A fast method for saliency detection by back-propagating a convolutional neural network and clamping its partial outputs

Conference Paper · May 2017

DOI: 10.1109/IJCNN.2017.7966307

2 authors, including:

Hui Jiang
York University
144 PUBLICATIONS **2,926** CITATIONS

A Fast Method for Saliency Detection by Back-Propagating A Convolutional Neural Network and Clamping Its Partial Outputs

Hengyue Pan
Department of Electrical Engineering and Computer Science
York University
Toronto, Ontario, Canada M3J 1P3
Email: panhy@cse.yorku.ca

Hui Jiang
Department of Electrical Engineering and Computer Science
York University
Toronto, Ontario, Canada M3J 1P3
Email: hj@cse.yorku.ca

***Abstract*—In this paper, we propose a fast deep learning method for object saliency detection using convolutional neural networks. In our approach, we use a gradient descent method to iteratively modify the input images based on the pixel-wise gradients to reduce a pre-defined cost function, which is defined to measure the class-specific objectness and clamp the class-irrelevant outputs to maintain image background. The pixel-wise gradients can be efficiently computed using the back-propagation algorithm. We further apply SLIC superpixels and LAB color based low level saliency features to smooth and refine the gradients. Our methods are quite computationally efficient, much faster than other state-of-the-art deep learning based saliency methods. Experimental results on two benchmark tasks, namely Pascal VOC 2012 and MSRA10k, have shown that our proposed methods can generate high-quality salience maps, at least comparable with many slow and complicated deep learning methods. Comparing with the pure low-level methods, our approach excels in handling many difficult images, which contain complex background, highly-variable salient objects, multiple objects, and/or very small salient objects.**

1. Introduction

In the past few years, deep convolutional neural networks (DCNNs) [1] have achieved the state-of-the-art performance in many computer vision tasks, starting from image recognition [2], [3], [4] and object localization [5] and more recently extending to object detection and semantic image segmentation [6], [7]. These successes are largely attributed to the capacity that large-scale DCNNs can effectively learn end-to-end from a large amount of labelled images in a supervised learning mode.

In this paper, we consider to apply the popular deep learning techniques to another computer vision problem, namely object saliency detection. The saliency detection attempts to locate the objects that have the most interests in an image, where human may also pay more attention [8]. The main goal of the saliency detection is to compute a saliency map that topographically represents the level of saliency for visual attention [9]. For each pixel in an image, the saliency map can provide how likely this pixel belongs to the salient objects [10]. Computing such saliency maps has recently raised a great amount of research interest [11]. The computed saliency maps have been shown to be beneficial to various vision tasks, such as image segmentation, object recognition and visual tracking [12]. The saliency detection has been extensively studied in computer vision, and a variety of methods have been proposed to generate the saliency maps for images. Under the assumption that the salient objects probably are the parts that significantly differ from their surroundings, most of the existing methods use low-level image features to detect saliency regions based on the criteria related to color contrast, rarity and symmetry of image patches [8], [10], [12], [13], [14], [15]. In some cases, the global topological cues may be leveraged to refine the perceptual saliency maps [9], [16]. In these methods, the saliency is normally measured based on different mathematical models, including decision theoretic models, Bayesian models, information theoretic models, graphical models, and spectral analysis models [11].

Different from the previous low level methods, we propose a novel deep learning method for the object saliency detection based on the powerful DCNNs. As shown in [2], [3], [4], relying on a pre-trained classification DCNN, we can achieve a fairly high accuracy in object category recognition for many real-world images. Even though DCNNs can recognize what kind of objects are contained in an image, it is not straightforward for them to precisely locate the recognized objects in the image. In [5], [6], [7], some rather complicated and time-consuming post-processing stages are needed to detect and locate the objects for semantic image segmentation. In [17], two DCNNs are applied to generate superpixel based global saliency features and local saliency features, which should be combined for the final saliency maps. In [18], it has proposed a DCNN back-propagation based saliency framework with a simple objective function. In [19], it has been further extended by introducing both original images and masked images as training data and applying image erosion and dilation to improve the raw saliency maps. These two papers are the most relevant to

the work in this paper.

In this work, we propose a much simpler and more computationally efficient method to generate a class-specific object saliency map directly from the classification DCNN model. In our approach, we use a gradient descent method to iteratively modify each input image based on the pixel-wise gradients to reduce a pre-defined cost function, which is defined to measure the class-specific objectness and clamp the class-irrelevant outputs to maintain image background. The gradients with respect to all image pixels can be efficiently computed using the back-propagation algorithm for DCNNs. After the back-propagation procedure, the discrepancy between the modified image and the original one is calculated as the raw saliency map for this image. The raw saliency maps are smoothed by using SLIC [20] superpixel maps and refined by using low level saliency features. Since we only need to run a small number of gradient descent iterations in the saliency detection, our methods are extremely computationally efficient (average processing time for one image in one GPU is around 1.22 seconds).

