inputs with no handpicked coefficients involved, and they have proven effective in our experiments in Sec. V-D. To obtain a better understanding of CFC, please refer to Sec. IV.

III. Fuzzy Superpixel Extraction

Fuzzy superpixel extraction (FSE) is inspired by SLIC [19]. As the name suggests, FSE considers superpixels as *FUZZY* sets of pixels instead of sets in traditional methods and thus converts the superpixel extraction into the problem of assigning values to the membership functions of the superpixels. By implementing FSE with neural network modules, FSE is differentiable and can be easily embedded into any network.

Assuming there are a fixed number of superpixels, and each of them has a limited pixel pool (the set of candidate pixels), FSE can be seen as a function $FSE : \mathbb{R}^{C\times|X|} \rightarrow [0,1]^{K\times|X|}$, for $\forall I$ and $S = FSE(I)$ s.t.

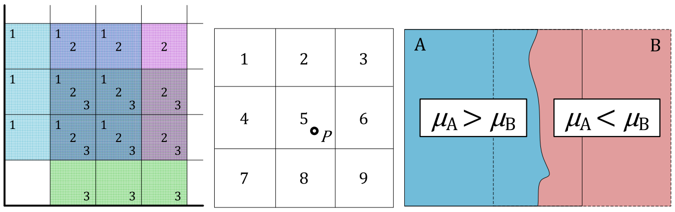
$$\begin{cases} \sum_{k=1}^{K} S_{k,x_0} = 1 & \forall x_0 \in X \\ S_{k_0,x} = 0 & \forall x \notin X_{k_0}, \end{cases} \qquad (7)$$

where K is the number of superpixels, and $X_k \subset X$ is the pixel pool of the kth superpixel. $S_{k,x}$ is the value of the membership function of superpixel k at pixel x, i.e., $S_{k,x} = \mu_k(x)$.

The idea of fixing both the number of superpixels and the pixel pools is borrowed from SLIC. However, SLIC is based

[1] The pretrained model was from the PyTorch project, https://pytorch.org.

on the k-means algorithm and thus is nondifferentiable, which makes it impossible for SLIC to participate in the training process. Therefore, we propose extending the idea of SLIC into a more general and differentiable form.



(a) Overlapping (b) Cell Numbers (c) Competition

Fig. 2. Illustration of pixel pools and the competition between superpixels.

In our experiments, pixel pools are defined as the simplest shape and are *rectangles, each composed of* 3×3 *grid cells*, and the stride between pools is 1 *cell* for both the horizontal and vertical directions, as shown in Fig. 2(a) and (b). Pixel pools of nearby superpixels overlap in cells as shown in Fig. 2(a), and in the overlapping cells, superpixels will compete for the ownership of pixels as in Fig. 2(c). However, in the proposed method, the process of competition is unnecessary, and it is only used for visualizing the superpixel representations.

A. Fuzzy Superpixel Extraction Network

Since the pixel pool of each superpixel is defined as shown in Fig. 2, the number of potential owner superpixels of each pixel should be limited, i.e., assuming Ω_x is the set of possible owners of pixel x, there is a constant $O >= |\Omega_x|$ for $\forall x$. In addition, $O = 9$ in our experiments because pixel pools have *a stride of* 1 *cell*, and each consists of 3×3 *grid cells*.

As stated in Sec. I, the fuzzy superpixel extraction network (FSEN) in the proposed method is implemented as a convolutional neural network (CNN) with the dimensions of its output S being fixed to $O \times |X|$. *Each element of* S *is associated with a tuple* (k, x)*, where* $x \in X$ *and* $k \in \Omega_x$*, and* $S_{k,x} = \mu_k(x)$. To fit the elements of S into matrices, cells around each pixel are numbered relative to the position of that pixel as P in Fig. 2 (b), and the superpixel centered at each of the cells will share its number. Thus, the order of superpixels in Ω_P can be defined, so that the values of their membership functions fit into the output vector of P.

