Salient Object Detection Based on Multiscale Segmentation and Fuzzy Broad Learning

XIAO LIN[†], ZHI-JIE WANG[‡], LIZHUANG MA[#], RENJIE LI[†], MEI-E FANG[⊤]

[†]*Department of Computer Science and Engineering, Shanghai Normal University, Shanghai, China*
[‡]*College of Computer Science, Chongqing University, Chongqing, China*
[#]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China*
[⊤]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China*
Email: cszjwang@yahoo.cn

Saliency detection has been a hot topic in the field of computer vision. In this paper, we propose a novel approach that is based on multiscale segmentation and fuzzy broad learning. The core idea of our method is to segment the image into different scales, and then the extracted features are fed to the fuzzy broad learning system for training. More specifically, it first segments the image into superpixel blocks at different scales based on the simple linear iterative clustering (SLIC) algorithm. Then, it uses the local binary pattern (LBP) algorithm to extract texture features, and computes the average color information for each superpixel of these segmentation images. These extracted features are then fed to the fuzzy broad learning system (FBLS) to obtain multiscale saliency maps. After that, it fuses these saliency maps into an initial saliency map, and uses the label propagation algorithm (LPA) to further optimize it, obtaining the final saliency map. We have conducted experiments based on several benchmark datasets. The results show that our solution can outperform several existing algorithms. Particularly, our method is significantly faster than most of deep learning based saliency detection algorithms, in terms of training and inferring time.

Keywords: Saliency detection; computer vision; image processing, machine learning

Received 00 January 2009; revised 00 Month 2009

1. INTRODUCTION

In the big data era, social images and video data can be found everywhere, and the amount of such data is growing sharply. It is time-consuming to deal with such a huge image data. Usually, users could be interested in a fraction of information for most of images. As such, it is important to extract important information from images, since extracting such information can benefit to many important applications such as image retrieval, visual tracking, etc. [1, 2, 3, 4, 5, 6, 7, 8]. Visual saliency (known as *saliency detection*) is one of the classical ways to find regions of interest in the images. Particularly, to date, there are a large bulk of works that address the problem of saliency detection [9, 10, 11, 12, 13, 14, 15].

Traditional saliency models are generally based on low-level information of the image, such as color, background, texture, edge and spatial position. The first saliency detection model based on the low-level information was proposed by Itti *et al.* [16]. In this model, the input image is decomposed into three channels: intensity, color and direction, and then the center-surround contrast is used to find the saliency region. After that, many low-level information based saliency detection models were developed, such as the global-to-local saliency detection framework [17], superpixel segmentaion based method [17, 18, 19]. However, this line of methods may not achieve the strong performance (as shown in Fig. 1), since low-level information cannot fully reflect the internal/underlying structure of the image.

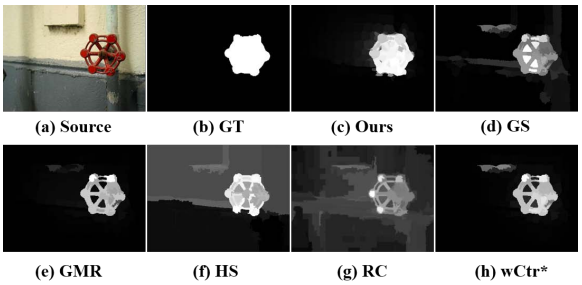


FIGURE 1: An example of the saliency maps generated by our proposed approach and several existing saliency models. From (a) to (h): source image, ground-truth (GT), ours, GS [20], GMR [19], HS [21], RC [22], wCtr* [23].

On the other hand, deep neural network [24, 25, 26] can extract many high-level semantic information of images, greatly increasing the number of features. Thus, it is usual that deep learning based saliency detection models can

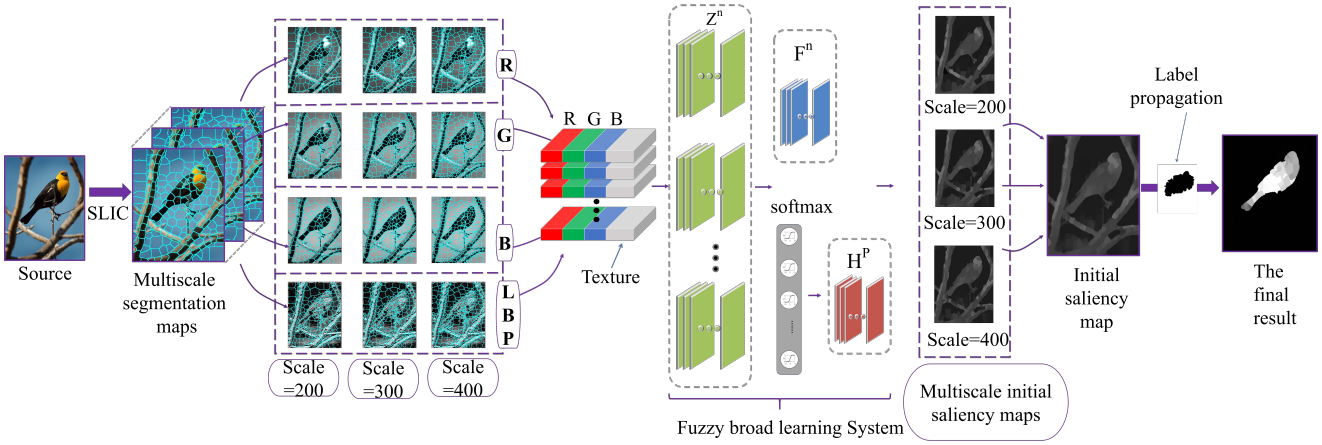


FIGURE 2: Framework of our solution.

achieve stronger performance than the traditional saliency detection algorithms [27, 28, 29, 25]. In this line of methods, although deep neural networks can greatly improve the effectiveness of saliency detection, their complex frameworks usually rely on high-performance computers, and easily incur the extremely long training processes.

To alleviate the issues mentioned above, we propose a novel method for saliency detection. The framework is shown in Fig. 2. In brief, we first use the simple linear iterative clustering (SLIC) algorithm [30] to segment the image at different scales, in order to effectively retain the structural integrity. Then, we use the local binary pattern (LBP) algorithm [31] to extract the texture features of the superpixel, and we compute the average value of the superpixel on (R, G, B) channels. We then combine the color information with the extracted texture information as the features of the superpixel block. Later, we take the superpixel features of images as the input, and feed them to a fuzzy broad learning system (FBLS) [32] to obtain multiscale salient maps. The FBLS can allow us to incorporate the idea of the incremental learning, which only need to increase the new data, fuzzy subsystems and/or enhancement nodes, without the need of re-training the whole dataset starting from scratch. The FBLS reduces the need for high-performance computers and also greatly reduces the training time, compared against deep learning based saliency detection models. After obtaining the multiscale saliency maps, we fuse them into an initial saliency map, and further use the label propagation algorithm (LPA) [33] to optimize it, obtaining the final saliency map. Our proposed model is superior to some existing methods in accuracy (see Fig. 1 for an illustration), and is superior to most of deep learning saliency detection models in terms of model training and inferring time. To summarize, this paper makes the following contributions:

- We propose a novel saliency detection model. Our model takes full use of the low-level information of the image at different scale, which allows us to effectively retain the structural integrity. Meanwhile, it leverages the fuzzy broad learning structure to achieve better training and inferring efficiency.
- We conduct extensive experiments based on various datasets. The experimental results demonstrate that our model can outperform several state-of-the-art algorithms in terms of accuracy, while it can achieve shorter training and inferring time, compared against many deep-learning based saliency detection models.

The rest of this paper is organized as follows. Section 2 reviews the related work. In Section 3, we present our proposed solution in detail. Section 4 presents experimental results and discussions. We conclude the paper in Section 5.

2. RELATED WORK

In the past decades, various saliency detection models were developed. Generally, these models can be classified into two categories: (1) the low-level information based models [9, 10, 11, 12, 13, 17, 18, 19]; and (2) the high-level semantic information based models [14, 15, 24, 26]. As for the first line of methods, they significantly leverage the low-level information features of images, e.g., color, texture, and contour. As an example, Xie *et al.* [12] developed a method that detects the rough contour of the salient object by using the Harris corner detection algorithm, which significantly leverages the color information in the image. However, this line of methods may not achieve the strong performance, since low-level information could not fully reflect the internal/underlying structure of the image, as pointed out in [34, 15]. To alleviate the limitation of relying only on the low-level information features, the high-level semantic information based models were heavily developed in recent several years [27, 28, 29, 25]. Many deep learning based saliency detection models generally belong to this branch, which leverage the high-level features of the images in various manners. For example, Li *et al.* [29] proposed a multiscale deep learning method for saliency detection. Luo *et al.* [25] proposed a non-local deep learning model for saliency detection. The line of methods capture representative high-level features, thereby detecting salient objects of certain sizes and categories more

precisely. A major drawback of this line of methods could be that they usually suffer too much time for training their models, since excessive parameters need to be trained via extensive datasets. Compared against the methods above, our paper suggests a multiscale fuzzy broad learning model for saliency detection, our model leverage the merits of fuzzy broad learning system, which can achieve favorable detection performance, while it has faster training time than many state-of-the-art deep learning based models, as demonstrated in Section 4.