Experimental results on two databases, namely Pascal VOC 2012 [21] and MSRA10k [22], have shown that our proposed methods can generate high-quality salience maps, at least comparable with many slow and complicated deep learning methods. On the other hand, comparing with the traditional low-level methods, our approach excels on many difficult images, containing complex background, highly-variable salient objects, multiple objects, and/or very small objects.

2. Our Approach for Object Saliency Detection

In this section we will consider the main idea of our DCNN based saliency detection method, and also discuss how to smooth and refine the raw saliency map for better performance.

2.1. Backpropagating and partially clamping DCNNs to generate raw saliency maps

As we have known, DCNNs can automatically learn all sorts of features from a large amount of labelled images, and a well-trained DCNN can achieve a very good classification accuracy in recognizing objects in images. In this work, based on the idea of explanation vectors in [23], we argue that the classification DCNNs themselves may have learned enough features and information to generate good object saliency for the images. Extending a preliminary study in [18], we explore a novel method to generate the saliency maps directly from DCNNs. The key idea of our approaches is shown in Figure 1. After an input image is recognized by a DCNN as containing one particular object, if we can modify the input image in such a way that the DCNN no longer recognizes the object from it and meanwhile attempts to maintain image background as much as possible, the discrepancy between the modified image and the original one may serve as a good saliency map for the recognized object. In this paper, we propose to use a gradient descent method to iteratively modify the input image based on the pixel-wise gradients to reduce a cost function formulated in the output layer of the DCNN. The proposed cost function is defined to measure the class-specific objectness. The cost function is reduced under the constraint that all class-irrelevant DCNN outputs are clamped to the original values, which is fundamentally different with [18], which has not try to keep the class-irrelevant output values. The image is modified by the gradients computed by applying the back-propagation procedure all the way to the input layer. In this way, the underlying object may be erased from the image while the irrelevant background may be largely retained.

First of all, we simply train a regular DCNN for the image classification. After the DCNN is learned, we may apply our saliency detection method to generate the class-specific object saliency map. For each input image X, we firstly use the pre-trained classification DCNN to generate its class label, denoted as l, as in a normal classification step. Meanwhile, we obtain the DCNN outputs prior to the final softmax layer, denoted as $\{o_k \mid k = 1, \cdots, N\}$ (N is the number of different classes). Apparently, o_l achieves the maximum value (due to the image is recognized as l). Here, we assume that the DCNN output o_l is mainly relevant to the underlying object in the image while the remaining DCNN outputs $\{o_k \mid k \neq l\}$ are more relevant to the image background excluding the underlying object. Under this assumption, we propose a procedure to modify the image to reduce the l-th output of the DCNN as much as possible and meanwhile clamp the other outputs to their original values o_k. We further denote the output nodes (prior to softmax) of the DCNN in the saliency generation procedure as $\{a_i \mid i = 1, \cdots, N\}$. Therefore, for the image X, we attempt modify X to reduce the corresponding largest DCNN output, i.e. a_l, subject to the constraint that all remaining DCNN outputs are clamped to their initial values: $a_k = o_k (k = 1, \cdots, N$ and $k \neq l)$.

Next, we propose to cast the above constraints as penalty terms to construct the following cost function:

$$\mathcal{F}(X|l) = a_l + \frac{\gamma}{2} \sum_{k \neq l} (a_k - o_k)^2 \tag{1}$$

where γ is a hyperparameter to balance the contribution from the constraints. In this way, we have converted the original constrained optimization problem into an unconstrained problem, and the value of the objective function can be easily reduced by using gradient descent methods. This simple unbounded objective function works very well in practice and it results in equally good saliency maps as other more complicated bounded objective functions we have investigated (but not reported here).