Finally, to fulfill the first requirement in Eq. (7), the last layer of FSEN is a spatial Softmax layer.

B. Superpixel Pooling and Recovering Layers

Two other important components related to FSEN are the superpixel pooling layer (SPL) and the superpixel recovery layer (SRL). Simply speaking, *they switch working levels between the pixel level and the superpixel level*. Specifically, SPL pools pixelwise features within each superpixel to generate a superpixelwise feature map, and SRL recovers pixelwise information according to the superpixel representation and the given superpixelwise information.

Sec. II defined $SPL : \mathbb{R}^{D_F \times |X|} \times \mathbb{R}^{O \times |X|} \rightarrow \mathbb{R}^{D_F \times K}$ and $SRL : \mathbb{R}^{K} \times \mathbb{R}^{O \times |X|} \rightarrow \mathbb{R}^{|X|}$; thus, the forward and backward processes of SPL and SRL can be formulated as below:

$$SPL_{i,k}(F_p, S) = \frac{\sum_{x \in X_k} S_{k,x} \cdot F_p^{(i,x)}}{\sum_{x \in X_k} S_{k,x}} \quad (8)$$

$$g^{(SPL)}_{F_p^{(i,x)}}(F_p, S, g_{F_s}) = \sum_{k \in \Omega_x} g_{F_s^{(i,k)}} \cdot \frac{S_{k,x}}{\sum_{y \in X_k} S_{k,y}} \quad (9)$$

$$g^{(SPL)}_{S_{k,x}}(F_p, S, g_{F_s}) = \sum_{i=1}^{D_F} g_{F_s^{(i,k)}} \cdot \left(\frac{F_p^{(i,x)}}{\sum_{y \in X_k} S_{k,y}} - \frac{\sum_{y \in X_k} S_{k,y} \cdot F_p^{(i,y)}}{\left(\sum_{y \in X_k} S_{k,y}\right)^2} \right) \quad (10)$$

$$SRL_x(Y_s, S) = \sum_{k \in \Omega_x} S_{k,x} \cdot Y_s^{(k)} \quad (11)$$

$$g^{(SRL)}_{Y_s^{(k)}}(Y_s, S, g_Y) = \sum_{x \in X_k} g_{Y_x} \cdot S_{k,x} \quad (12)$$

$$g^{(SRL)}_{S_{k,x}}(Y_s, S, g_Y) = g_{Y_x} \cdot Y_s^{(k)}, \quad (13)$$

where $g_\nu = \partial E / \partial \nu$ is the partial derivative of error E with respect to variable ν, and $g_\nu^{(M)}$ is a term of $\partial E / \partial \nu$ given by module M, i.e., $\partial E / \partial \nu = \sum_M g_\nu^{(M)}$.

IV. Controlled Filter Convolution

In neural networks, the input of a layer sometimes consists of several feature maps generated by different modules. Theoretically, deep learning techniques should be capable of balancing the weights of different parts of the input. However, when channels of the feature maps differ greatly, learning processes could prove difficult to converge in practice.

Therefore, we propose using controlled filter convolution (CFC) instead of traditional convolution layers to process inputs consisting of 2 feature maps. CFC is inspired by dynamic filter convolution (DFC) [20] and edge-conditioned convolution (ECC) [21], which are designed to handle graph data, and it can be seen as a task-specific version of DFC. Sec. IV-A presents a detailed explanation.

As shown in Fig. 3, CFC is mainly used in the saliency evaluation network to process superpixelwise features, and superpixel shape description features simultaneously. Specifically, CFC balances two different parts of the information possessed by superpixels to provide a unified feature map for the following process.

