Self-explanatory Deep Salient Object Detection

Huaxin Xiao, Jiashi Feng, Yunchao Wei, Maojun Zhang

Abstract—Salient object detection has seen remarkable progress driven by deep learning techniques. However, most of deep learning based salient object detection methods are black-box in nature and lacking in interpretability. This paper proposes the first self-explanatory saliency detection network that explicitly exploits low- and high-level features for salient object detection. We demonstrate that such supportive clues not only significantly enhances performance of salient object detection but also gives better justified detection results. More specifically, we develop a multi-stage saliency encoder to extract multi-scale features which contain both low- and high-level saliency context. Dense short- and long-range connections are introduced to reuse these features iteratively. Benefiting from the direct access to low- and high-level features, the proposed saliency encoder can not only model the object context but also preserve the boundary. Furthermore, a self-explanatory generator is proposed to interpret how the proposed saliency encoder or other deep saliency models making decisions. The generator simulates the absence of interesting features by preventing these features from contributing to the saliency classifier and estimates the corresponding saliency prediction without these features. A comparison function, saliency explanation, is defined to measure the prediction changes between deep saliency models and corresponding generator. Through visualizing the differences, one can interpret the capability of different deep neural networks based saliency detection models and demonstrate that our proposed model indeed uses more reasonable structure for salient object detection. Extensive experiments on five popular benchmark datasets and the visualized saliency explanation demonstrate that the proposed method provides new state-of-the-art.

Index Terms—Neural networks, deep learning, salient object detection, model interpretability

I. INTRODUCTION

Salient object detection aims to localize and segment the most visually attractive objects from a given image, simulating the visual attention process in human vision systems. Segmenting out conspicuous objects extensively facilitates numerous computer vision tasks by filtering irrelevant distracting information, such as for object tracking [28], image retrieval [5], weakly-supervised object segmentation [30], [41], to name a few.

Recently, remarkable progress for salient object detection has been made with the development and deployment of deep neural networks (DNNs) [8], [9], [14], [16]–[20], [24], [27], [38]–[40], [43] of various architectures. Among them, some DNN based models [18], [39], [43] simply determine whether a local region is salient or not through a single classifier on top of its hierarchically extracted representation. The performance of such models is severely limited by the narrow "receptive field" over the raw input image due to ignoring important context clues. To remedy deficiency of the narrow perspective by only applying a network locally, another global networks in [18], [39], [43] are introduced to provide global object context. Besides relying on regional prediction, some pixel-level dense prediction networks were developed and adopted. For instance, the fully convolutional networks (FCNs) [20] can produce dense prediction by associating each pixel with a classifier. One of the advantages offered by such dense prediction networks is to preserve rich spatial and local details that are recognized to be crucial for saliency detection [24], [27]. Upon this kind of models, diverse refinements, e.g., introducing recurrent units [14], [40] and superpixel-based filtration [9], [19], [22], are proposed to better exploit fine details of the salient object. Due to the various challenges in salient object detection, e.g., low contrast between the salient object and the background, difficult situations in the non-salient regions and multiple salient objects (will be discussed in Section IV-D2), how to properly integrate the local and global perspective of saliency context is still crucial for improving performance. To better exploit multi-level clues without incurring additional network engineering efforts or expensive refinement operations, this paper introduces a novel end-to-end architecture where features from multiple levels are directly and recursively integrated to provide more powerful support for detecting salient objects.

Besides developing a novel deep model for salient object...
detection, we observe that all of existing DNN based models are lacking in interpretability because of their “black-box” nature. Enhancing transparency of deep models would offer reasonable explanations on their predictions and in turn benefit these models in following two aspects. First, revealing the internal mechanism of deep saliency detection models can tell which part of an image is most influential (either positively or negatively) for prediction making. Such explanation is helpful to understand the effectiveness of different architectures in saliency prediction. Secondly, understandable and transparent property of DNNs will encourage saliency detection model to exploit really useful cues for prediction and inspire novel network architectures. Hence, we argue that a desired deep saliency detection model should not only detect the salient object accurately but also provide self-explanations on its predictions.

To develop a salient object detection model with both superior performance and clear interpretability, we propose a Self-Explanatory saliency detection Network (SENet) in this work. The SENet consists of two essential components: a saliency encoder and a self-explanatory generator. The saliency encoder predicts saliency scores for each pixel from the input image, or in other words it encodes a raw input image into a saliency map. Within the encoder, we propose to explicitly widen access of the saliency classifier to multi-level clues by efficiently reusing multi-scale features through novel dense short- and long-range connections. These multi-range dense connections facilitate the encoder to make saliency predictions by exploiting all computed feature maps, which is beneficial for modeling the saliency context and preserving object boundary. Besides, all features are explicitly connected to the classifier via dense connections. Such architecture can improve the model training process as the gradients from the final classifier are directly propagated backwards to features at multiple levels—which can effectively alleviate the notorious vanishing gradient issue [10]. Benefited from this novel architecture of the encoder, our proposed SENet can detect the salient region with high confidence and produce a complete and sharp salient region, as illustrated by the example given in the second row of Figure 1.