Obviously, this cost function is constructed based on the assumption that the recognized l-th output of the DCNN, i.e. a_l, corresponds to the foreground area in the input image while the remaining outputs of DCNN are more relevant to the image background. Therefore, if we modify the image X to reduce the above cost function and hopefully the underlying object (belonging to class l) will be removed

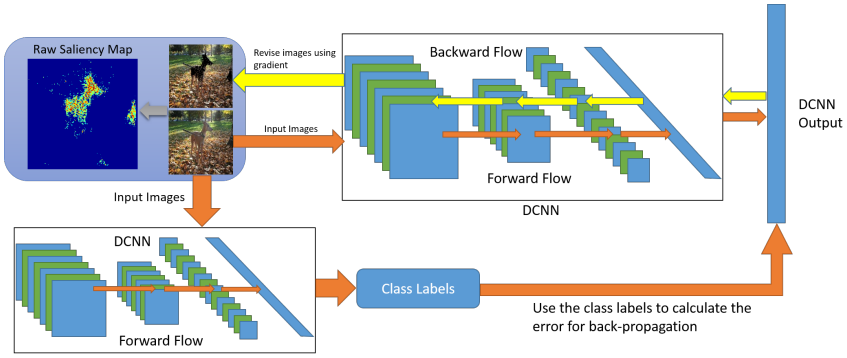

Figure 1. The proposed method to generate the object-specific saliency maps directly from DCNNs.

as the consequence due to that fact that a_l is reduced significantly, but the background remains largely unchanged due to the rest DCNN outputs are clamped in this procedure. In this paper, we propose to use an iterative gradient descent procedure to modify X as follows:

$$X^{(t+1)} \leftarrow X^{(t)} - \epsilon \cdot \max\left(\frac{\partial \mathcal{F}(X|l)}{\partial X}\Big|_{X=X^{(t)}}, 0\right) \quad (2)$$

where ϵ is the learning rate, and we floor all negative gradients in the gradient descent updates. We have good rationale for doing that. As we know, the values of all image pixels are non-negative in nature. If we want to reduce the DCNN output of a target class, conceptually speaking, we have two different ways: i) removing the underlying objects by cutting them out (technically subtracting positive values from image pixels); ii) covering the underlying objects by smearing them (technically adding positive values to image pixels). In order to correctly localize the objects by differentiating the original image and the modified one, it is clear that we prefer to use i) not ii) to reduce the DCNN output. This is the basic reason for us to floor negative gradients. Moreover, after flooring all negative gradients, we ensure the final difference images do not have negative values, which significantly facilitate the postprocessing.

We have observed in our experiments that the cost function $\mathcal{F}(X|l)$ can be significantly reduced by running only a small number of updates (typically 30-35 iterations) for each image, which guarantees the efficiency of the proposed method. This iterative updating procedure shows better performance than [18], which adopts linear approximation to the CNN objective function and uses the gradients to modify input images only once to generate saliency maps. As we know, DCNNs are highly nonlinear and it is beneficial to use multiple gradients to iteratively modify images by taking the nonlinearity into account.

We can easily compute the above gradients using the standard back-propagation algorithm. Based on the cost function $\mathcal{F}(X|l)$ in Eq. (1), we can derive the error signals

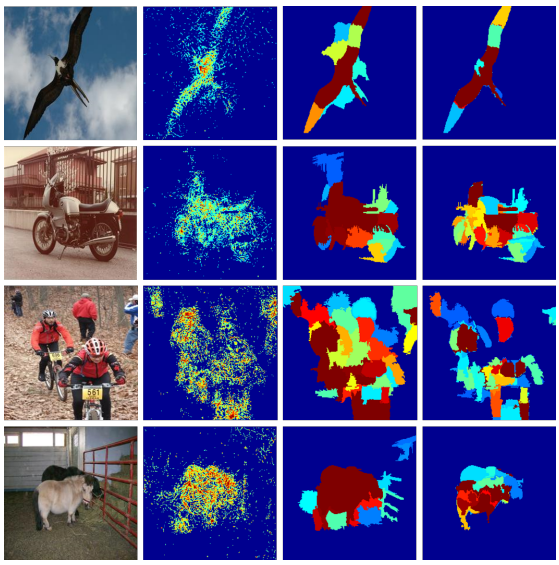

Figure 2. From left to right: original images, raw saliency maps, smoothed saliency maps and refined saliency maps

in the output layer, $e_i = \frac{\partial \mathcal{F}(X|l)}{\partial a_i}$ $(i = 1, \cdots, N)$, as follows:

$$e_i = \begin{cases} \gamma(a_i - o_i) & \text{if } i \neq l, \\ 1 & \text{if } i = l. \end{cases} \quad (3)$$

These error signals are back-propagated all the way to the input layer to derive the above gradient, $\frac{\partial \mathcal{F}(X|l)}{\partial X}$, for saliency detection. Notice that during the above process all weights of the DCNN should keep unchange.