A. Formulation of CFC

CFC is a modification of the traditional convolution; thus, its formulation will be presented with a comparison to that of convolution layers in this section. To simplify the formulas, 3 important symbols need to be defined:

- R_x: assuming R is the set of all receptive fields, $R_x \subset R$ is the set of receptive fields containing pixel x, i.e., $R_x = \{r \in R : x \in r\}$;

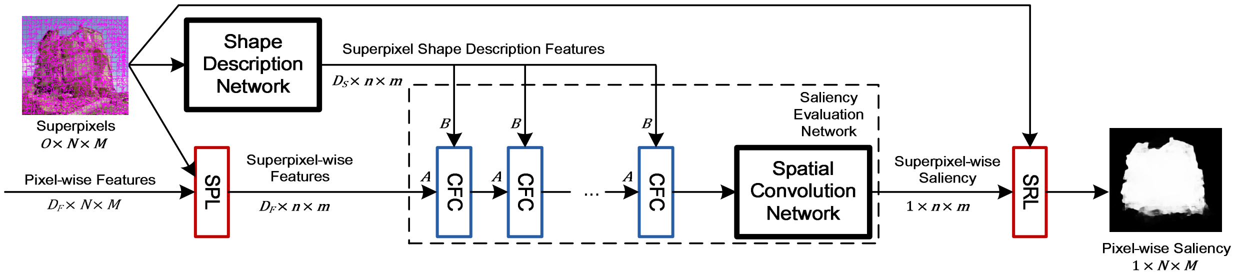


Fig. 3. The working process of saliency evaluation networks with controlled filter convolution. Components marked by red boxes are nonparameterized layers, those marked by black boxes are conventional convolutional neural networks, and the blue boxes mark controlled filter convolution layers.

- $\theta_r(x)$: when pixel x belongs to receptive field r, i.e., $x\in r$, $\theta_r(x)$ gives the position of x within r; and
- $\eta_r(i)$: it is the pixel $x\in r$ that satisfies $\theta_r(x)=i$.

Mathematically, convolution operations on any kind of data can all be described as a weighted sum of the elements in a selected receptive field, i.e., $o_r=\sum_{x\in r}\vec{e}_x^{\mathrm{T}}\vec{w}_{\theta_r(x)}+\beta_{\theta_r(x)}$, where o_r is the output for receptive field r, $\vec{e}_x$ is the feature vector associated with pixel x, $\vec{w}_i$ is the ith weight vector, and β_i is the ith optional bias parameter. Since the shape of receptive field r would not affect this formulation, without losing generalization, the equations in this section will only be presented in the 1D version for simplification.

As stated above, CFC simultaneously accepts two different feature maps, A and B. For convenience, their associated feature vectors are $\vec{a}_x$ and $\vec{b}_x$, respectively, i.e., $\vec{a}_x$ and $\vec{b}_x$ are similar to $\vec{e}_x$ in traditional convolution operations. Thus, the forward process can be formulated as below:

$$o_r=\sum_{x\in r}\left(\vec{a}_x^{\mathrm{T}}\mathbf{H}_{\theta_r(x)}\vec{b}_x+\vec{a}_x^{\mathrm{T}}\vec{\gamma}_{\theta_r(x)}+\vec{\alpha}_{\theta_r(x)}^{\mathrm{T}}\vec{b}_x+\beta_{\theta_r(x)}\right) \tag{14}$$

where $\{(\mathbf{H}_i,\vec{\gamma}_i,\vec{\alpha}_i,\beta_i)\}_{i=1}^{|r|}$ is a set of parameters, which is represented as Γ in the remainder of this section. Obviously, only one filter is considered in Eq. (14), but it can be easily applied to multifilter scenarios by maintaining a different parameter set Γ for each individual filter.

Eq. (14) can also be described as an example of the DFC framework [20]. DFC defines filters as functions of the input feature vectors, i.e., $\vec{w}_i^{(r)}=\vec{\mathfrak{w}}(\vec{e}_{\eta_r(i)})$ and $\beta_i^{(r)}=\mathfrak{b}(\vec{e}_{\eta_r(i)})$, and thus, the convolution becomes $o_r=\sum_{x\in r}\vec{e}_x^{\mathrm{T}}\vec{\mathfrak{w}}(x)+\mathfrak{b}(x)$.