We remark that, SLIC [30] is an excellent and stable algorithm for image segmentation. It is a k-means based local clustering on pixels in *LAB color space*[1], which can cluster adjacent pixels with similar features into one block. In existing works there are already many saliency detection algorithms (including traditional and deep learning based ones) incorporated it [35, 12, 13]. An obvious benefit of using SLIC algorithm in saliency detection is that, it can simplify the calculations of the rest steps, greatly improving the computation efficiency [35, 12]. In addition, it also benefits to effectively retaining the structural integrity [12, 13]. Inspired its superiorities, in our model we also integrate this algorithm. Nevertheless, compared against existing saliency detection algorithms that use SLIC, our model is obviously different from theirs, since all those models did not use the FBLS.

As for the FBLS, it combines/integrates the Takagi-Sugeno (TS) fuzzy system [32] on basis of broad learning system (BLS) [36, 37, 38]. The BLS is a framework that can replace the structure of deep learning [38, 37, 36]. The strong points of BLS are that, it can alleviate the troubles/issues (e.g., long training time) caused by a large number of parameters in deep learning based models. Unlike deep learning models, which consist of the variable structure and the stack of neuron layers, the BLS is constructed by neurons, which are composed of feature nodes and enhancement nodes. They do not need to stack layers in deep, but only need to expand in broad and get weights by calculating pseudo-inverse. To fusion TS fuzzy system with BLS, the FBLS replaces the feature nodes in BLS with a group of TS fuzzy subsystems, and then the intermediate outputs (generated by all fuzzy subsystems) are sent to the enhancement nodes as a vector connection for further nonlinear transformation. The final output is generated by combing the inputs from both the fuzzy subsystem and the enhancement nodes. Since the FBLS saves the adjustment process of sparse autoencoder in BLS, it reduces the complexity of the structure and improves image recognition performance [32]. On the other hand, compared against the TS fuzzy system, it has fewer rules and shorter running time, and it works better. Particularly, even for the high-dimensional input data, it can also perform well. Inspired by these merits, in our paper we attempt to leverage FBLS for saliency detection.

[1]It is one of the most popular color spaces (e.g., RGB, CMYK, HSV, YUV) for measuring object colors. The full name of "LAB color space" is "CIELAB color space", defined by the International Commission on Illumination (CIE) in 1976. More information please refer to https://www.xrite.com/blog/lab-color-space.

3. OUR SOLUTION

3.1. Training via FBLS

We need some preprocessing operations before training our model. We first discuss the preprocessing operations, and then show the details of the training process.

3.1.1. Preprocessing for Training

Before training FBLS, if the format of the image dataset does not meet our requirements, we need to preprocess the image dataset [39]. The method for preprocessing the training set is as follows. Firstly, SLIC is applied to several different scales. Let N_{pixel} be the number of pixels of the image, and K_{super} be the desired number of superpixels. In SLIC algorithm, the surrounding pixels of clustering center are moved based on the distance similarity measure:

$$\begin{aligned} d_{lab} &= \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \\ d_{xy} &= \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \\ D_S &= d_{lab} + \frac{\kappa}{S} d_{xy} \end{aligned} \tag{1}$$

where D_S is the sum of the *lab* distance and the *xy* plane distance normalized by the grid interval S; the variable κ is a parameter, which can be used to control the compactness of a superpixel. The larger the value of κ is, the more compact the clustering is.

Then, the average color of R, G, and B channels of each superpixel is extracted. Meanwhile, the LBP algorithm is used to extract the texture features of each superpixel, and finally all information extracted from each superpixel is pulled into a one-dimensional vector. The dataset then becomes a matrix of all the superpixel information for all the images. The corresponding truth value is to classify the corresponding superpixel block of each image, where "1" represents that the superpixel block belongs to the foreground and "0" represents that the superpixel block belongs to the background. The preprocessed data can be fed to FBLS for training, which is equivalent to the binary classification of each superpixel block.

3.1.2. Training

We use the preprocessed image dataset and the corresponding ground truth maps for training. That is, the input data is $X = (x_1, x_2, ..., x_N) \in \mathbb{R}^{N \times M}$, and the ground truth maps Y. The overall structure of our training model is shown in Fig. 3.

In our model, we use the first-order Takagi-Sugeno (TS) fuzzy subsystem (see the left part of Fig. 3). It is used to map the input data $x_s = (x_{s1}, x_{s2}, ..., x_{sM}), s = 1, 2, ..., N$ to the i-th fuzzy subsystem with k^i fuzzy rules. Fig. 4 plots the framework of the first-order TS fuzzy subsystem. Note that, each subsystem shall produce two results: (1) the intermediate output vector Z_{si} (cf., the dashed rectangle in

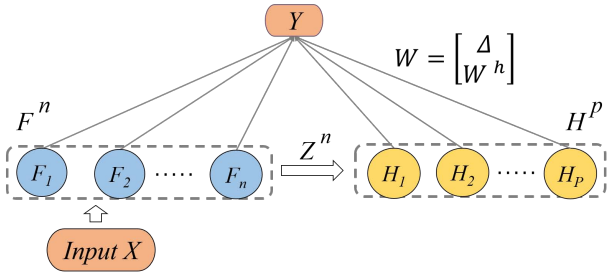


FIGURE 3: The structure of our training model. It contains input and output layers. The input layer contains n fuzzy subsystems and p enhancement nodes. The fuzzy subsystems and the enhancement nodes are connected to the output layer through the weight W.

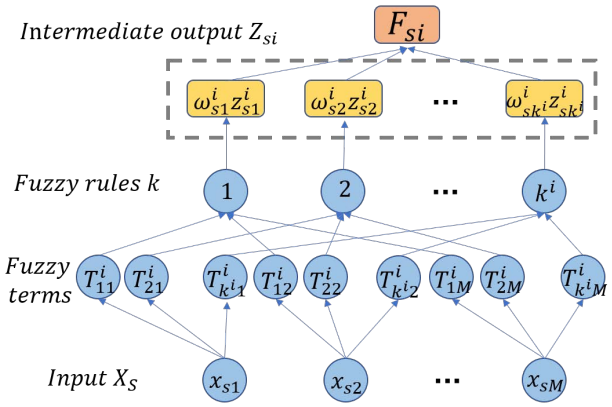


FIGURE 4: The structure of the first-order TS fuzzy subsystem.

Fig. 4), and (2) the output vector F_{si} (cf., the top of Fig. 4). Later, we will show more details on how to obtain them.

► *Obtaining Z_{si} and F_{si}.* As for each fuzzy rule, it shall produce two elements: (1) temporal vector z^i_{sk}, and the weighted fire strength ω^i_{sk}. Formally, they are computed as follows (see Equations 2 and 3).

$$z^i_{sk} = f^i_k(x_{s1}, x_{s2}, ...x_{sM}) = \sum_{t=1}^{M} \alpha^i_{kt} x_{st} \tag{2}$$

where $k = 1, 2, ..., k^i$ is the fuzzy rule of the i-th fuzzy subsystem, and α^i_{kt} is a coefficient that is uniformly distributed on $[0, 1]$.

$$\omega^i_{sk} = \frac{\prod_{t=1}^{M} \mu^i_{kt}(x_{st})}{\sum_{k=1}^{k^i} \prod_{t=1}^{M} \mu^i_{kt}(x_{st})} \tag{3}$$

where $\mu^i_{kt}(x)$ refers to the Gauss membership function, which is computed as

$$\mu^i_{kt}(x) = e^{-\left(\frac{x - c^i_{kt}}{\sigma^i_{kt}}\right)^2} \tag{4}$$

where c^i_{kt} denotes the center of Gauss membership function for the fuzzy subsystem. A set of clustering centers can be obtained by processing the training dataset via the k-means algorithm. In addition, σ^i_{kt} is an integer.