At the end of the gradient descent updates, the raw object saliency map $\mathbf{S}$ is computed as the difference between the modified image and the original one, i.e. $X^{(0)} - X^{(T)}$. For colour images, we average the differences over the RGB channels to obtain a pixel-wise raw saliency map, which is then normalized to be of unit norm. After that, we can apply a simple threshold to filter out some weak signals (in most situations they are corresponding to background) of the raw saliency maps (see the second column in Figure 2).

2.2. SLIC based saliency map smoothing

In practice, we have found that the continuity of the above raw saliency map $\mathbf{S}$ is still not good enough in many cases. The main reason is that the DCNN outputs are not totally independent and their correlation is not considered in the above procedure. Roughly speaking, we have observed that most of the strong signals in the gradients are located in the saliency region. However, from Figure 2 we can see that some problems may still exist, such as background noises, blurred edges or small holes in the foreground. In order to further smooth the saliency maps, we use SLIC superpixels [20] to impose a continuity constraint that all image pixels located in a superpixel always have the same saliency value. More specifically, we firstly generate the superpixel maps of all test images (In our experiments we will spilt each test image into 75 superpixels, and the compact factor is set to 10). If i-th pixel in an image belongs to the jth superpixel P_j, then the smoothed saliency value can be calculated as Eq. (4) shows:

$$\bar{\mathbf{S}}_i = \frac{1}{N_j} \sum_{k \in P_j} \mathbf{S}_k \ \ (\forall i \in P_j) \tag{4}$$

Where N_j is the number of pixels in P_j, and we use $\bar{\mathbf{S}}$ to denote the smoothed saliency maps. We can again remove some weak signals in $\bar{\mathbf{S}}$. Obviously, comparing with $\mathbf{S}$, we can see that $\bar{\mathbf{S}}$ may fill holes in the saliency regions, sharpen the object edges, and also significantly reduce the isolated background noises (see the third column in Figure 2).

2.3. Refine saliency maps using low level features

In the last part, we have generated the smoothed saliency maps, which can provide much better performance than the original raw saliency maps. On top of that, we propose to introduce some constraints based on low-level features to further improve the quality of the saliency maps.

Based on the main idea of [14], we can generate low level saliency features for each test image in very short time. Firstly, we again apply the SLIC superpixel generation method in [20] to generate superpixel maps for the test images. Next, for one superpixel P_i in an image, we calculate its color feature C_i by averaging the LAB color value over its all pixels, and use the color feature to calculate its global color contrast GC_i as:

$$GC_i = \sum_j \parallel C_i - C_j \parallel_2^2 . \tag{5}$$

where $\| \cdot \|_2$ denotes the Euclidean distance.

Following [14], we can further calculate the color distribution maps and smooth the global color contrast maps as the raw low-level saliency maps, which is denoted as S_L. Moreover, S_L is applied to refine the smoothed saliency map $\bar{\mathbf{S}}$ that generated from the last step. Here, we normalize S_L between α and $1+\alpha$, where $0 < \alpha < 1$. The reason to use α is that the low level features contain a lot of errors, which may over-smooth some saliency values in the foreground of some images. By using α, we can prevent this refining procedure from removing some correct saliency regions in $\bar{\mathbf{S}}$. The refined saliency map $\hat{\mathbf{S}}$ can be generated as:

$$\hat{\mathbf{S}} = S_L \odot \bar{\mathbf{S}}. \tag{6}$$

where $\odot$ denotes the element-wise multiplication.

At the end, we further filter out some weak signals in $\hat{\mathbf{S}}$ and re-normalize it (see the fourth column in Figure 2). The entire algorithm to generate the final saliency maps is shown in **Algorithm 1**.

Algorithm 1 DCNN based Object Saliency Detection

Input: an input image X, DCNN, SLIC superpixel map P, low level saliency feature S_L;
Use DCNN to recognize the object label for X as l;
$X^{(0)} = X$;
for each epoch $t = 1$ **to** T **do**
  forward pass: compute the cost function $\mathcal{F}(X|l)$;
  backward pass: back-propagate to input layer to compute gradient: $\frac{\partial \mathcal{F}(X|l)}{\partial X}$;
  $X^{(t)} \leftarrow X^{(t-1)} - \epsilon \cdot \max\left(\frac{\partial \mathcal{F}(X|l)}{\partial X}, 0\right)$;
end for
Average over RGB: $\mathbf{S} = \frac{1}{3}\sum_{i=1}^{3}(X_i^{(0)} - X_i^{(T)})$;
Prune noises with a threshold θ: $\mathbf{S} = \max(\mathbf{S} - \theta, 0)$;
Normalize: $\mathbf{S} = \frac{\mathbf{S}}{\|\mathbf{S}\|}$;
Smoothing: using P to smooth $\mathbf{S}$ as $\bar{\mathbf{S}}$;
Prune noises again
Refine: $\hat{\mathbf{S}} = S_L \odot \bar{\mathbf{S}}$;
Prune noises and normalize again;
Output: the refined saliency map $\hat{\mathbf{S}}$;