We simplified the model by defining $\vec{\mathfrak{w}}(x)$ and $\mathfrak{b}(x)$ as linear functions and specific to each position. Specifically, considering $\vec{a}_x$ as $\vec{e}_x$, we obtain

$$\begin{aligned}\vec{\mathfrak{w}}_{\theta_r(x)}(\vec{b}_x)&=\mathbf{H}_{\theta_r(x)}\vec{b}_x+\vec{\gamma}_{\theta_r(x)} &(15)\\ \mathfrak{b}_{\theta_r(x)}(\vec{b}_x)&=\vec{\alpha}_{\theta_r(x)}^{\mathrm{T}}\vec{b}_x+\beta_{\theta_r(x)}, &(16)\end{aligned}$$

and similarly, $\vec{b}_x$ as $\vec{e}_x$, i.e., when considering one of the input feature maps as input, the filters of CFC are *CONTROLLED* by the other feature map. Meanwhile, in the above equations, $\vec{a}_x$ and $\vec{b}_x$ are mutually commutable, which also proves that CFC is capable of balancing the influence of A and B.

B. Optimization of CFC

To obtain the gradients of CFC, first let us consider the partial derivatives of each single output element:

$$\begin{aligned}\frac{\partial o_r}{\partial\vec{a}_x}&=\mathbf{H}_{\theta_r(x)}\vec{b}_x+\vec{\gamma}_{\theta_r(x)} &(17)\\ \frac{\partial o_r}{\partial\vec{b}_x}&=\mathbf{H}_{\theta_r(x)}^{\mathrm{T}}\vec{a}_x+\vec{\alpha}_{\theta_r(x)}. &(18)\end{aligned}$$

Then, the input gradient of each input element can be presented as the sum of the partial derivatives among all receptive fields containing this input element:

$$g_{\vec{e}}^{(CFC)}(A,B,\Gamma,g_o)=\sum_{r\in R_x}g_{o_r}\cdot\frac{\partial o_r}{\partial\vec{e}},\vec{e}\in\left\{\vec{a}_x,\vec{b}_x\right\} \tag{19}$$

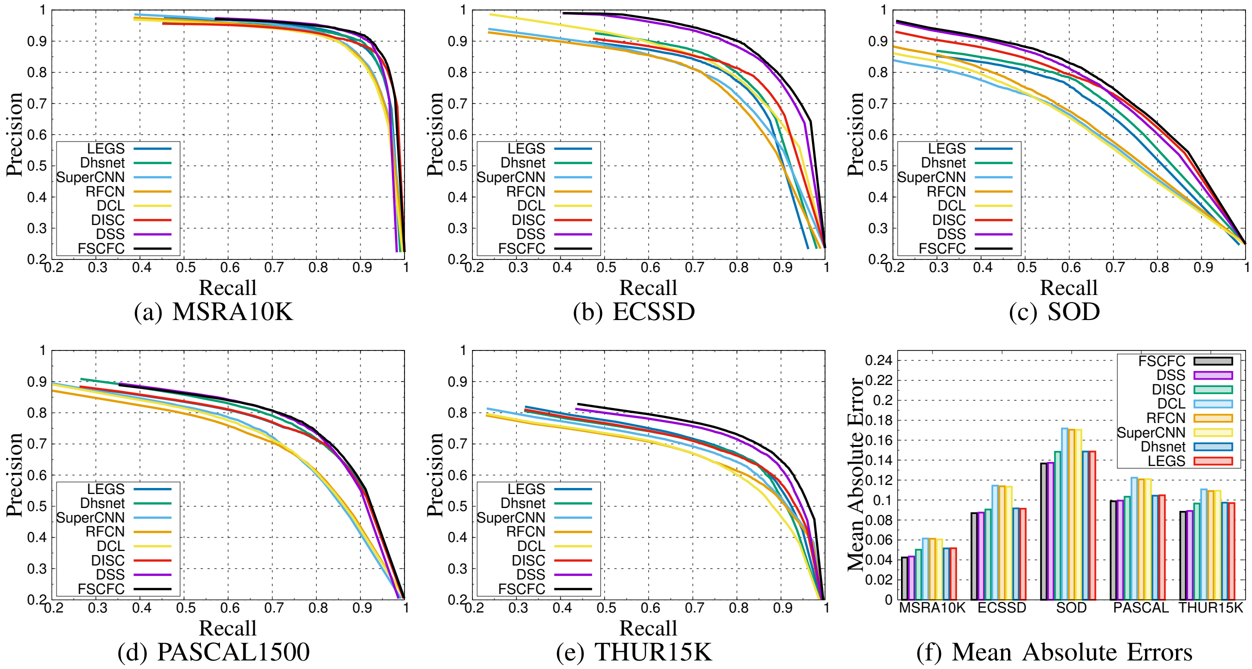
Parameter set Γ of CFC consists of 4 parts and is slightly different than that of conventional convolution; thus, the gradients of different parameters are discussed separately. However, due to the simple nature of Eq. (14), the formulas of the gradients can be easily derived with matrix operations:

