Deep Learning for Object Saliency Detection and Image Segmentation

Hengyue Pan†  Bo Wang‡∗  Hui Jiang†
† Department of Electrical Engineering and Computer Science
Lassonde School of Engineering, York University, 4700 Keele Street, Toronto, Canada
‡ Department of Computer Science, Stanford University, California, USA
panhy@cse.yorku.ca, wangbo.yunze@gmail.com, hj@cse.yorku.ca

Abstract

In this paper, we propose several novel deep learning methods for object saliency detection based on the powerful convolutional neural networks. In our approach, we use a gradient descent method to iteratively modify an input image based on the pixel-wise gradients to reduce a cost function measuring the class-specific objectness of the image. The pixel-wise gradients can be efficiently computed using the back-propagation algorithm. The discrepancy between the modified image and the original one may be used as a saliency map for the image. Moreover, we have further proposed several new training methods to learn saliency-specific convolutional nets for object saliency detection, in order to leverage the available pixel-wise segmentation information. Our methods are extremely computationally efficient (processing 20-40 images per second in one GPU). In this work, we use the computed saliency maps for image segmentation. Experimental results on two benchmark tasks, namely Microsoft COCO and Pascal VOC 2012, have shown that our proposed methods can generate high-quality salience maps, clearly outperforming many existing methods. In particular, our approaches excel 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) [13] have achieved the state of the art performance in many computer vision tasks, starting from image recognition [12, 23, 22] and object localization [20] and more recently extending to object detection and semantic image segmentation [9, 11]. 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 on the image [17]. The main goal of the saliency detection is to compute a saliency map that topographically represents the level of saliency for visual attention [25]. For each pixel in an image, the saliency map can provide how likely this pixel belongs to the salient objects [4]. Computing such saliency maps has recently raised a great amount of research interest [3]. The computed saliency maps have been shown to be beneficial to various vision tasks, such as image segmentation [6], object recognition and visual tracking. The saliency detection has been extensively studied in computer vision. 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 based on the criteria related to contrast, rarity and symmetry of image patches [6, 17, 18, 4]. In some cases, the global topological cues may be leveraged to refine the perceptual saliency maps [10, 25, 15]. In these methods, the saliency is normally measured based on different mathematical models, including decision theoretic models, Bayesian models, information theoretic models, graphical models, spectral analysis models [3].

In this paper, we propose a novel deep learning method for the object saliency detection based on the powerful DCNNs. As shown in [12, 23, 22], relying on a well-trained DCNN, we can achieve a fairly high accuracy in object category recognition for many real-world images. Even though DCNNs can recognize what objects are contained in an image, it is not straightforward for DCNNs to precisely locate the recognized objects in the image. In [20, 9, 11], some rather complicated and time-consuming post-processing stages are needed to detect and locate the
objects for semantic image segmentation. 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 (GD) method to iteratively modify each input image based on the pixel-wise gradients to reduce a cost function measuring the objectness of the image. The gradients with respect to all image pixels can be efficiently computed using the back-propagation algorithm for DCNNs. At the end, the discrepancy between the modified image and the original one is calculated as the saliency map for this image. Moreover, as more and more images with pixel-wise segmentation labels become available, e.g. [8, 16], we further propose two more methods to leverage the available pixel-wise segmentation information to learn saliency-specific DCNNs for the object saliency detection. In these methods, the original images as well as the corresponding masked images, in which all objects are masked out according to the pixel-wise labels, are used to train two DCNNs whose output labels are modified to include the masked objects and/or the original objects. Afterwards, we similarly use the GD method to modify each input image to reduce two cost functions formulated to measure the objectness for each case. The saliency map is generated in the same way as the discrepancy between the original and modified images. Since we only need to run a very small number of GD iterations in the saliency detection, our methods are extremely computationally efficient (processing 20-40 images per second in one GPU). The computed saliency maps may be used for many computer vision tasks. In this work, as one particular application, we use the computed saliency maps to drive an popular image segmenter in [1] to perform image segmentation. Experimental results on two databases, namely Microsoft COCO [16] and Pascal VOC 2012 [8], have shown that our proposed methods can generate high-quality saliency maps, clearly outperforming many existing methods. In particular, our DCNN-based approaches excel on many difficult images, containing complex background, highly-variable salient objects, multiple objects, and/or very small objects.