3. Experiments

We select two benchmark databases to evaluate the performance of the proposed object saliency detection methods, namely Pascal VOC 2012 [21] and MSRA10k [22]. For Pascal VOC 2012, we use the 1449 validation images in its segmentation task as the test set, while for MSRA10k we directly use all $10,000$ images to do the test. Both databases provide the pixel-wise segmentation map (ground truth), thus we can easily measure the performances of different saliency algorithms. Notice that all images in the two databases should be re-scaled to 224-by-224 for our experiments.

As [12], for each saliency map, we vary the cutoff threshold from 0 to 255 to generate 256 precision and recall pairs, which are used to plot a PR-curve. Besides, we also use F_β to measure the performance for both saliency detection and segmentation, which is calculated based on precision $Prec$ and recall Rec values with a non-negative weight parameter β as follows [10]:

$$F_\beta = \frac{(1+\beta^2)Prec \times Rec}{\beta^2 Prec + Rec} \tag{7}$$

In this paper, we follow [12] to set $\beta^2 = 0.3$ to emphasize the importance of $Prec$. We may derive a sequence of

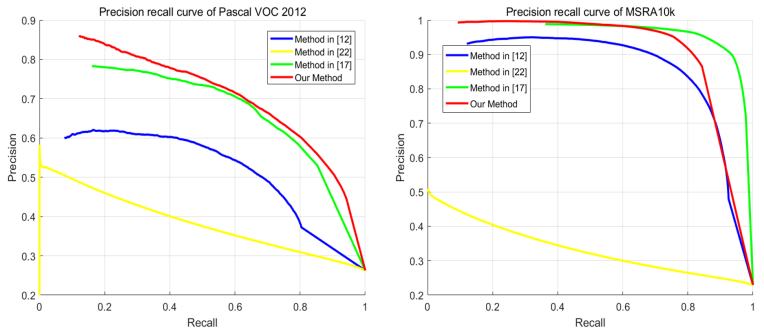

Figure 3. The PR-curves of different saliency methods on the Pascal VOC 2012 test set (Left) and MSRA10k (Right).

F_β values along the PR-curve for each saliency map and the largest one is selected as the performance measure (see [10]) [1].

3.1. Databases

Pascal VOC 2012 database [21] is a classical but still challenging image database that can be used for several vision tasks including image classification and saliency. This database currently contains 5717 training images and 5823 validation images with 20 labeled categories. However, among them, only 1449 validation images that include ground truth information are used to evaluate the performance in our image saliency tasks. Therefore, to expand the training set and improve the classification performance of the DCNN, we merge the original training set with the remaining 4374 validation images without ground truth to form a new training set, which has 10197 training samples. For images that are labelled to have more than one class of objects, we use the area of the labelled objects to measure their importance, and use the class of the largest object to label the images for our DCNN training process.

Unfortunately, the Pascal training set is still relatively small for DCNN training. Therefore, we have used a pre-trained DCNN for the ImageNet database, which contains 13 convolutional layers and 3 fully connected layers[2], as the initial network, and only use the above-mentioned training data to fine-tune this DCNN with MatConvNet in [25]. The top-1 error on the validation set of Pascal VOC 2012 is 16.7%. This classification error implies that the training sample size of Pascal VOC 2012 is still not enough for training deep convolutional networks well. However, as we will see, the proposed algorithms can still yield good performance for saliency detection. If we have more training data, we may expect even better saliency results.

MSRA10k [22] is another widely-used image saliency database, which is constructed based on Microsoft MSRA saliency database [8]. MSRA10k selects $10,000$ images from MSRA and includes pixel-wised salient objects information instead of bounding boxes, which make it suitable for our task. However, MSRA10k dose not include the corresponding training set and class labels of all images. Therefore, for MSRA10k, we directly use the DCNN *imagenet-vgg-verydeep-16* [24] (without any fine-tuning) to proceed our algorithm.

1. The codes of the proposed method can be downloaded via: https://github.com/mowangphy/cnn_based_saliency

2. We use the net *imagenet-vgg-verydeep-16* [24].

Methods	Region Contrast [12]	DCNN based Method [18]	Deep Saliency [17]	Our Method
Time	1.92s	0.03s	4.38s	1.22s

TABLE 1. THE TIME FOR PROCESSING ONE IMAGE OF DIFFERENT SALIENCY METHODS.