$$\begin{aligned}g_{\beta_i}(A,B,\Gamma,g_o)&=\sum_{r\in R}g_{o_r} &(20)\\ g_{\vec{\gamma}_i}(A,B,\Gamma,g_o)&=\sum_{r\in R}g_{o_r}\cdot\vec{a}_{\eta_r(i)} &(21)\\ g_{\vec{\alpha}_i}(A,B,\Gamma,g_o)&=\sum_{r\in R}g_{o_r}\cdot\vec{b}_{\eta_r(i)} &(22)\\ g_{\mathbf{H}_i}(A,B,\Gamma,g_o)&=\sum_{r\in R}g_{o_r}\cdot\vec{a}_{\eta_r(i)}\vec{b}_{\eta_r(i)}^{\mathrm{T}}. &(23)\end{aligned}$$

V. Evaluation

To evaluate the salient object detection performance of the proposed method, which will be noted as fuzzy superpixel and controlled filter convolution (FSCFC), a series of experiments are conducted on 5 datasets. These datasets are all publicly released datasets with pixel-level manual annotations for the salient object detection task, and they are widely used in DSOD studies, which makes them appropriate benchmarks for performance evaluation. They are as follows:

- MSRA10K [16] contains 10,000 pictures selected from the MSRA dataset. Most of the pictures only have one salient object located near the center; therefore, the



(a) MSRA10K

(b) ECSSD

(c) SOD

(d) PASCAL1500

(e) THUR15K

(f) Mean Absolute Errors

Fig. 4. Comparison of experimental results of the competing methods on 5 public datasets. (a)-(e) show the PR curves of all competing methods on each of the 5 datasets, and (f) contains the MAE results on all the datasets to show a numerical comparison between the competing methods.

dataset is relatively clean and, thus, easy to process. However, because the numbers of both pictures and object categories are relatively large, it can serve as a good training set;

- ECSSD [24] has 1,000 annotated pictures with mostly centered objects and a complex background;
- SOD [25] contains 300 images from the Berkeley Segmentation Dataset (BSD) [26], which have a large variety of salient objects;
- PASCAL1500 [27] contains 1,500 images selected from the PASCAL VOC 2012 segmentation task. These images usually have both a complicated background and multiple objects, which makes PASCAL1500 a rather challenging dataset; and
- THUR15K [28] is a large dataset with 15,000 pictures from 5 object categories, but only 6,233 of the pictures have corresponding annotations. This is also a relatively clean dataset since most of the pictures contain a single centered object.

Since ECSSD, SOD and PASCAL1500 are relatively small, they are difficult to use to train neural networks. Therefore, in the following experiments, networks are trained on MSRA10K dataset and then applied to the other 4 datasets. During training, images in MSRA10K were first divided into 3 subsets: the training set (8,000 images), evaluation set (1,000 images) and test set (1,000 images), and then, the networks were trained on the training set while being evaluated on the evaluation set. In the test stage, trained networks were applied to the test set of MSRA10K and the other 4 datasets. In these experiments, our method competed with 7 state-of-the-art DSOD methods, including SuperCNN [17], LEGS [11], DCL [12], DISC [18], Dhsnet [13], RFCN [29] and DSS [16]. The results are analyzed in detail in Sec. V-B.