2. Related Work

In the literature, the previous saliency detection methods mostly adopt the well-known bottom-up strategy [6, 17, 18, 4]. They rely on the local image features derived from patches to detect contrast, rarity and symmetry to identify the salient objects in an image. Meanwhile, some other methods have been proposed to take into account some global information or prior knowledge to screen the local features. For example, in [25], a boolean map is created to represent global topological cues in an image, which in turn is used to guide the generation of saliency maps. In [15], the visual saliency algorithm considers the prior information and the local features simultaneously in a probabilistic model. The algorithm defines task-related components as the prior information to help the feature selection procedure. The traditional saliency detection methods normally work well for the images containing simple dominant foreground objects in homogenous backgrounds. However, they are usually not robust enough to handle images containing complex scenes [14].

As an important application, the saliency maps may be used as a good guidance for various image segmentation algorithms. In [7], a recursive segmentation process is used, where each iteration focuses on different saliency regions. As a result, the algorithm can output several potential segmentation candidates from the saliency maps. These candidates may be further merged by maximizing likelihood at all image pixels by considering the low-level features like colour and texture. In [6], a region contrast based image saliency method is proposed to generate the saliency maps, and the SaliencyCut algorithm is used to derive image segmentation from the saliency maps. The SaliencyCut algorithm is based on the standard GrabCut [19] but it uses the proposed saliency maps instead of manually selected bounding boxes for initialization.

Recently, some deep learning techniques have been proposed for object detection and semantic image segmentation [20, 9, 11]. These methods typically use DCNNs to examine a large number of region proposals from other algorithms, and use the features generated by DCNNs along with other post-stage classifiers to localize the target objects. They initially rely on bounding boxes for object detection. More recently, more and more methods are proposed to directly generate pixel-wise image segmentation, e.g. [11]. In this paper, instead of directly generating the high-level segmentation from DCNNs, we propose to use DCNNs to generate middle-level saliency maps in a very efficient way, which may be fed to other traditional computer vision algorithms for various vision tasks, such as semantic segmentation, video tracking, etc.

The work in [21] is the most relevant to the work in this paper. In [21], the authors have borrowed the idea of explanation vectors in [2] to generate a static pixel-wise gradient vector of the network learning objective function, and use it as a saliency map. In our work, we instead use an iterative gradient descent method to generate more reliable and robust saliency maps. More importantly, we have proposed two new methods to learn saliency-specific DCNNs and define the corresponding cost functions, which measure objectness in each model for saliency detection.

3. Our Approach for Object Saliency Detection

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 classi-
fication accuracy in recognizing objects in images. In this work, based on the idea of explanation vectors in [2], 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 [21], we explore several novel methods 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, 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 (GD) 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 to measure the class-specific objectness. The gradients are computed by applying the back-propagation procedure all the way to the input layer.

In section 3.1, we first introduce several different ways to learn DCNNs for saliency detection. In section 3.2, we present our algorithm used to generate the saliency maps from DCNNs in detail.

3.1. Learning DCNNs for Object Saliency

Comparing with the traditional bottom-up methods, DCNNs may potentially learn more prior information for saliency detection. The first type is the class prior, which is provided by class labels of all training images. The second one is the pixel-wise object prior, which may be available as the object masking information in some data sets.

First of all, the regular classification DCNN may be used for saliency detection, which is named as CNN1 hereafter. As shown in Figure 2, CNN1 takes an image as input and it contains a node in the output layer for each object category. CNNs is trained using all labeled images in the training set.

If the pixel-wise object masking information is available, we may mask out the corresponding objects in the original images to generate the so-called masked images. In this way, we may learn different DCNNs to learn the pixel-wise masking information, which will lead to much better DCNNs for the saliency detection purpose. For example, we may learn another DCNN with the masked images only, named as CNN2. As in Figure 2, CNN2 is trained by using all masked images in the training set as input and it has a node in the output layer corresponding to each masked object class.