To enhance model interpretability, a self-explanatory generator is introduced for SENet that infers an influence distribution over the learned representations from the saliency encoder. As a probabilistic model, the generator estimates the probability of encoder predictions with absence of some input features. A comparison function, saliency explanation, is defined to track the prediction changes. In this way, the saliency explanation can clearly present the positive or negative effect of these absent features over the saliency predictions. More importantly, after going through all input features, we can integrate these results to explain why particular saliency prediction is made for the entire features (e.g., an image).

We provide example saliency detection results from different models along with explanation on their decision to better illustrate the idea. As shown in the first row of Figure 1, we apply the self-explanatory generator to provide explanation for three state-of-the-arts and the SENet. The interesting features in the generator are the raw input image. The red areas indicate the regions that provide positive support for salient object detection, and the blue ones represent that such regions could suppress and distract the performance of saliency detection. The white color areas mean no significant sensitivity for saliency detection. From the visualized explanation, one can observe that the proposed SENet correctly relies on the fish (salient object) for saliency prediction and regards the surroundings of the salient object as negative factors. This means leaving out the local area around the fish can improve saliency prediction. This is understandable intuitively as leaving the area around the object out would make the object more salient.

For the compared models in Figure 1, the positive areas (which influence the model decision most significantly) spread among salient and non-salient region that lead to false positive detection results in saliency maps. These visualization results well explain why those models give inferior saliency detection performance compared with ours.

In short, the main contributions of this paper can be summarized as follows:

- We develop a saliency detection model that is self-explanatory. Different from existing deep saliency detection models that are generally “black-box”, our proposed model is able to provide reasonable explanation for saliency detection. Results on benchmark datasets clearly demonstrate that the proposed model provides a new state-of-the-art.
- We propose a new end-to-end deep saliency encoder with dense connections. It enriches the access of classifier to the encoded features by dense short- and long-range connections. These dense connections introduce multi-level features reusing and implicit deeply supervision which makes the network effective and efficiency.
- We proposed a generator to explain saliency predictions. The generator estimates the encoder prediction with the absence of some interesting features. Through visualizing the differences between the predictions of generator and encoder, we can offer interpretable evidence for understanding the underlying basis of saliency predictions.

The rest parts of this paper are organized as follows. Section II reviews existing DNN based saliency detection approaches and the methods to interpret DNNs. Section III presents the details of the proposed SENet. Extensive experimental results on 5 popular benchmarks, the network interpretation and comparisons with state-of-the-art are presented in Section IV. And Section V concludes this paper.

II. RELATED WORK

A. DNN based Salient Object Detection

Over the past years, DNNs have become powerful tools on various computer vision tasks and achieved state-of-the-art performance. Here, we focus on reviewing the related works that employ DNNs to detect salient objects.

One of popular schemes for DNN based saliency detection is to combine multi-networks for multi-scale saliency context. Different context networks can supplement each other and provide various information. Wang et al. [39] proposed a local network on cropped image patches for pixel-level saliency
detection. To ensure label consistency, another network was trained to predict the saliency score of each object proposal from a global perspective. In [43], two DNNs were jointly trained and integrated to capture the global and local context. Li and Yu [18] aggregated three types of DNNs features from only salient object region, immediate neighboring region of salient object and the full image excluding the salient object respectively. Lee et al. [16] encoded the hand-crafted features by multiple convolutional units as supplement to the high-level DNNs features. Several fully connected layers were adopted to concatenate these features and classify the saliency region. In [19], a segment-wise spatial pooling branch was proposed to refine the boundaries of salient object which was inferred by a multi-scale FCNs branch. Similarly, Tang and Wu [38] generated the segments in region-level network adaptively. A fusion network was trained to combine the pixel-level and region-level saliency map.

Another strategy for DNN based saliency detection is to refine the saliency results in stages. To obtain an entire object perception capability, Li et al. [22] performed a multi-task learning scheme in conjunction with the task of semantic segmentation. Then the saliency map was refined by graph Laplacian regularized nonlinear regression to generate fine-grained boundaries. Kuen et al. [14] exploited the spatial transformer and recurrent network units to iteratively attend to image region and refine it progressively. Liu and Han [24] proposed a hierarchical recurrent convolutional unit to refine the details of saliency map step by step. Wang et al. [40] fed forward image and previous stage saliency map to the FCNs recurrently. The recurrent architecture of [40] made the network learn to correct the previous errors. In [12], region proposals were adopted to learn the boundaries of salient object. Then the coarse and regional saliency maps were merged and refined by image specific low-to-mid level information. Hu et al. [9] incorporated DNNs with level set function for learning more accurate boundaries and compact saliency. Then, the level set map was refined by a superpixel-based guided filter layer to propagate saliency.