TABLE I
NETWORK CONFIGURATION OF THE PROPOSED METHOD.

FEN	**FSEN**	**SEN**	
$conv2d^{(3)}_{3,64}$	$conv2d^{(48,16,32)}_{9,64}$	$cfc2d^{(3)}_{64,8,80}$	$conv2d^{(3)}_{128,64}$
ReLU	ReLU	ReLU	ReLU
$conv2d^{(3)}_{64,64}$	$conv2d^{(3)}_{64,64}$	$cfc2d^{(3)}_{80,8,80}$	$conv2d^{(3)}_{64,64}$
	ReLU	ReLU	ReLU
	$conv2d^{(3)}_{64,64}$	$cfc2d^{(3)}_{80,8,80}$	$conv2d^{(1)}_{64,64}$
	ReLU	ReLU	ReLU
	$conv2d^{(3)}_{64,32}$	$conv2d^{(3)}_{80,128}$	$conv2d^{(1)}_{64,64}$
	ReLU	ReLU	ReLU
SDN	$conv2d^{(3)}_{32,32}$	$conv2d^{(3)}_{128,256}$	$conv2d^{(1)}_{64,64}$
$conv2d^{(5)}_{64,64}$	ReLU	ReLU	ReLU
ReLU	$conv2d^{(3)}_{32,32}$	$conv2d^{(3)}_{256,512}$	$conv2d^{(1)}_{64,1}$
$conv2d^{(3)}_{64,64}$	ReLU	ReLU	Sigmoid
ReLU	$conv2d^{(3)}_{32,16}$	$conv2d^{(3)}_{512,256}$	
$conv2d^{(3)}_{64,8}$	ReLU	ReLU	
	$conv2d^{(1)}_{16,9}$	$conv2d^{(3)}_{256,128}$	
	Softmax2d	ReLU	

A. Network Configuration

For a better understanding of the following experiments, the experiment network configuration of FSCFC is given here.

In Tab. I, a ReLU is a rectified linear unit, a Softmax2d is a spatial Softmax layer, and Sigmoid represents the sigmoid function. $conv2d^{(k,s,p)}_{i,o}$ is a 2D convolution layer with i input channels, o output channels, $k \times k$ kernel, a stride of s and p padding. When not given, s is the default and set as 1 and p is the default set as $(k-1)/2$. $cfc2d_{a,b,o}^{(k)}$ is similar to $conv2d_{i,o}^{(k,s,p)}$ with a and b representing the number of channels in the input feature map A and B, respectively.

In addition, before being passed to the network, images are all resized to 256×256 pixels. The number of superpixels, i.e., K is set to 256.

B. Salient Object Detection Performance

As shown in Fig. 4, FSCFC significantly outperformed all the other competing methods on all the datasets except PASCAL1500, in which the performances of FSCFC and DSS were very close. In addition to FSCFC, DSS achieved the second-best performance on most of the datasets, especially on ECSSD and THUR15K. The reason could be that the backbone of DSS is a pretrained VGG16 network, which contains extra information from ImageNet. Although SuperCNN, DISC and FSCFC all benefit from superpixels, their performances showed that FSE, which is neural network-based, is more effective than nondifferentiable methods in the DSOD task.

Fig. 4(f) also shows the mean absolute errors (MAE) of the competing methods. When the PR curves are difficult to analyze, MAE can provide an easier method for comparing the performances of different methods. However, although theoretically, a lower MAE means that the output and ground truth are closer and thus means better performance, there are usually considerably more background pixels than salient pixels in the benchmark datasets; thus, models assigning more pixels to the background usually have a better MAE, while their PR curves are not as good.