With the above two elements, we can get the intermediate output vector Z_{si} (cf., the dashed rectangle in Fig. 4) of the i-th fuzzy subsystem as follows.

$$Z_{si} = (\omega^i_{s1} z^i_{s1}, \omega^i_{s2} z^i_{s2}, ..., \omega^i_{sk^i} z^i_{sk^i}) \tag{5}$$

Similarly, we can obtain the output vector F_{si}. It is computed as

$$F_{si} = (\sum_{k=1}^{k^i} \omega^i_{sk} \lambda_1, ..., \sum_{k=1}^{k^i} \omega^i_{sk} \lambda_C) \tag{6}$$

where $\lambda_c = \sum_{t=1}^{M} \delta^i_{kc} \alpha^i_{kt} x_{st}, c = 1, 2, ..., C$, where c means the dimension corresponding to the training target. To reduce the total number of parameters in the training process, Equation 6 can be further decomposed as:

$$F_{si} = \sum_{t=1}^{M} \alpha^i_{kt} x_{st} (\omega^i_{s1}, ..., \omega^i_{sk^i}) \begin{pmatrix} \delta^i_{11} & \dots & \delta^i_{1C} \\ \vdots & & \vdots \\ \delta^i_{k^i 1} & \dots & \delta^i_{k^i C} \end{pmatrix} \tag{7}$$

To this step, we have shown how to obtain (1) the intermediate output vector Z_{si} and (2) the output vector F_{si}, using the TS fuzzy subsystem (cf., Fig. 4). Next, we show how to obtain F^n, Z^n, and H^p shown in Fig. 3.

► *Obaining F^n, Z^n, and H^p.* For all training samples, the output matrix of the i-th fuzzy subsystem is computed as:

$$F_i = (F_{1i}, F_{2i}, ..., F_{Ni})^T \triangleq D\Omega^i \delta^i \in \mathbb{R}^{N \times C} \tag{8}$$

where

$$D = diag\{\sum_{t=1}^{M} \alpha^i_{kt} x_{1t}, ..., \sum_{t=1}^{M} \alpha^i_{kt} x_{Nt}\},$$

$$\Omega^i = \begin{pmatrix} \omega^i_{11} & \dots & \omega^i_{1k^i} \\ \vdots & & \vdots \\ \omega^i_{N1} & \dots & \omega^i_{Nk^i} \end{pmatrix},$$

$$\delta^i = \begin{pmatrix} \delta^i_{11} & \dots & \delta^i_{1C} \\ \vdots & & \vdots \\ \delta^i_{k^i 1} & \dots & \delta^i_{k^i C} \end{pmatrix}.$$

So, for all n fuzzy subsystems, the output can be computed as:

$$\begin{aligned} F^n &= \sum_{i=1}^{n} F_i = \sum_{i=1}^{n} D\Omega^i \delta^i \\ &= D(\Omega^1, ..., \Omega^n) \begin{pmatrix} \delta^1 \\ \vdots \\ \delta^n \end{pmatrix} \\ &\triangleq D\Omega\Delta \in \mathbb{R}^{N \times C} \end{aligned} \tag{9}$$

where $\Omega = (\Omega^1, ..., \Omega^n) \in \mathbb{R}^{N \times (k^1 + k^2 + ... + k^n)}$ and $\Delta = ((\delta^1)^T, ..., (\delta^n)^T)^T \in \mathbb{R}^{(k^1 + k^2 + ... + k^n) \times C}$.

Algorithm 1 The Training Process of FBLS

Input: Training samples (X,Y), numbers of fuzzy rules k^i, n fuzzy subsystems, p enhancement nodes groups.
Output: The connecting weight of FBLS

Set $\sigma_{kt}^i = 1$, and initialize the coefficient α_{kt}^i in [0,1];
while $i <= n$ **do**
  Use k-means algorithm to calculate k^i clustering centers of training data X.
  Initialize the gaussian member function with $\sigma_{kt}^i = 1$ and k^i clustering centers.
  for $s = 1$; $s \leq N$ **do**
    Calculate Z_{si} using Eq. 5;
    Calculate F_{si} using Eq. 7;
  end for
  Obtain Z_i using Eq. 10;
  Calculate F_i using Eq. 8;
  $i = i+1$;
end while
Obtain Z^n using Eq. 11;
for $j = 1$; $j \leq p$ **do**
  Randomly generate W_j and β_j;
  Calculate $H_j = \varphi_j(Z^n W_j + \beta_j)$;
end for
Obtain H^p using Eq. 12;
Obtain F^n using Eq. 9;
Calculate the pseudoinverse $A^+ = (\lambda I + AA^T)^{-1}A^T$;
return $W = A^+ Y$;

Similarly, for all training samples, the intermediate output matrix of the i-th fuzzy subsystem can be computed as

$$Z_i = (Z_{1i}, Z_{2i}, ..., Z_{Ni})^T \in \mathbb{R}^{N \times k^i}, i = 1,2,...,n. \quad (10)$$

So, for all n fuzzy subsystems, we can obtain the intermediate output Z^n (cf., the right arrow in Fig. 3), which connects to the enhancement layer. It is computed as

$$Z^n = (Z_1, Z_2, ..., Z_n)^T \in \mathbb{R}^{N \times (k^1 + k^2 + ... + k^n)}, i = 1,2,...,n. \quad (11)$$

With Z^n, we use the nonlinear transformation in the enhancement layer (with p groups of enhancement nodes) to obtain the output H^p (cf., the right dashed rectangle in Fig. 3) of the enhancement layer. It is defined as

$$H^p = (H_1, H_2, ..., H_p) \in \mathbb{R}^{N \times (E_1 + E_2 + ... + E_p)} \quad (12)$$

where H_j ($j = 1,2,...,p$) is computed as

$$H_j = \varphi_j(Z^n W_j + \beta_j) \in \mathbb{R}^{N \times E_j} \quad (13)$$

where E_j is the number of neurons in the j-th group of enhancement nodes, W_j and β_j are the weight and bias terms of the intermediate output Z^n (of the fuzzy subsystem mapped to the enhancement layer), respectively; and $\varphi(\cdot)$ is a nonlinear transformation.

► *Putting All Together.* Finally, after obtaining the output of all fuzzy subsystems and enhancement nodes, we can get the final output Y of FBLS as follows:

$$\begin{aligned} Y &= [F_1, F_2, ..., F_n | [\varphi_1(Z^n W_1 + \beta_1), ..., \varphi_p(Z^n W_p + \beta_p)] W^h] \\ &= [F^n | H^p W^h] \\ &= [D\Omega\Delta | H^p W^H] \\ &= [D\Omega | H^p] \begin{bmatrix} \Delta \\ W^h \end{bmatrix} \\ &= AW \end{aligned} \quad (14)$$

where $A = [D\Omega | H^p]$, $W = \begin{bmatrix} \Delta \\ W^h \end{bmatrix}$, W^h is the mapping coefficient from the enhancement layer to the final output layer. Note that, Y in the training phase is the ground truth. Our goal in the training phase is to train W in Eq. 14. In other words, by Eq. 14, we can obtain the coefficient matrix W using the pseudoinverse calculation. That is,

$$W = A^+ Y \quad (15)$$

where A^+ is computed as

$$A^+ = (A^T A)^{-1} A^T \quad (16)$$

We remark that, the pseudo-inverse calculation is somewhat time-consuming, we can use the following method [36, 37, 38] to optimize the calculation. That is,

$$A^+ = \lim_{\lambda \to 0} (\lambda I + AA^T)^{-1} A^T \quad (17)$$

where λ is an equilibrium factor. By combing Eqs. 15 and 17, we can get the weight W. The overall process of training FBLS is summarized in Algorithm 1.

Algorithm 2 Incremental Learning

Input: Training samples (X,Y), the initial weight W.
Output: Updated weight W_{n+1}

while Model's accuracy does not meet the requirement **do**
  Obtain Z^{n+1} using Eq. 18;
  for $j = 1$; $j \leq p$ **do**
    Randomly generate W_j and β_j;
    Calculate $H_j = \varphi_j(Z^{n+1} W_j + \beta_j)$;
  end for
  Obtain $A_{n+1} = [A | Z_{n+1} | H_{n+1}^p]$;
  Calculate the pseudoinverse of A_{n+1} using Eq. 21;
end while
Update the initial weight W by Eq. 23;
return $W_{n+1} = W$;

► *Incremental Learning.* For our model, it is flexible to train newly added data, or to use additional enhancement nodes and fuzzy subsystems. We can use the incremental way to reconstruct the model. In what follows, we address how to perform the incremental learning by adding fuzzy subsystems (remark: the incremental learning by adding training samples or enhancement nodes can be handled similarly).

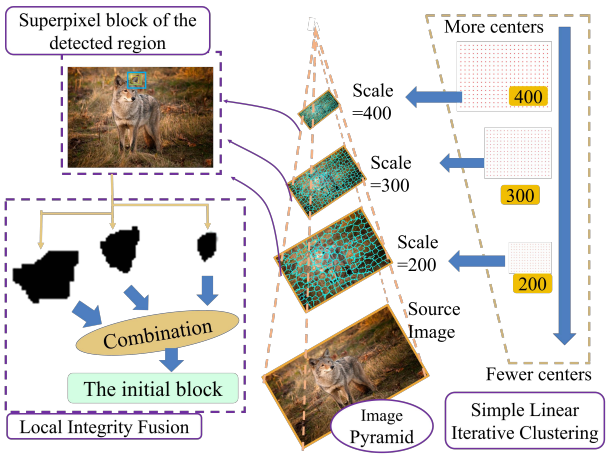


FIGURE 5: The fusion of multiscale saliency maps is to merge different superpixel blocks in the same region, so as to better maintain the integrity of the object structure.