3.2. Saliency Results

In this part we will provide saliency detection results on the selected two databases. In the following, the PR-curves, F_β values and some sample images will be used to compare different methods.

3.2.1. Efficiency. We firstly consider the speed of our saliency method. Here we will not take the DCNN training time into account because for all of the experiments based on one database, we need only train DCNN once. We can even directly use the will trained DCNN for ImageNet classification for our method without any fine-tune, and the saliency results are also good. Our computing platform includes Intel Xeon E5-1650 CPU (6 cores), 64 GB memory and Nvidia Geforce TITAN X GPU (12 GB memory). The time consumption of processing one image of different algorithms are listed in Table 1.

From Table 1 we can learn that our method yields faster processing speed than [12] and [17]. Due to the iterative updating procedure (35 epochs vs. 1 epoch) and the introducing of SLIC superpixel and low level feature, our method is slower than [18]. However, in the next part we can find that the proposed method has much better performance than [18].

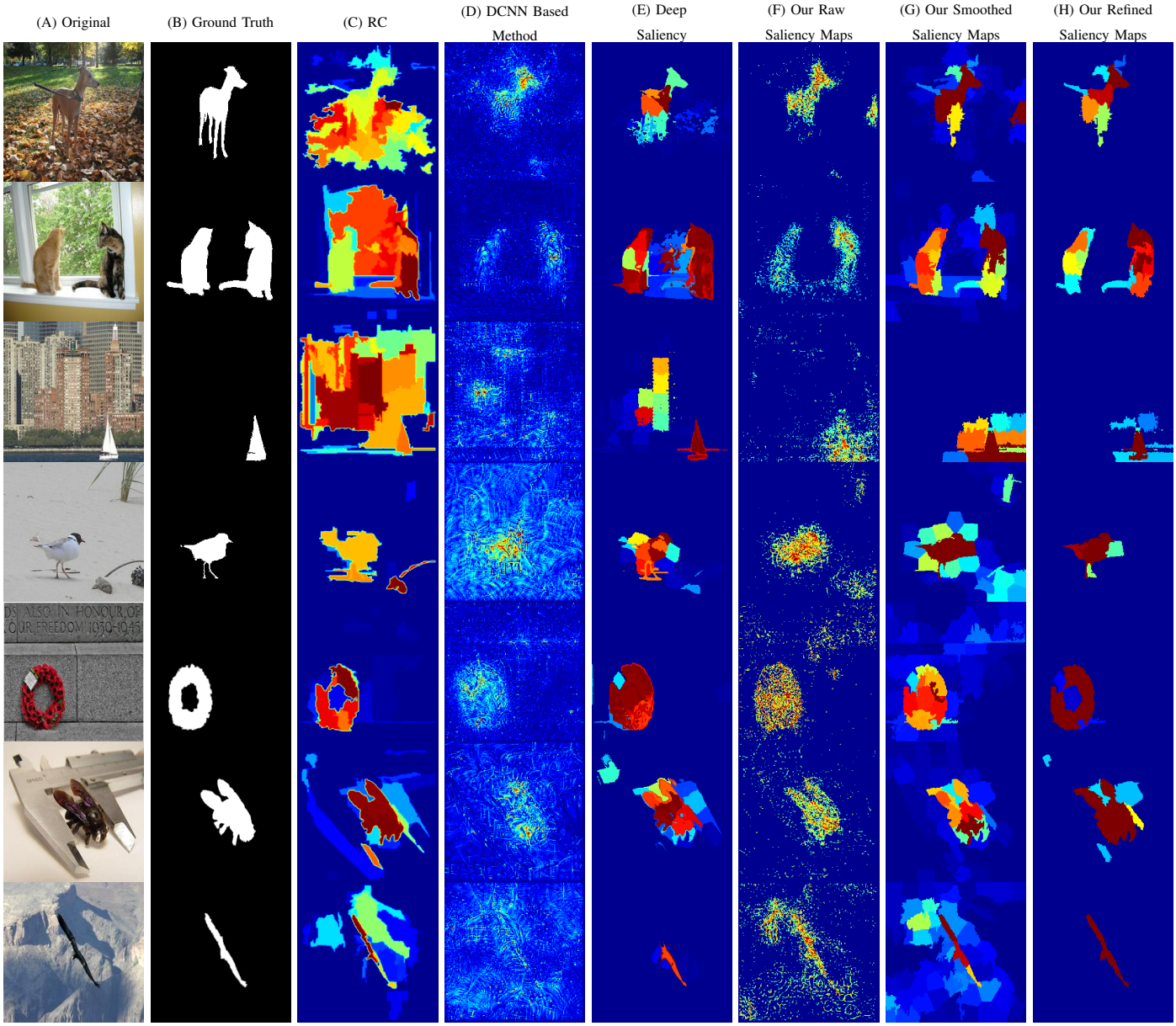

Figure 4. Saliency Results of Pascal VOC 2012 (Row 1 to 4) and MSRA10k (Row 5 to 7). (A) original images, (B) ground truth, (C) Region Contrast saliency maps [12], (D) DCNN based saliency maps by using [18], (E) multi-context deep saliency method [17], (F) our raw saliency maps, (G) our smoothed saliency maps, (H) our refined saliency maps.

3.2.2. The Selection of Hyperparameters. In the cost function Eq. (1), the variable γ is important because it can balance the contribution from the constraints. If we set $\gamma = 0$, the objective function will degenerate to a similar form as [18], and this case can be viewed as the method in [18] (run 35 epochs instead of 1 epoch) plus our post-processing process. We selected different configuration of γ to show the importance of this parameter. The results are shown in Table 2.

γ	0.0	1.0	5.0	10.0	20.0	30.0
F_β	0.487	0.601	0.647	0.661	0.670	0.671
γ	40.0	60.0	100.0	**200.0**	300.0	500.0
F_β	0.672	0.676	0.680	**0.685**	0.677	0.676

TABLE 2. F_β VALUES FOR DIFFERENT γ (PASCAL VOC 2012, RUN 35 EPOCHS, $\beta = 0.3$).

From the experiment results we can learn the importance of γ in the proposed objective function. Specifically, the existing of γ can significantly improve the saliency performance, and when γ less than 200, a larger γ will lead to better saliency performance. Based on the results in Table 2, we set $\gamma = 200$ in our rest experiments.