C. Effect of Fuzzy Superpixel Extraction

Fuzzy superpixel extraction (FSE) is one of the most important parts of FSCFC; thus, a series of experiments are conducted to evaluate its effect compared with the SLIC [19] algorithm, which is one of the state-of-the-art superpixel extraction methods.

In the experiments in this section, FSE in our working network is replaced with SLIC to form a competitive network. For this network to work, an additional preprocessing stage was added between the SLIC and the neural network. This preprocess converts the superpixels given by SLIC to a matrix $S^{\#}$ similar to S so that it can be used as the input of the SDN, SPL and SRL. In addition, in the backward process, gradients with respect to $S^{\#}$ were discarded.

Fig. 5(a) presents the results of the above experiments. On all the datasets, FSE outperformed SLIC significantly, especially on MSRA10K. This is probably because, unlike SLIC, FSE can participate in the training process, which makes it easier for FSE to be optimized according to the actual data.

D. Effect of Controlled Filter Convolution

In FSCFC, controlled filter convolution (CFC) is used as a substitute for the traditional convolution to improve the performance of the entire network. However, without CFC it is still possible to build working networks for our purpose. Therefore, the necessity of using CFC needs to be evaluated, and thus, we conducted some experiments in this section.

To build the competitive network, the CFC layers in the SEN were all replaced with traditional convolution layers, specifically spatial convolution (SC) layers. Because the superpixelwise feature map and superpixel shape description feature map, which were originally processed by the CFC layers, are both supposed to be the input of SEN, they were concatenated by channels before passing to the SEN. The rest of the network was the same as that using the CFC layers.

To reduce the influence of the number of network parameters, the newly added convolution layers were carefully configured to have approximately the same number of parameters as the original CFC layers. As shown in Fig. 5(b), even with the same number of parameters, the network consisting of CFC layers outperformed its counterpart, which has only spatial convolution layers.

The different behaviors can be explained by the different kernel operations on which CFC and traditional convolution are based. In traditional convolution, the output is merely a weighted sum of all the input feature maps, but in CFC, the two input feature maps have a more complex relationship with each other, which increases the expression ability.

E. Effect of Spatial Regularization

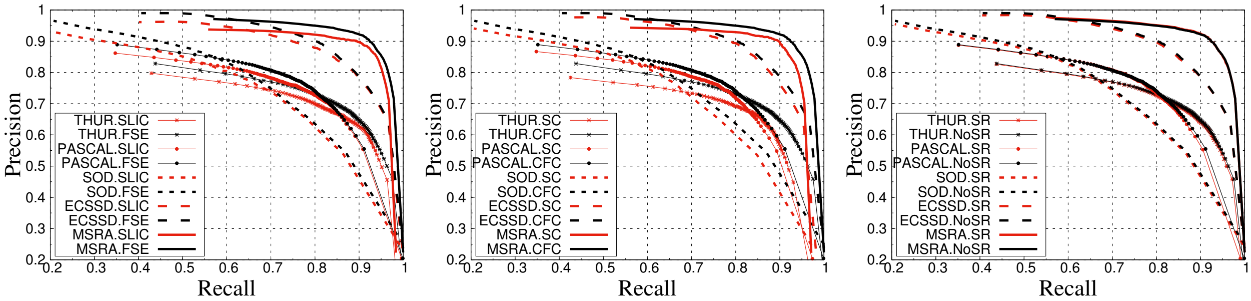
Some of the SOD methods, including DISC, use spatial regularization (SR) to boost their performance, i.e., the methods use prior knowledge (or assumption) of the spatial distribution of salient pixels to guide the detection process. Some methods directly add SR rules, such as favoring pixels near the center, to their models, and some recent DSOD methods such as DISC add an additional input channel besides the original color channels to provide their networks with SR information.

However, SR rules are mostly empirical and highly dependent on the properties of the training dataset, which can lead to bad generalization performance. In the experiments in Sec. V-B, two different training strategies (with and without SR) were applied to verify the effectiveness of SR in DSOD.