Without loss of generality, assume that the intermediate output of the additional fuzzy subsystem is Z^{n+1}. It can be defined as follows:

$$Z^{n+1} = (Z_1, Z_2, ..., Z_n, Z_{n+1})^T \tag{18}$$

Similarly, the intermediate output Z^{n+1} is fed into the enhancement layer. Then, the output of the enhancement layer becomes $H^p_{n+1} = (H_1, H_2, ..., H_p)$, where H_j is computed as

$$H_j = \varphi_j(Z^{n+1}W_j + \beta_j), j = 1, 2, ..., p \tag{19}$$

So, we can obtain a new matrix A_{n+1} as follows:

$$A_{n+1} = [A|Z_{n+1}|H^p_{n+1}] \tag{20}$$

Also, we can calculate the pseudoinverse. It can be described as:

$$(A_{n+1})^+ = \begin{bmatrix} (A)^+ - DB^T \\ B^T \end{bmatrix} \tag{21}$$

where $D = (A)^+[Z_{n+1}|H^p_{n+1}]$, $C = [Z_{n+1}|H^p_{n+1}] - AD$, and B^T is represented as

$$B^T = \begin{cases} (C)^+ & \text{if } C \neq 0 \\ (D^T D + 1)^{-1} D^T (A)^+ & \text{if } C = 0 \end{cases} \tag{22}$$

Finally, for the additional fuzzy subsystem, the coefficient W of the model can be updated as follows:

$$W_{n+1} = \begin{bmatrix} W - DB^T Y \\ B^T Y \end{bmatrix} \tag{23}$$

The overall process of incremental learning is shown in Algorithm 2.

3.2. Generating Saliency Map

As for test phase, the input image is also segmented at different scales, by using SLIC algorithm. These superpixel segmentation maps with different scales are used as the input of our multiscale fuzzy broad learning system. In this phase, saliency maps with different scales are to be generated, respectively, Assume that k different scales are used, we can denote these generated initial saliency maps as $I^1_{scale}, I^2_{scale}, ..., I^k_{scale}$. The overall process of generating an initial saliency map is shown in Algorithm 3.

Algorithm 3 Generate the Initial Saliency Map

Input: Input X, numbers of fuzzy rules k^i, fuzzy subsystems n, enhancement nodes E_j and enhancement node groups p.
Output: The initial saliency map

```
Set σ^i_kt = 1, and initialize the coefficient α^i_kt in [0,1];
Initialize k in k-means algorithm with k^i clustering centers;
while i <= n do
    Initialize the gaussian member function with σ^i_kt = 1 and k clustering centers.
    for s = 1; s ≤ N do
        Calculate Z_si using Eq. 5;
        Calculate F_si using Eq. 7;
    end for
    Obtain Z_i using Eq. 10;
    Calculate F_i using Eq. 8;
    i = i + 1;
end while
Obtain Z^n using Eq. 11;
for j = 1; j ≤ p do
    Randomly generate W_j and β_j;
    Calculate H_j = φ_j(Z^n W_j + β_j);
end for
Obtain all enhancement node groups H^p using Eq. 12;
Obtain F^n using Eq. 9;
Calculate Y using Eq. 14;
Obtain the corresponding output image; //i.e., an initial saliency map
return the initial saliency map;
```

Then, we can fuse these initial saliency maps as follows:

$$I = I^1_{scale} \otimes I^2_{scale} \otimes ... \otimes I^k_{scale} \tag{24}$$

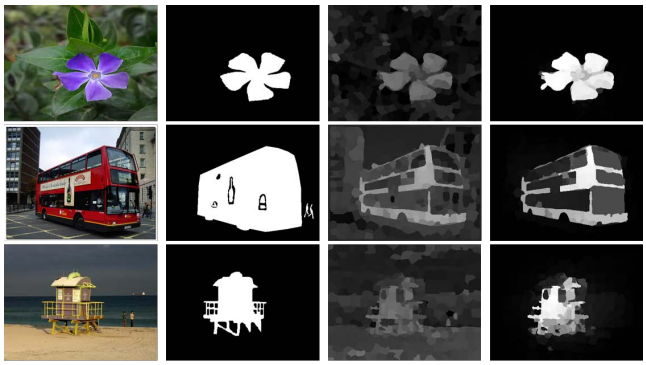
Fig. 5 shows the rationale of multiscale fusion process. Furthermore, we observe that the contrast between foreground and background in the fused image I could be not obvious, as shown in Fig. 6(c). To further improve the quality of saliency map I, we use the label propagation algorithm (LPA) [33] as the post-processing, which can strengthen the contrast between foreground and background, as shown in Fig. 6(d).

4. EXPERIMENTS

In this section, we first describe our experimental setup (Section 4.1), and then cover the experimental results and discussions (Section 4.2 $\sim$ 4.3).

TABLE 1: The training process on the MSRA-5000 dataset with incremental learning.

N_{fs}	N_{en}	MAE	Additional Training Time (s)	Accumulative Training Time (s)	Additional Test Time (s)	Accumulative Test Time (s)
10	100	0.29	2.3325	2.3325	0.3643	0.3643
10 $\longrightarrow$ 20	100 $\longrightarrow$ 300	0.23	2.1211	4.4454	0.2611	0.6254
20 $\longrightarrow$ 30	300 $\longrightarrow$ 500	0.19	2.8406	7.2942	0.3525	0.9779
30 $\longrightarrow$ 40	500 $\longrightarrow$ 700	0.16	3.9348	11.2290	0.4688	1.4466
40 $\longrightarrow$ 50	700 $\longrightarrow$ 900	0.14	5.0429	16.2719	0.3554	1.8020
50 $\longrightarrow$ 60	900 $\longrightarrow$ 1100	0.14	5.9883	22.4602	0.4330	2.2350
60 $\longrightarrow$ 70	1100 $\longrightarrow$ 1300	0.13	7.0116	29.4718	0.4596	2.6946
70 $\longrightarrow$ 80	1300 $\longrightarrow$ 1500	0.13	9.1372	38.6090	0.4121	3.1067
80 $\longrightarrow$ 90	1500 $\longrightarrow$ 1700	0.13	10.5632	49.1722	0.3874	3.4941
90 $\longrightarrow$ 100	1700 $\longrightarrow$ 1900	0.14	12.2455	61.4177	0.4531	3.9472



(a)Source (b)GT (c)Without LPA (d)With LPA

FIGURE 6: Visual comparison of saliency detection results with and without LPA post-processing.

4.1. Experimental Setup

4.1.1. Datasets

Like most neural network models, training our model also requires a large amount of data. We divided MSRA-5000 dataset [28] into 4,000 training images and 1,000 test images. To evaluate our model, we also test our solution on several other publicly available datasets. All these datasets contain the corresponding pixel-level annotations. Next, we describe briefly them.

- *MSRA-5000* [28]. This is a large dataset containing 5,000 images with rich contents. It is widely used in salient object detection. Most of images only have a single object foreground; the background and foreground are quite different in color, and the background is not very complex.
- *PASCAL-S* [40]. This dataset was selected from the validation set of the PASCAL VOC 2012 segmentation challenge. It contains 850 natural images with a fairly complex background, and it often has more than one foreground object for most of images. It is one of the most complex datasets in the field of saliency detection.
- *ECSSD* [21]. This dataset is also one of the complex datasets containing complex scenes. It contains 1,000 complex images from the Internet.
- *DUT-OMRON* [19]. This is a large dataset with 5,166 complex images. It is a very challenging dataset for saliency detection.
- *HKU-IS* [41]. This dataset contains 4,447 natural images. The complexity of the image scenes is not very high, but most of images contain multiple foreground objects.
- *SOD* [21]. This dataset contains 300 complex images from the Internet, most of which have little difference in color between the foreground and background, and so it is a very challenging dataset.

4.1.2. Evaluation Metrics

In order to evaluate the proposed algorithm, we adopted several widely-used evaluation metrics: (i) precision-recall (PR) curves, (ii) F-measure, and (iii) mean absolute error (MAE). Additionally, we also report precision (P) and recall (R), respectively. We binarized the saliency maps generated by our solution to obtain the binary result Γ, and calculate the precision and recall values through the following formulas:

$$P = \frac{|\Gamma \cap GT|}{|\Gamma|}, R = \frac{|\Gamma \cap GT|}{|GT|}. \tag{25}$$

In addition, F-measure is the weighted harmonic average of P and R. It is computed as:

$$F_\beta = \frac{(1+\beta^2)\cdot P \cdot R}{\beta^2 \cdot P + R} \tag{26}$$

Following [42, 43, 44, 21], we set β^2 to 0.3. The MAE is computed as:

$$MAE = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} |\overline{S}(x,y) - \overline{GT}(x,y)| \tag{27}$$

where W and H are the width and height of the saliency map, respectively; $S(x,y)$ is the saliency value of the pixel point (x,y), $GT(x,y)$ is the ground truth value corresponding to the pixel point (x,y).