Another hyper-parameter we need to consider is the number of epochs for the saliency procedure. Generally

	Aim	MISS CB-1	Region Contrast	DCNN based Method	Deep Saliency	Our Raw Saliency Map	Our Smoothed Saliency Map	Our Refined Saliency Map
Pascal	0.587	0.583	0.561	0.347	0.678	0.574	0.659	**0.685**
MSRA10k	-	-	0.843	0.357	**0.927**	0.591	0.687	0.902

TABLE 3. THE F_β VALUE OF DIFFERENT SALIENCY METHODS ON PASCAL VOC 2012 AND MSRA10K DATABASES: AIM [26], MISS CB-1 [15], REGION CONTRAST [12], DCNN BASED METHOD IN [18], DEEP SALIENCY [17] AND OUR THREE KINDS OF SALIENCY MAPS

speaking, a larger number of epoch may increase the performance but slow down the speed. Thus we hope to set a suitable number of epoch to balance the F_β and execution time. Based on the experiments, we use $T = 35$ as a good tradeoff between performance and efficiency. During the back-propagation, we use 20.0 as the initial learning rate, and the learning rate should multiply with 0.999 after each epoch. We use 100 mini-batch size in the saliency process.

3.2.3. Pascal VOC 2012. To measure the performance of object saliency detection on Pascal VOC 2012, we first plot the PR-curves for different methods, which are all shown in Figure 3 (left). From the PR-curves, we can see that the performance of our proposed saliency detection methods significantly outperform the region contrast in [12] and the DCNN based saliency method in [18]. The proposed method also yield better performance than the method in [17] (which the currently the state-of-the-art method in saliency detection).

Table 3 shows the F_β values of the different saliency and segmentation methods (for Pascal VOC 2012 we include two extra well-known methods Aim [26] and MISS CB-1 [15] as baselines), from which we can see that the proposed saliency detection method gives the better F_β value than [26], [15], [12], [18], and [17]. Moreover, comparing with [17], our method also yields much faster speed. Also, by comparing the F_β of our raw saliency map, smoothed saliency map and refined saliency map, we can learn the advantages introduced by our post-processing.

There are many other segmentation methods on Pascal VOC 2012 database, and many of them need to use very complicate algorithms, such as MCG [27] and GrabCut [28]. This paper focuses on saliency detection, not segmentation, thus we only use very simple post-processing methods and avoid to apply those relatively complicate segmentation methods. According to [15], when applying MCG to generate segmentation map, the F_β score will increase from 0.583 to 0.679 (still slightly worse than our final performance).