Moreover, as shown in Figure 2, we train a slightly modified DCNN, named as CNN3, with both original labelled images as well as all masked images, in which all labelled objects are masked out based on the pixel-wise masking. For CNN3, we expand its output layer to include two nodes for each object category: one for the normal objects and the other for the masked objects. For example, when we use an original image containing a giraffe to learn CNN3, we use the label information corresponding to the regular giraffe node in the output layer, denoted as Giraffe. Meanwhile, when we use the same image with the animal region masked out, we use the label information corresponding to the masked giraffe node in the output layer, denoted as Giraffe. Comparing with CNN2, CNN3 is trained in a way to learn the contrast information between original labelled images and their masked versions.
3.2. Generating Saliency Maps from DCNNs

After the three DCNNs (CNN1, CNN2 and CNN3) are learned, we may apply our saliency detection methods to generate the class-specific object saliency map, as shown in Figure 1.

For each input image, we firstly use CNN1 to generate its class label, denoted as $l$, as in a normal classification step. Next, we may use one of the DCNNs to generate the saliency map. In this step, the selected DCNN is kept unchanged and instead we attempt to modify the input image in the pixel level to reduce a cost function, which is defined to measure the class-specific objectness in each case. In the following, we introduce how to define the cost function for each DCNN and the details to generate the saliency maps.

For CNN1, we denote its output nodes after softmax as $\{y_i^{(1)} | i = 1, \cdots, N\}$, each of which corresponds to one class label ($N$ classes in total). Assume an input image $X$ is recognized as class $l$, we may define the following cost function to measure the class-specific objectness in this case:

$$F^{(1)}(X|l) = \ln y_l^{(1)}. \quad (1)$$

The key idea here is that we try to modify the image $X$ to reduce the above cost function and hopefully the underlying object (belonging to class $l$) will be removed as the consequence. In this paper, we propose to use an iterative GD procedure to modify $X$ as follows:

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

where $\epsilon$ is a learning rate, and we floor all negative gradients in the GD updates. We have observed in our experiments that the cost function $F^{(1)}(X|l)$ can be significantly reduced by running only a small number of updates (typically 10-15 iterations) for each image.

We can easily compute the above gradients using the standard back-propagation algorithm. Based on the cost function $F^{(1)}$ in eq.(1), we can derive the error signals in the output layer as $e_i^{(1)} = \delta(i - l) - y_i^{(1)}$ ($i = 1, \cdots, N$), where $\delta(\cdot)$ stands for the Kronecker delta function. These error signals are back-propagated all the way to the input layer to derive the above gradient, $\frac{\partial F^{(1)}(X|l)}{\partial X}$, for saliency detection.

For CNN2, we denote its output nodes after softmax as $\{y_i^{(2)} | i = 1, \cdots, N\}$, each of which corresponds to one class of masked objects. Given an input image $X$ and its recognized class $l$ (from CNN1), we define the following cost function for this case:

$$F^{(2)}(X|l) = -\ln y_l^{(2)}. \quad (3)$$

Similarly, we apply the above GD algorithm in eq.(2) to modify the image to reduce this cost function. By reducing $F^{(2)}$, we try to increase the probability of the corresponding masked class. Intuitively, we attempt to alter the input image to match the masked images in that class as much as possible. In the same way, the error signals in the output layer can be simply derived as $e_i^{(2)} = y_i^{(2)} - \delta(i - l)$ ($i = 1, \cdots, N$), which are back-propagated all the way to the input layer to compute $\frac{\partial F^{(2)}(X|l)}{\partial X}$.
Algorithm 1 GD based Object Saliency Detection

**Input:** an input image \( X \), CNN1, CNN2 and CNN3; Use CNN1 to recognize the object label for \( X \) as \( l \); Choose a saliency model (CNN1 or CNN2 or CNN3); \( X^{(0)} = X \);

**for each** epoch \( t = 1 \) to \( T \) **do**

**forward pass:** compute the cost function \( F(X|l) \);

**backward pass:** back-propagate to input layer to compute gradient:

\[
X^{(t)} = X^{(t-1)} - \epsilon \cdot \max \left( \frac{\partial F(X|l)}{\partial X}, 0 \right);
\]

**end for**

Average over RGB: \( S = \frac{1}{3} \sum_{i=1}^{3} (X^{(0)}_i - X^{(T)}_i) \);

Prune noises with a threshold \( \theta \): \( S = \max(S - \theta, 0) \);

Normalize: \( S = \frac{S}{\|S\|} \);