4.1.3. Implementation Details

We used MATLAB R2016a to implement our solution. Our experiments were conducted on a PC with 2.90GHz

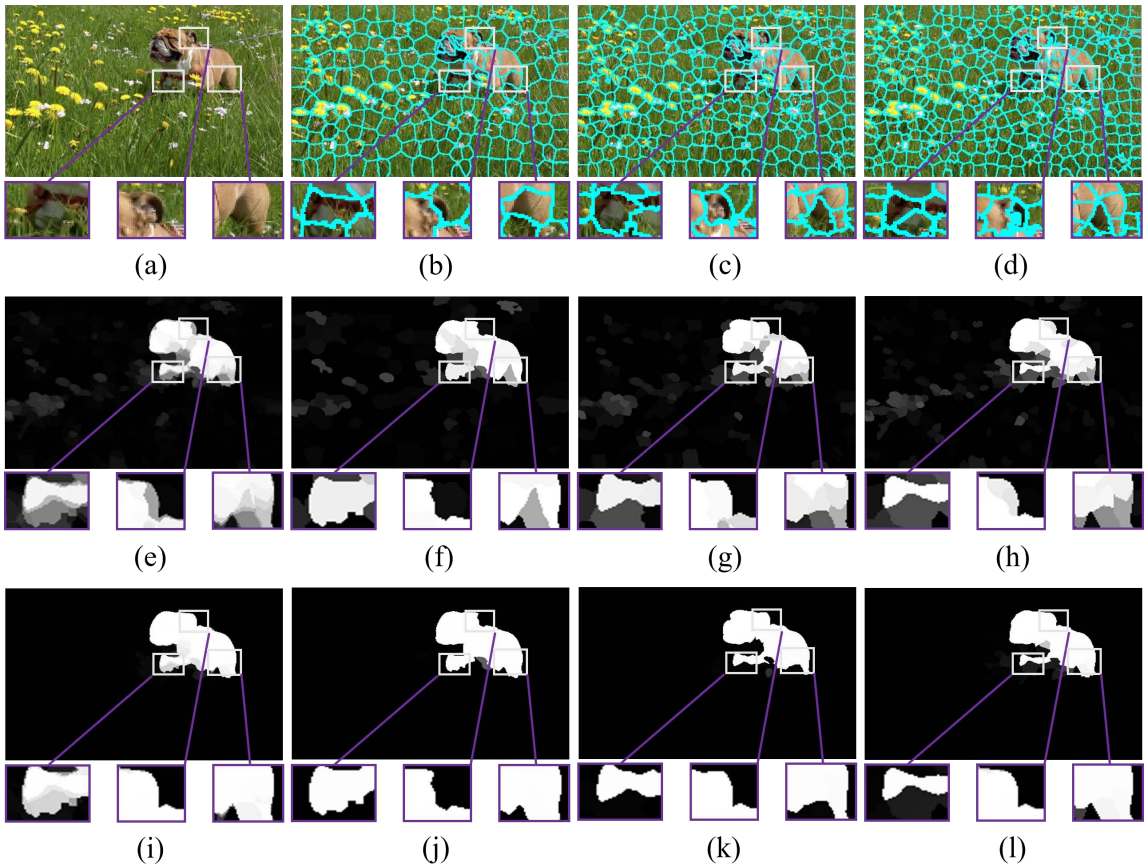


FIGURE 7: The results of multiscale segmentation maps, initial saliency maps, and final saliency maps. The first line: (a) Source image; (b) image segmented by $scale = 200$; (c) image segmented by $scale = 300$; (d) image segmented by $scale = 400$. The second line: (e) initial saliency map obtained by fusing the results in (f),(g), and (h); while (f), (g), and (h) are initial saliency maps with $scale = 200, 300,$ 400, respectively. The third line: (i), (j), (k), and (l) are final saliency maps obtained by using LPA based on the results in (e), (f), (g), and (h). The three small maps attached to the bottom of each map are three regions (i.e., dog's foreleg, ear, and hind leg) to be analyzed.

Intel i7-7820HK CPU and 32GB memory. We generated randomly the value of coefficient α_{kt}^{i} in $[0,1]$, we set $\sigma_{kt}^{i} = 1$, $\kappa = 10$, and $\lambda = 10^{-6}$. The connection coefficients W_j and β_j are randomly generated in $[-1,1]$, and we used "sigmoid" as the nonlinear function to activate the enhancement nodes. We applied the SLIC algorithm to three different scales ($scale1 = 200$, $scale2 = 300$, $scale3 = 400$). We compared our proposed solution with both low-level information based saliency detection algorithms and also deep learning based saliency detection algorithms. For some algorithms (e.g., GMR [19] and GC [45]), we used the implementation provided by authors. As for algorithms without publicly available implementations, we directly compared the results presented in their papers.

4.2. Experimental Results

Let N_{fr}, N_{fs}, and N_{en} be the numbers of fuzzy rules, fuzzy subsystems, and enhancement nodes, respectively. At the beginning, we set them to 5, 10, and 100 respectively, in order to train our model. Yet, we found that the accuracy of the trained model was insufficient. Then, we adjusted N_{fr} ranging from 5 to 30, N_{fs} ranging from 10 to 100, and N_{en} ranging from 100 to 1900. To illustrate, Tab. 1 shows our incremental learning process when $N_{fr} = 25$. The results show that the model is favorable when $N_{fr} = 25$, $N_{fs} = 70$, and $N_{en} = 1300$. In the rest of experiments, we used such a setting, unless stated otherwise.

Fig. 7 compares the detailed results of multiscale segmentation maps, their initial saliency maps, and final saliency maps. As shown in Fig. 7 (a), the puppy's foreleg, ear and hind leg are the regions that are difficult to be correctly judged as salient regions. In Fig. 7 (b), there are only 200 superpixel blocks, and so the foreleg, ear and hind leg of the puppy are not well segmented. This incurs that the generated saliency map cannot correctly classify these regions (see Fig. 7 (f)). Although these regions are well segmented in Fig. 7 (c), the segmented superpixel blocks still contain some background regions. This leads to the misjudgment of the foreground regions connected to the background as the background regions (see Fig. 7 (g); it shows that the regions under the foreleg and hind leg of the puppy are dark in color). However, Fig. 7 (d) over-segments these regions, resulting that the saliency results of