Finally, in Figure 4 (Row 1 to 4), we provide some examples of the saliency detection results from the Pascal VOC 2012 test set. From these examples we can see that the region contrast algorithm does not work well when the input images have complex background or contain highly variable salient objects, and this problem is fairly common among most bottom-up saliency and segmentation algorithms. On the other hand, we can also see that with the help of SLIC superpixels and low level features, our method can provide better performance than [17].

3.2.4. MSRA10k. Similarly, we also use PR-curves and F_β to evaluate the saliency and segmentation performance on MSRA10k database. From Figure 3 (right), we can see that the proposed method is significantly better than [18], and also has slightly better performance than [12]. As shown in Table 3, our methods also give better F_β value than [12] and [18].

Compare with Pascal VOC 2012, we can find that our post-process methods play more important role on MSRA10k. The main reason is that the images in MSRA10k are much simpler, and the low level features will work well on it.

From Figure 3 and Table 3, we can see that our method performs slightly worse than [17] in the MSRA10k database. The main reason is attributed to that we directly use a mismatched DCNN trained from the ImageNet dataset. We cannot fine-tune the model for this database due to the lack of the training set and class labels in MSRA10k. As shown in the figures, the gap between two methods is very small even though we use a mismatched DCNN for our method.

In Figure 4, we also select some MSRA10k images to show the saliency results (Row 5 to 7).

4. Conclusion

In this paper, we have proposed a novel DCNN-based method for object saliency detection. The method firstly train a regular DCNN for saliency detection. After that, for each test image, we firstly recognize the image class label, and then we can use the pre-trained DCNN to generate a saliency map. Specifically, we attempt to reduce a cost function defined to measure the class-specific objectness of each image, and we back-propagate the corresponding error signal all way to the input layer and use the gradient of inputs to revise the input images. After several iterations, the difference between the original input images and the revised images is calculated as a raw saliency map. The raw saliency maps are then smoothed and refined by using SLIC superpixels and low level saliency features. We have evaluated our methods on two benchmark tasks, namely Pascal VOC 2012 [21] and MSRA10k [22]. Experimental results have shown that our proposed methods can generate high-quality saliency maps in relatively short time (more than 3 times faster than the state-of-the-art DCNN based method in [17]), which clearly outperforming many other existing methods. Comparing with many low-level feature methods, our DCNN-based approach excels on many difficult images, containing complex background, highly-variable salient objects, multiple objects, and very small objects.

References

[1] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, 1995.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *arXiv preprint arXiv:1409.4842*, 2014.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations (ICLR)*, 2015.

[5] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2014.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[7] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *ECCV*, 2014, pp. 297–312.

[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 (PAMI)*, vol. 33, no. 2, pp. 353–367, 2011.

[9] J. Zhang and S. Sclaroff, "Saliency detection: A boolean map approach," in *ICCV*, 2013.

[10] A. Borji, D. N. Sihite, and L. Itti, "Salient object detection: A benchmark," in *ECCV*. Springer, 2012, pp. 414–429.

[11] A. Borji and L. Itti., "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 35, no. 1, pp. 185–207, 2013.

[12] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu, "Global contrast based salient region detection," in *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 409–416.

[13] N. Riche, M. Mancas, B. Gosselin, and T. Dutoit, "Rare: A new bottom-up saliency model," in *IEEE International Conference on Image Processing (ICIP)*, 2012, pp. 641–644.

[14] K. Fu, C. Gong, J. Yang, Y. Zhou, and I. Y.-H. Gu, "Superpixel based color contrast and color distribution driven salient object detection," *Signal Processing: Image Communication*, vol. 28, no. 10, pp. 1448–1463, 2013.

[15] S. Rahman and N. Bruce, "Saliency, scale and information: Towards a unifying theory," in *Advances in Neural Information Processing Systems*, 2015, pp. 2179–2187.

[16] J. Harel, C. Koch, and P. Perona, "Graph-based visual saliency," in *Advances in neural information processing systems (NIPS)*, 2006, pp. 545–552.

[17] 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.

[18] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *arXiv preprint arXiv:1312.6034*, 2014.

[19] H. Pan, B. Wang, and H. Jiang, "Deep learning for object saliency detection and image segmentation," *arXiv preprint arXiv:1505.01173*, 2015.

[20] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2274–2282, 2012.

[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[22] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A survey," *ArXiv e-prints*, 2014.

[23] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Mueller, "How to explain individual classification decisions," *Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010.

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

[25] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *CoRR*, vol. abs/1412.4564, 2014.

[26] N. Bruce and J. Tsotsos, "Saliency based on information maximization," in *Advances in neural information processing systems*, 2005, pp. 155–162.

[27] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 328–335.

[28] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.