**Output:** the raw saliency map \( S \);

Finally, for CNN3, we denote its output nodes after softmax as \( \{ y_i^{(3)} \mid i = 1, \ldots, 2N \} \), each of which corresponds to either an image class or a masked class. Given an input image \( X \) and its recognized class \( l \), we find the output node corresponding to the masked class of \( l \), denoted as \( \bar{l} \). We define the cost function for CNN3 as follows:

\[
F^{(3)}(X|l) = -\ln y_i^{(3)}.
\]  

Similarly, the image is modified by running the GD algorithm in eq.(2) to reduce \( F^{(3)} \), or equivalently increase \( y_i^{(3)} \). Since all output nodes are normalized by softmax, by increasing \( y_i^{(3)} \), its original output node \( y_i^{(3)} \) will be reduced accordingly. Intuitively speaking, by doing so, we attempt to use the contrast information learned by CNN3 to modify an image from its original class to match the corresponding masked version for the object saliency detection. Similarly, the error signals in the output layer is derived as \( e_i^{(3)} = y_i^{(3)} - \delta(i - \bar{l}) \), where \( i = 1, \ldots, 2N \).

At the end of the gradient descent updates, the object saliency map 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 may apply a simple threshold to filter out some background noises of the raw saliency maps. The entire algorithm to generate the raw saliency maps is shown in **Algorithm 1**.

For each image, we can obtain 3 different saliency maps with the three different DCNNs. We have found that we may obtain even better results if we combine the saliency maps from CNN2 and CNN3 by taking an average between them. We can also use a simple image dilation and erosion method to smooth the raw saliency maps to derive the final saliency maps.

4. Saliency Refinement and Image Segmentation

Here, as one application, we use the derived saliency maps to perform semantic image segmentation.

Inspired by the recent work in [11], we aim to refine our saliency map using segmentation and also achieve a binary salient object segmentation. We make use of a recent state-of-art image segmentation tool called Multiscale Combinatorial Grouping (MCG) [1], which provides us with a well-defined contour map and also a set of object proposals. The idea of refining the saliency map is simple: we randomly select 50 points from salient point sets and use these selected points as seed information to perform an interactive image segmentation. We restrict it to be a binary segmentation to extract salient foreground. We independently run this experiment 100 times and average the binary segmentation results, then we can get a refined saliency.

To obtain the final binary salient object segmentation, we use the top 50 object proposals generated by MCG. For each proposal associated with super-pixel segmentation, we choose the one with the highest Jaccard index value with a thresholded binary mask from the provided saliency map. Specifically, given the final saliency map as \( S \), we get a binary mask \( M_1 = I\{S > \delta\} \), where \( \delta \) is a threshold (we set it to be 0.5 in this work). For each super-pixel segmentation from each proposal, denoted as \( M_2 \), we calculate the Jaccard index as follows:

\[
Jaccard(M_1, M_2) = \frac{\|M_1 \cap M_2\|}{\|M_1 \cup M_2\|}
\]

The super-pixel segmentation that has the largest Jaccard index with the thresholded saliency map is chosen as the final salient object segmentation.

5. Experiments

We select two benchmark databases to evaluate the performance of the proposed object saliency detection and image segmentation methods, namely Microsoft COCO [16] and Pascal VOC 2012 [8]. Both databases provide the class label of each image as well as the pixel-wise segmentation map (ground truth), thus we can generate the masked images to train the required DCNNs in our proposed methods. Here we compare our approaches with two existing methods: i) the first one is the Region Contrast saliency method and the SaliencyCut segmentation method in [6]. This method is one of the most popular bottom-up image saliency detection methods in the literature and it has achieved the state-of-the-art image saliency and segmentation performance on many tasks; ii) the second one is the DCNN based image saliency detection method proposed in [21]. Similar to our approaches, this method also use DCNNs and the back-propagation algorithm to generate
saliency maps. In our experiments, we use the precision-recall curves (PR-curves) against the ground truth as one metric to evaluate the performance of saliency detection. As [6], 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 $F_\beta$ 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 $\beta$ as follows [4]:

$$F_\beta = \frac{(1 + \beta^2) \times Prec \times Rec}{\beta^2 \times Prec + Rec} \quad (5)$$

In this paper, we follow [6] to set $\beta^2 = 0.3$ to emphasize the importance of $Prec$. Note that we only get a single $F_\beta$ value for each binary segmentation map for segmentation. However, we may derive a sequence of $F_\beta$ values along the PR-curve for each saliency map and the largest one is selected as the performance measure (see [4]).