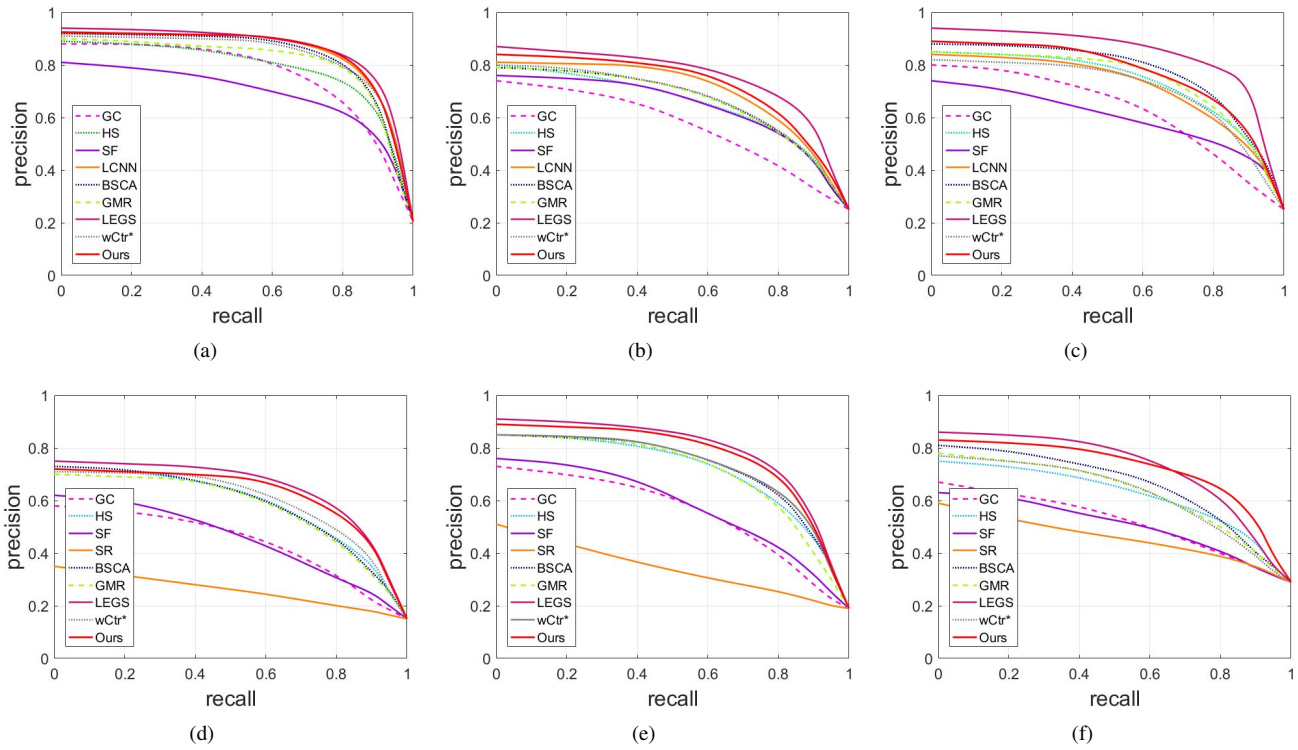


FIGURE 8: PR curves of different algorithms. (a) MSRA-5000; (b) PASCAL-S; (c) ECSSD; (d) DUT-OMRON; (e) HKU-IS; (f) SOD.

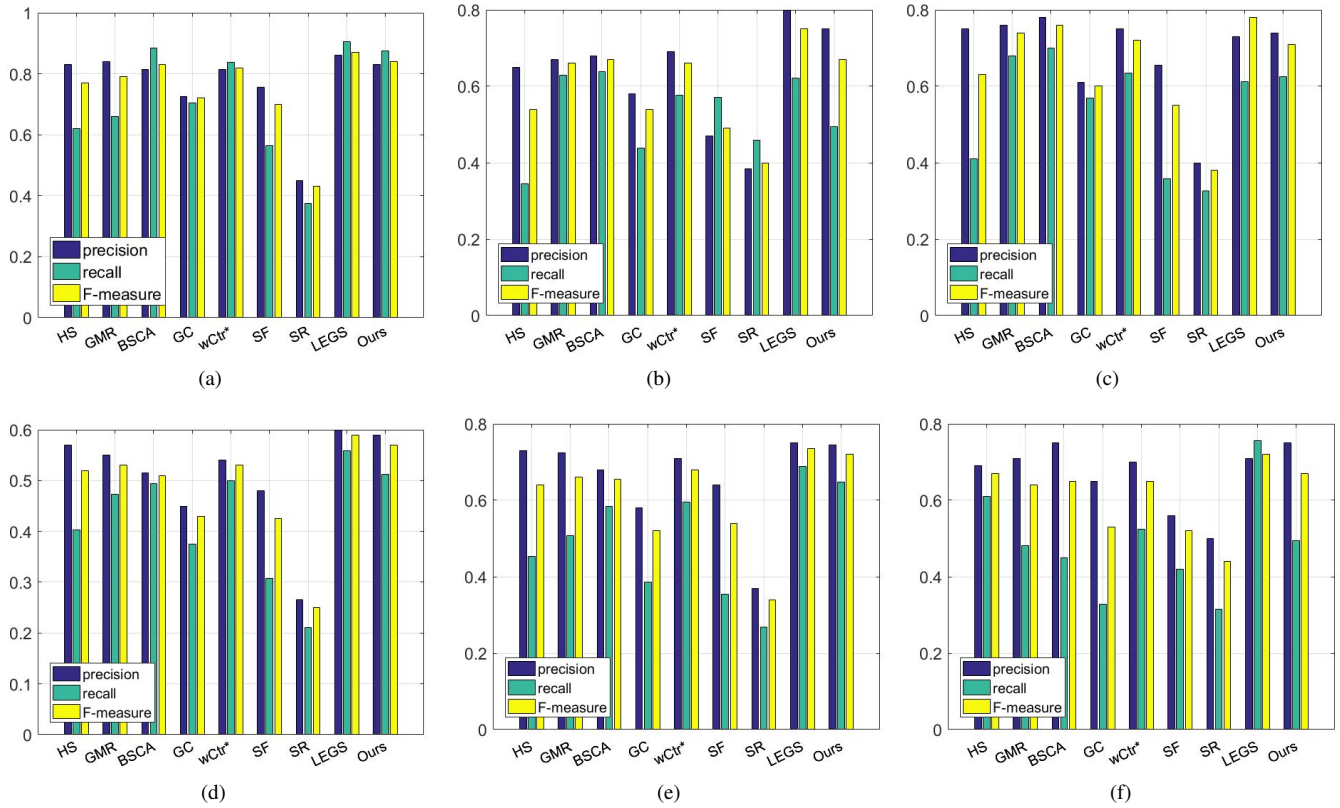
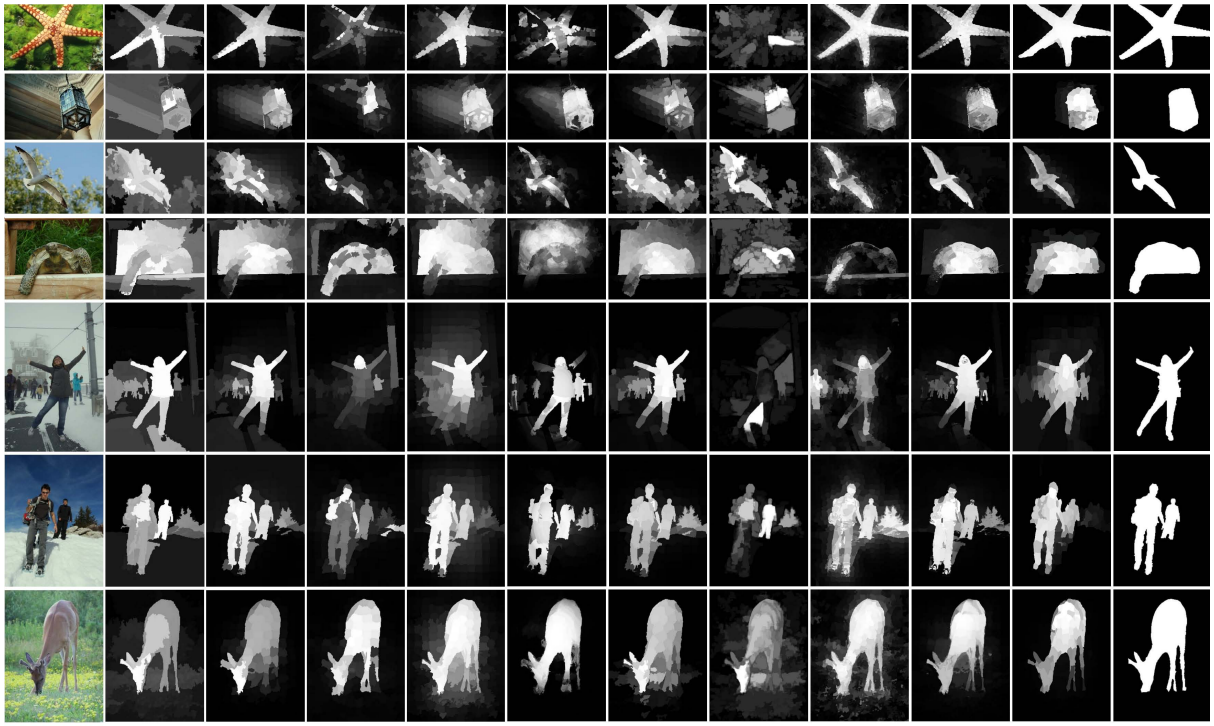


FIGURE 9: Overall performance. (a) MSRA-5000; (b) PASCAL-S; (c) ECSSD; (d) DUT-OMRON; (e) HKU-IS; (f) SOD.

TABLE 2: Comparison with existing algorithms. The comparison measures include F-measure (larger is better) and MAE(smaller is better). The top three for the best results have been highlighted in blue, red and green.