As shown in Fig. 5(c), two training strategies showed no significant difference on MSRA10K, and on the other 4 datasets SR seemingly impaired the generalization ability of the models and thus caused the performance to drop slightly. This cannot disprove the effectiveness of SR in nondeep SOD methods, but it proved that the necessity of using SR rules in DSOD methods is doubtful.

VI. Conclusion

This paper proposed a novel deep salient object detection method using fuzzy superpixel extraction (FSE) and controlled filter convolution (CFC), which are specially designed for using in DSOD methods. By making the membership functions of superpixels continuous, FSE can be easily implemented with neural network modules and embedded into any network with the help of SPL and SRL. CFC modified dynamic filter convolution into a task-specific linear version that accepts two



(a) FSE (b) CFC (c) SR

Fig. 5. Results of analysis experiments. (a) compares FSE and SLIC as the superpixel extraction method for DSOD, (b) compares the networks with and without CFC layers, and (c) shows the performance of FSCFC with and without SR. Each figure of (a)-(c) contains the results on all 5 datasets.

input feature maps and is capable of balancing their influences without any hand-picked coefficients involved.

To evaluate the proposed method, i.e., FSCFC, a series of experiments were conducted to compare the performance of F-SCFC with 7 state-of-the-art DSOD methods on 5 widely used benchmark datasets. These experiments proved that FSCFC significantly outperforms the competing DSOD methods on all the datasets. Meanwhile, the experimental results on ECSSD, SOD, PASCAL1500 and THUR15K also showed that FSCFC has higher generalization ability than the competing methods.

References

[1] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[2] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on*. IEEE, 2009, pp. 1597–1604.

[3] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 733–740.

[4] X. Li, H. Lu, L. Zhang, X. Ruan, and M.-H. Yang, "Saliency detection via dense and sparse reconstruction," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2976–2983.

[5] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 3166–3173.

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

[7] K. Wang, L. Lin, J. Lu, C. Li, and K. Shi, "Pisa: Pixelwise image saliency by aggregating complementary appearance contrast measures with edge-preserving coherence," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3019–3033, 2015.

[8] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," *IEEE Transactions on Pattern analysis and machine intelligence*, vol. 33, no. 2, pp. 353–367, 2011.

[9] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, "Salient object detection: A discriminative regional feature integration approach," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2083–2090.

[10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004.

[11] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, "Deep networks for saliency detection via local estimation and global search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3183–3192.

[12] G. Li and i. Yu, "Deep contrast learning for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 478–487.

[13] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 678–686.

[14] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1265–1274.

[15] G. Lee, Y.-W. Tai, and J. Kim, "Deep saliency with encoded low level distance map and high level features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 660–668.

[16] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr, "Deeply supervised salient object detection with short connections," in *IEEE CVPR*, 2017, pp. 3203–3212.

[17] S. He, R. W. Lau, W. Liu, Z. Huang, and Q. Yang, "Supercnn: A superpixelwise convolutional neural network for salient object detection," *International journal of computer vision*, vol. 115, no. 3, pp. 330–344, 2015.

[18] T. Chen, L. Lin, L. Liu, X. Luo, and X. Li, "Disc: Deep image saliency computing via progressive representation learning." *IEEE Trans. Neural Netw. Learning Syst.*, vol. 27, no. 6, pp. 1135–1149, 2016.

[19] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," Tech. Rep., 2010.

[20] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 667–675.

[21] M. Simonovsky and N. Komodakis, "Dynamic edgeconditioned filters in convolutional neural networks on graphs," in *Proc. CVPR*, 2017.

[22] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[24] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1155–1162.

[25] V. Movahedi and J. H. Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE, 2010, pp. 49–56.

[26] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423.

[27] W. Zou, K. Kpalma, Z. Liu, and J. Ronsin, "Segmentation driven low-rank matrix recovery for saliency detection," in *24th British machine vision conference (BMVC)*, 2013, pp. 1–13.

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

[29] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, "Saliency detection with recurrent fully convolutional networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 825–841.