### 5.1. Databases

Microsoft COCO [16] is a new image database that may be used for several vision tasks including image classification and segmentation. The database currently contains 82,783 training images and 40,504 validation images with 80 labeled categories. In our experiments, we only select the images that contain one category of objects because these images are more compatible with the available DCNN baseline, which is normally trained using the ImageNet data. The selected COCO subset contains 6869 training images and 3479 validation images with 18 different classes.

Pascal VOC 2012 database [8] can also be used for our proposed algorithms, but its sample size is much smaller comparing with COCO. We use the whole dataset, which has 1464 training images and 1449 validation images with 20 label categories in total. 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 most important object to label the images for our DCNN training process.

As we have mentioned earlier, we need to train the three DCNNs, i.e., CNN1, CNN2 and CNN3, for each dataset. However, because the training sets are relatively small in both COCO and Pascal, we have used a well-trained DCNN for the ImageNet database, which contains 5 convolutional layers and 2 fully connected layers\(^1\). We only use the above-mentioned training data to fine-tune this DCNN for each task with MatConvNet in [24]. For the Pascal VOC 2012 data, we further use 5-fold cross-validation to expand the training sample size. We use the training set and about 80% of the validation data to fine-tune the model and it is used to test the remaining 20% of data. We rotate five times to cover the entire test set. In Table 1, we have listed the top-1 and top-5 classification error rates when the fine-tuned DCNNs are used to recognize the test sets on these two tasks.

| Dataset   | CNN1   | CNN2   | CNN3   |
|-----------|--------|--------|--------|
| MS COCO   | Top-1  | 12.2%  | 19.1%  | 16.7%  |
|           | Top-5  | 2.4%   | 3.2%   | 4.0%   |
| Pascal    | Top-1  | 20.3%  | 35.1%  | 26.5%  |
| VOC 2012  | Top-5  | 3.1%   | 8.4%   | 9.7%   |

Table 1. The classification error rates of three CNNs on the MS COCO and Pascal VOC 2012 test sets.

The classification errors on the test sets imply that the training sample size is still not enough for training deep convolutional networks well, especially for Pascal VOC 2012. However, as we will see, the proposed algorithms can still yield good performance for saliency detection and segmentation. If we have more training data that include class labels and the masked images, we may expect even better saliency and segmentation results.

### 5.2. Saliency and Segmentation Results

In this part we will provide saliency detection and segmentation results on these two databases. In the following, the PR-curves, $F_\beta$ values and some sample images will be used to compare different methods.

#### 5.2.1 Microsoft COCO

For the object saliency detection, we first plot the PR-curves for different methods, which are all shown in Fig. 3. From the PR-curves, we can see that the performance of our proposed saliency detection methods significantly outperform the region contrast in [6] and the DCNN based saliency method in [21]. Moreover, it has shown that CNN2 and CNN3 yields better performance than CNN1, which demonstrates that the utilization of masked images in model training can further improve the saliency detection performance.

Figure 4 shows the $F_\beta$ values of the different saliency and segmentation methods, from which we can see that the proposed three saliency detection methods give the better $F_\beta$ values than [6] and [21]. Starting from our saliency maps, the MCG-based segmentation algorithm can yield a good performance as well. Moreover, the segmentation results have also shown the benefits to use the masked images as prior information in the DCNN training. Finally, in Figure 7 (Column 1 to 5), we also provide some examples of the saliency detection and segmentation results from the COCO test set. From these examples we can see that the re-

\(^1\)We use the net imagenet-vgg-s in http://www.vlfeat.org/matconvnet/ [5].
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 masked images in training our proposed DCNN-based saliency detection methods concentrate much better on the salient objects. Note that the segmentation results based on [21] are not shown in Figure 7 since they are significantly worse than others.

5.2.2 Pascal VOC 2012

Similarly, we also use PR-curves and $F_\beta$ to evaluate the saliency and segmentation performance on Pascal VOC 2012 database. From Fig. 5, we can see that the proposed methods are significantly better than [21], and the DCNNs that make use of masked images yield comparable performance as [6]. As shown in Fig. 6, our methods still give slightly better $F_\beta$ values for both saliency detection and segmentation than [6] but the difference between them is not significant. This may be partially attributed to the poor DCNN models in the Pascal dataset, which is fine-tuned by only a very small number of in-domain images. In Fig. 7, we also select several Pascal images to show the saliency and segmentation results (Column 6 to 10). Some of these examples have suggested that our methods are able to handle the images that contain multiple objects.