Model	MSRA-5000		PASCAL-S		ECSSD		DUT-OMRON		HKU-IS		SOD	
	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE	F_β	MAE
HS	0.77	0.16	0.54	0.25	0.63	0.23	0.52	0.23	0.64	0.21	**0.67**	0.28
GMR	0.79	0.13	**0.66**	0.22	**0.74**	0.19	0.53	0.19	0.66	0.17	0.64	0.26
HPS	0.71	0.21	0.52	0.26	0.60	0.25	-	-	-	-	-	-
BSCA	0.83	0.13	**0.67**	0.22	**0.76**	0.18	0.51	0.19	0.65	0.17	0.65	0.25
GC	0.72	0.16	0.54	0.27	0.60	0.23	0.43	0.22	0.52	0.21	0.53	0.28
BMS	0.75	0.16	0.58	0.21	0.62	0.22	-	-	-	-	-	-
HCT	0.77	0.14	0.54	0.23	0.64	0.20	-	-	-	-	-	-
wCtr*	0.82	0.11	**0.66**	0.20	0.72	0.17	0.53	**0.14**	**0.68**	**0.14**	0.65	**0.23**
PISA	**0.84**	0.11	**0.66**	0.20	**0.76**	**0.15**	0.54	**0.14**	**0.72**	**0.13**	**0.66**	**0.22**
SF	0.70	0.17	0.49	0.24	0.55	0.22	0.43	0.15	0.54	0.17	0.52	0.27
SR	0.43	0.22	0.40	0.26	0.38	0.26	0.25	0.18	0.34	0.22	0.44	0.30
OMSC	**0.831**	**0.091**	0.64	**0.181**	0.709	**0.15**	**0.60**	**0.126**	-	-	-	-
LEGS	**0.87**	**0.08**	**0.75**	**0.16**	**0.78**	**0.12**	**0.59**	**0.13**	**0.74**	**0.12**	**0.72**	**0.20**
LCNN	0.79	0.12	0.65	**0.17**	0.71	**0.16**	-	-	-	-	-	-
Ours	**0.84**	**0.10**	**0.67**	**0.17**	**0.71**	**0.17**	**0.57**	**0.14**	**0.72**	**0.16**	**0.67**	**0.22**



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l)

FIGURE 10: Visual comparison of saliency maps. From left to right: (a) Source Image; (b) HS [21]; (c) GMR [19]; (d) HPS [46]; (e) MK [47]; (f) DS [48]; (g) wCtr* [23]; (h) BMS [49]; (i) HCT [50]; (j) LCNN [26]; (k) Our approach; (l) The ground truth (GT)

these regions do not seem to be a whole region (see Fig. 7 (h)). By fusing the initial saliency maps generated by three scales, these small and complex regions of the images can be correctly classified as salient regions (see Fig. 7 (e)). In other words, the saliency map with fusion compensates for the shortcomings of saliency maps generated at different scales. Furthermore, by applying the post-processing on Fig. 7 (e), the final saliency map (see Fig. 7 (i)) is improved. This demonstrates the effectiveness of the post-processing operation. This finding is consistent with that presented in Fig. 6. Meanwhile, Fig. 7 (i) is better than other several saliency maps (e.g., Figs. 7 (j), (k), (l)). This further reflects the effectiveness of multiscale segmentation.

Table 2 makes a quantitative comparison with existing

TABLE 3: The infer time by existing algorithms

Method	GC[45]	HS[21]	GMR[19]	BSCA[35]	HCT[50]	wCtr*[23]	NLDF[25]	Amulet[15]	BMS[49]	LEGS[43]	LCNN[26]
Time(s)	0.037	0.528	0.149	0.420	0.017	0.250	0.080	0.063	0.575	1.550	0.188

algorithms including HS [21], GMR [19], HPS [46], BSCA [35], GC [45], BMS [49], HCT [50], wCtr* [23], PISA [51], SF [52], SR [53], OMSC [54], LEGS [43], LCNN [26]. The latter two methods employ deep neural networks. We can see that, on MSRA-5000, PASCAL-S and DUT-OMRON datsets our method can achieve a higher F-measure value and a lower MAE value, compared against (almost) all these low-level information based models. These results demonstrate the superiority of our proposed solution. On the other hand, the performance of our solution is close to deep learning based models (see the last three lines in this table), while our model is significantly faster than deep learning based models in terms of both training and test time, as demonstrated later. Further, we can understand that our model has a good generalization ability, since we trained our model using the MSRA-5000 dataset, while the trained model can work well for other datasets, as shown in this table.

Fig. 8 compares the PR curves of several existing methods including HS [21], GMR [19], BSCA [35], GC [45], wCtr* [23], SF [52], LEGS [43] and LCNN [26]. It can be seen that among these methods, the performance of our solution is ranked top-2. Furthermore, Fig. 9 also reports the precision, recall and F-measure values of these methods. In general, our solution performs well on all these datasets.

Fig. 10 makes a qualitative comparison with several existing algorithms. We can see that, (i) when the foreground object is in the middle of the image (e.g., lines 1 and 7), our approach can effectively detect the foreground region of the image. (ii) When the object is at the edge of the image (lines 2, 3 and 6), our approach can also effectively detect the foreground region. (iii) When background objects' positions are close to the foreground objects and their colors are similar to the foreground objects, our approach has a lower false detection rate (e.g., line 5). (iv) When the foreground object overlaps with the background object and the color difference is not obvious (line 4), our approach can still accurately detect the foreground object. In general, for almost all these scenes, our method can always effectively detect the foreground object and generate the saliency map similar to the ground truth. These results further demonstrate the feasibility and effectiveness of our solution.

4.3. Discussion

To generate 1,000 saliency maps, our solution used 1,820 seconds for preprocessing, 2.71 seconds for computation on the neural networks, and 14.26 seconds for fusing and optimization. Excluding the preprocessing time, the infer time of our solution is about 0.017 seconds for a single image on average. Table 3 summarizes the results of some existing algorithms. It can be seen that, the infer time of our solution is competitive. Particularly, we observed that, most of deep learning based methods take a long time to train their models (e.g., GBR [55] consumes 12 hours on an NVIDIA GTX-1080Ti GPU and an Intel E5-2630 CPU processor, UCF [56] consumes 23 hours on an NVIDIA Titan X GPU and an i7-4790 CPU, NLDF [25] consumes about 9 hours on NVIDIA Tian X GPU, Amulet [15] consumes about 16 hours on a NVIDIA Titan X GPU and an i7-4790 CPU). In contrast, training our model on a cheap computer (2.90GHz Intel i7-7820HK CPU and 32GB memory) used about 7,310 seconds (in which preprocessing images used 7,280 seconds, while the training process used about 30 seconds); its training efficiency is significantly faster than that of deep learning based models. Another potential benefit of our model is that, increasing the number of nodes or new data can only increase the extra training time, unlike deep learning based models that need to re-train their models starting-from-scratch.

5. CONCLUSION

In this paper, we have proposed a novel saliency detection algorithm. The central idea of our method is to segment the image into different scales, and then the extracted features are fed to the fuzzy broad learning system for training. We have conducted extensive experiments based on publicly available datasets to evaluate the performance of our proposed method. The experimental results have demonstrated that our method can outperform most of low-level information based saliency detection algorithms, and its performance is close to some deep learning based saliency detection algorithms. Particularly, our method is significantly faster than most of deep learning based saliency detection algorithms, in terms of training and inferring time. In the future, we would like to develop methods to further speedup the preprocessing efficiency, and to further improve the quality of generating saliency maps.

ACKNOWLEDGEMENTS

We thank the editors and anonymous reviewers very much for their efforts in evaluating our manuscript. This work was partially supported by the National Key R&D Program of China (2018YFC0810204, 2018YFB1004400), the NSFC (U1811264, 61502220, 61572326, 61775139, 61772164, 61872242, 61972425), the Key R&D Program of Guangdong Province (2018B010107005, 2019B010120001), and Shanghai Science and Technology Innovation Action Plan Project (16111107502, 17511107203), and the program for tackling key problems in Henan Science and Technology (172102310636).

REFERENCES

[1] Mahadevan, V. and Vasconcelos, N. (2009) Saliency-based discriminant tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1007–1013.

[2] Zhu, Y., Urtasun, R., Salakhutdinov, R., and Fidler, S. (2015) segdeepm: Exploiting segmentation and context in deep neural networks for object detection. *CVPR*, pp. 4703–4711.

[3] Chopin, J., Laga, H., and Miklavcic, S. J. (2016) The influence of object shape on the convergence of active contour models for image segmentation. *Comput. J.*, **59**, 603–615.

[4] Chen, L. C., Yang, Y., Wang, J., Xu, W., and Yuille, A. L. (2016) Attention to scale: Scale-aware semantic image segmentation. *Computer Vision and Pattern Recognition*, pp. 3640–3649.

[5] Fendri, E., Frikha, M., and Hammami, M. (2017) Adaptive person re-identification based on visible salient body parts in large camera network. *Comput. J.*, **60**, 1590–1608.

[6] Xu, J., Guo, X., Tu, Q., Li, C., and Men, A. (2015) A novel video saliency map detection model in compressed domain. *34th IEEE Military Communications Conference, MILCOM 2015, Tampa, FL, USA, October 26-28, 2015*, pp. 157–162.

[7] Srivastava, P. and Khare, A. (2018) Content-based image retrieval using local binary curvelet co-occurrence pattern - A multiresolution technique. *Comput. J.*, **61**, 369–385.

[8] Huang, J., Yang, X., Zhang, R., Lu, F., and Fang, X. (2010) Integrating visual saliency and consistency for re-ranking image search results. *IEEE Transactions on Multimedia*, **13**, 653–661.

[9] Hongyang, L., Huchuan, L., Zhe, L., Xiaohui, S., and Brian, P. (2015) Inner and inter label propagation: salient object detection in the wild. *IEEE Trans Image Process*, **24**, 3176–3186.

[10] Filali, I., Allili, M. S., and Benblidia, N. (2016) Multi-scale salient object detection using graph ranking and global–local saliency refinement. *Signal Processing: Image Communication*, **47**, 380–401.

[11] Nie, M., Wang, Z., Yin, J., and Yao, B. (2019) Reachable region query and its applications. *Inf. Sci.*, **476**, 95–105.

[12] Yulin, X., Huchuan, L., and Ming-Hsuan, Y. (2013) Bayesian saliency via low and mid level cues. *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, **22**, 1689–1698.

[13] Wang, Z., Ma, L., Lin, X., and Zhong, H. (2018) Saliency detection via multi-center convex hull prior. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 1867–1871.

[14] Dodge, S. F. and Karam, L. J. (2018) Visual saliency prediction using a mixture of deep neural networks. *IEEE Trans. Image Processing*, **27**, 4080–4090.

[15] Zhang, P., Wang, D., Lu, H., Wang, H., and Xiang, R. (2017) Amulet: Aggregating multi-level convolutional features for salient object detection. *IEEE International Conference on Computer Vision*, pp. 202–211.

[16] Itti, L., Koch, C., and Niebur, E. (1998) A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, **20**, 1254–1259.

[17] Zhang, M., Pang, Y., Wu, Y., Du, Y., Sun, H., and Zhang, K. (2018) Saliency detection via local structure propagation. *J. Visual Communication and Image Representation*, **52**, 131–142.

[18] Wang, H., Liu, G., Ke, H., Chen, Z., and Li, H. (2018) Saliency region detection method based on background and spatial position. *IJPRAI*, **32**, 1–20.

[19] Yang, C., Zhang, L., Lu, H., Xiang, R., and Yang, M. H. (2013) Saliency detection via graph-based manifold ranking. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3166–3173.

[20] Wei, Y., Wen, F., Zhu, W., and Sun, J. (2012) Geodesic saliency using background priors. *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part III*, pp. 29–42.

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

[22] Cheng, M., Zhang, G., Mitra, N. J., Huang, X., and Hu, S. (2011) Global contrast based salient region detection. *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pp. 409–416.

[23] Zhu, W., Liang, S., Wei, Y., and Sun, J. (2014) Saliency optimization from robust background detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2814–2821.

[24] Li, X., Zhao, L., Wei, L., Yang, M.-H., Wu, F., Zhuang, Y., Ling, H., and Wang, J. (2016) Deepsaliency: Multi-task deep neural network model for salient object detection. *IEEE Transactions on Image Processing*, **25**, 3919–3930.

[25] Luo, Z., Mishra, A. K., Achkar, A., Eichel, J. A., Li, S., and Jodoin, P. (2017) Non-local deep features for salient object detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 6593–6601.

[26] Li, H., Chen, J., Lu, H., and Chi, Z. (2017) Cnn for saliency detection with low-level feature integration. *Neurocomputing*, **226**, 212–220.

[27] Lee, G., Tai, Y., and Kim, J. (2016) Deep saliency with encoded low level distance map and high level features. *CVPR*, pp. 660–668.

[28] Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., and Shum, H.-Y. (2011) Learning to detect a salient object. *IEEE Transactions on Pattern analysis and machine intelligence*, **33**, 353–367.

[29] Li, G. and Yu, Y. (2016) Visual saliency detection based on multiscale deep CNN features. *IEEE Trans. Image Processing*, **25**, 5012–5024.

[30] Radhakrishna, A., Appu, S., Kevin, S., Aurelien, L., Pascal, F., and Sabine, S. (2012) Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **34**, 2274–2282.

[31] Ojala, T., Pietikäinen, M., and Harwood, D. (1994) Performance evaluation of texture measures with classification based on kullback discrimination of distributions. *12th IAPR International Conference on Pattern Recognition, Conference A: Computer Vision & Image Processing, ICPR 1994, Jerusalem, Israel, 9-13 October, 1994, Volume 1*, pp. 582–585.

[32] Dragan, K. and Emil, L. (2004) Identification of complex systems based on neural and takagi-sugeno fuzzy model. *IEEE Trans Syst Man Cybern B Cybern*, **34**, 272–282.

[33] Zhang, T., Zhou, Y., Huo, S., and Hou, C. (2017) Label propagation based saliency detection via graph design. *Image*

Processing (ICIP), 2017 IEEE International Conference on, pp. 460–464.

[34] Zhang, J., Zhang, T., Dai, Y., Harandi, M., and Hartley, R. I. (2018) Deep unsupervised saliency detection: A multiple noisy labeling perspective. *CVPR*, pp. 9029–9038.

[35] Qin, Y., Lu, H., Xu, Y., and Wang, H. (2015) Saliency detection via cellular automata. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 110–119.

[36] Zhang, J., Xia, C., Zhang, C., Cui, L., Fu, Y., and Yu, P. S. (2017) BL-MNE: emerging heterogeneous social network embedding through broad learning with aligned autoencoder. *ICDM*, pp. 605–614.

[37] Chen, C. and Liu, Z. (2017) Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. *IEEE Transactions on Neural Networks & Learning Systems*, **29**, 10–24.

[38] Cao, B., Mao, M., Viidu, S., and Yu, P. S. (2017) Hitfraud: A broad learning approach for collective fraud detection in heterogeneous information networks. *ICDM*, pp. 769–774.

[39] Feng, S. and Chen, C. L. P. Fuzzy broad learning system: A novel neuro-fuzzy model for regression and classification. *IEEE Transactions on Cybernetics*, **PP**, 1–11.

[40] Li, Y., Hou, X., Koch, C., Rehg, J. M., and Yuille, A. L. (2014) The secrets of salient object segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 280–287.

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

[42] Cheng, M.-M., Mitra, N. J., Huang, X., Torr, P. H., and Hu, S.-M. (2015) Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**, 569–582.

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

[44] Borji, A., Cheng, M.-M., Jiang, H., and Li, J. (2015) Salient object detection: A benchmark. *IEEE transactions on image processing*, **24**, 5706–5722.

[45] Cheng, M.-M., Warrell, J., Lin, W.-Y., Zheng, S., Vineet, V., and Crook, N. (2013) Efficient salient region detection with soft image abstraction. *Proceedings of the IEEE International Conference on Computer vision*, pp. 1529–1536.

[46] Li, X., Li, Y., Shen, C., Dick, A., and Van Den Hengel, A. (2013) Contextual hypergraph modeling for salient object detection. *Proceedings of the IEEE international conference on computer vision*, pp. 3328–3335.

[47] Jiang, B., Zhang, L., Lu, H., Yang, C., and Yang, M.-H. (2013) Saliency detection via absorbing markov chain. *Proceedings of the IEEE international conference on computer vision*, pp. 1665–1672.

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

[49] Zhang, J. and Sclaroff, S. (2013) Saliency detection: A boolean map approach. *Proceedings of the IEEE international conference on computer vision*, pp. 153–160.

[50] Kim, J., Han, D., Tai, Y.-W., and Kim, J. (2014) Salient region detection via high-dimensional color transform. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 883–890.

[51] Wang, K., Lin, L., Lu, J., Li, C., and Shi, K. (2015) Pisa: Pixelwise image saliency by aggregating complementary appearance contrast measures with edge-preserving coherence. *IEEE Transactions on Image Processing*, **24**, 3019–3033.

[52] Perazzi, F., Krähenbühl, P., Pritch, Y., and Hornung, A. (2012) Saliency filters: Contrast based filtering for salient region detection. *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pp. 733–740.

[53] Hou, X. and Zhang, L. (2007) Saliency detection: A spectral residual approach. *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), 18-23 June 2007, Minneapolis, Minnesota, USA*.

[54] Ji, Y., Zhang, H., Tseng, K., Chow, T. W. S., and Wu, Q. M. J. (2019) Graph model-based salient object detection using objectness and multiple saliency cues. *Neurocomputing*, **323**, 188–202.

[55] Tan, X., Zhu, H., Shao, Z., Hou, X., Hao, Y., and Ma, L. (2018) Saliency detection by deep network with boundary refinement and global context. *ICME*, pp. 1–6.

[56] Zhang, P., Wang, D., Lu, H., Wang, H., and Yin, B. (2017) Learning uncertain convolutional features for accurate saliency detection. *ICCV*, pp. 212–221.

[57] Zhai, Y. and Shah, M. (2006) Visual attention detection in video sequences using spatiotemporal cues. *ACM International Conference on Multimedia*, pp. 815–824.