6. Conclusion

In this paper, we have proposed several novel DCNN-based methods for object saliency detection and image segmentation. The methods may utilize both original training images and masked images to train several DSCNNs. For each test image, we firstly recognize for the image class label, and then we can use any of the these DCNNs 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 signals 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 saliency map. The saliency
maps can be used to initialize an image segmentation algorithm to derive the final segmentation results. We have evaluated our methods on two benchmark tasks, namely MS COCO [16] and Pascal VOC 2012 [8]. Experimental results

Figure 7. Saliency Results of MS COCO (Column 1 to 5) and Pascal (Column 6 to 10). (A) original images, (B) masked images, (C) Region Contrast saliency maps [6] (D) DCNN based saliency maps by using [21], (E) to (H) raw saliency maps using CNN1, CNN2, CNN3 and CNN2 + CNN3, (I) smoothed saliency maps of (H) using image dilation and erosion, (J) refined saliency maps of (I), (K) segmentation using SaliencyCut [6] and (L) our segmentation results based on (J).
have shown that our proposed methods can generate high-quality salience maps, clearly outperforming many existing methods. In particular, our DCNN-based approaches excel on many difficult images, containing complex background, highly-variable salient objects, multiple objects, and very small objects.

Acknowledgement

This work is partially supported by a research donation from iFLYTEK Co., Ltd., Hefei, China. We acknowledge NVIDIA for donating a Tesla K40 GPU card under the Academic Partnership Program. The first author is supported by a scholarship from China Scholarship Council (CSC).

References

[1] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marqués, and J. Malik. Multiscale combinatorial grouping. In Computer Vision and Pattern Recognition (CVPR), 2014. 2, 5
[2] 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, 11:1803–1831, 2010. 2, 3
[3] A. Borji and L. Itti. State-of-the-art in visual attention modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 35(1):185–207, 2013. 1
[4] A. Borji, D. N. Sihite, and L. Itti. Salient object detection: A benchmark. In ECCV, pages 414–429. Springer, 2012. 1, 2, 6
[5] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In British Machine Vision Conference, 2014. 6
[6] 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), pages 409–416. IEEE, 2011. 1, 2, 5, 6, 7, 8
[7] M. Donoser, M. Urschler, M. Hirzer, and H. Bischof. Saliency driven total variation segmentation. In IEEE International Conference on Computer Vision (ICCV), pages 817–824, 2009. 2
[8] 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, 88(2):303–338, June 2010. 2, 5, 6, 8
[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014. 1, 2
[10] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In Advances in neural information processing systems (NIPS), pages 545–552, 2006. 1
[11] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In ECCV, pages 297–312, 2014. 1, 2, 5
[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems 25, pages 1097–1105, 2012. 1
[13] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361, 1995. 1
[14] J. Li, Y. Tian, and T. Huang. Visual saliency with statistical priors. International Journal of Computer Vision, 107(3):239–253, 2014. 2
[15] J. Li, Y. Tian, T. Huang, and W. Gao. Probabilistic multi-task learning for visual saliency estimation in video. International Journal of Computer Vision, 90(2):150–165, 2010. 1, 2
[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In ECCV, pages 740–755, 2014. 2, 5, 6, 8
[17] 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), 33(2):353–367, 2011. 1, 2
[18] N. Riche, M. Mancas, B. Gosselin, and T. Dutoit. Rare: A new bottom-up saliency model. In IEEE International Conference on Image Processing (ICIP), pages 641–644, 2012. 1, 2
[19] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (TOG), 23(3):309–314, 2004. 2
[20] 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. 1, 2
[21] 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. 2, 3, 5, 6, 7, 8
[22] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations (ICLR), 2015. 1
[23] 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. 1
[24] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. CoRR, abs/1412.4564, 2014. 6
[25] J. Zhang and S. Sclaroff. Saliency detection: A boolean map search. In Advances in Neural Information Processing Systems 27, pages 1097–1105, 2014. 1