Recently, a simplified and efficient DNN [27] was proposed to combines local and global information through a multi-resolution grid structure and a boundary penalty loss was implemented to ensure spacial consistency. Hou et al. [8] exploited deeply supervision [15] with stage-wise short connections on multi-scale feature maps. A final saliency map was inferred by merging the middle side outputs while throwing away the deepest and the shallowest side outputs. Li et al. [17] further developed salient object detection into instances level. A multi-scale shared saliency network was proposed to generate high-quality saliency masks and salient object contours. Based on the detected object contours, [17] generated the salient object proposals by a MAP-based subset optimization method to obtain the salient instances.

In this paper, we attempt to unify the multi-stages features by dense refinement. Compared to the multi-network scheme [18], [38], [39], [43], the proposed method need no extra network to model the salient object context or capture the boundary of object. The refinement operation is completed by the dense and long connections which are more effective and efficient than the recurrent units [14], [40] or the regional filtration [9], [19], [22].

B. Explanation on DNN Models

With the rapid progress of DNNs, the characteristic of non-transparency can be a barrier to further understand and improve DNNs. Hence, enhancing interpretability of deep learning models has emerged as an urgent requirement [44]. Generally, there are two approaches to understand DNNs: forward propagation based and backpropagation based [34].

For the forward propagation based approaches, they leave interesting features out and feed forward the rest parts of the features to the DNNs. The change of DNNs prediction explains the response of DNNs over the interesting features. Zeiler and Fergus [42] directly masked out parts of an image by a gray square, and monitoring the activation of intermediate layers or the output of the classifier. Based on the work of [32], Zintgraf et al. [44] adopted a more strict rule to leave out the interesting features. The absence of interesting features was formulated as Bayesian inference which was more reasonable than the direct gray patch [42].

The backpropagation based approaches propagate the gradient from classifier towards interesting neuron or input image and visualize the activation to explain the DNNs. Based on the gradient of the certain class score w.r.t. the input image, Simonyan et al. [35] provided two types of explanation: produce an image that maximizes the interesting class prediction or compute an activation map for the interesting class. During the backpropagation of [35], the gradient of the ReLUs would be zero if the input to the ReLU is negative in the phase of forward propagation. To overcome the discontinued gradients of [35], Sprigenberg et al. [37] proposed guided backpropagation to adds an additional guidance signal from the higher layers to usual backpropagation. Moreover, Shrikumar et al. [34] proposed to compare the activation of a neuron with its reference state to avoid the discontinuities in the gradient and highlight inputs that contribute positively or negatively to the output.

All above-mentioned methods focus on interpreting a single classifier for image classification. In this paper, we attempt to explain the behavior of pixel-wise dense classifiers. To this end, we extend the work of [44] to analyze the influence of interesting features over the dense classifiers and provide interpretation for the dense prediction, i.e., saliency detection. We fuse multiple explanation maps instead of the single category used by [44] and propose a comparison function, i.e., saliency explanation, to measure the prediction changes of deep saliency models.

III. PROPOSED SELF-EXPLANATORY NETWORK

We start with introducing the overall architecture of the proposed SENet. We then proceed to provide details of the two novel and critical components, i.e., the encoder for saliency prediction and the generator for explanation, one by one.
Fig. 2. Overall architecture of SENet. The saliency encoder employs a fully convolutional architecture to map the input image to a pixel-level saliency map. First, the saliency encoder passes the input image through a pre-trained image classification network and extracts features at multiple levels. A specifically designed context block is introduced to encode these features into saliency-diagnosis context features. Then, the encoded features are upsampled and concatenated to lower-level features iteratively by the dense short- and long-range connections. The self-explanatory generator is proposed to interpret the saliency prediction made by the encoder. For a given image, the generator purposively makes some parts of the image absent by preventing those parts from contributing to the encoder and tracks changes of the saliency prediction. After all parts of the image are examined, the discovered changes of the encoder are integrated to form the saliency explanation map. Best viewed in color.

A. Overall Architecture

The proposed SENet model consists of two novel components: the saliency encoder for saliency detection and the self-explanatory generator that identifies explanation for supporting saliency detection. The overall architecture is shown in Figure 2. The saliency encoder encodes an input image to a pixel-level saliency map where each element indicates the probability of the corresponding pixel belonging to a salient object. The encoder employs a fully convolutional architecture to make the dense prediction and it works as follows. First, the saliency encoder passes the input image through a pre-trained image classification network and extracts features at multiple levels. A specifically designed context block is introduced to encode these features into saliency-diagnosis context features. Such encoded features are then upsampled and concatenated with lower-level features recursively via newly proposed dense short- and long-range connections until the resolution of the features equals to the input image. These dense connections can ensure the saliency classifier fully exploit relevant context for making saliency prediction with enhanced accuracy.

The self-explanatory generator in Figure 2 is proposed to interpret the saliency prediction made by the encoder. For a given image, the generator purposively masks some parts of the image by preventing those parts from contributing to the encoder and tracks changes of the saliency prediction. In this way, the supportive evidence used from the raw input image can be effectively identified. After all parts of the image are examined, the discovered changes of the encoder are integrated to form the saliency explanation map (as shown in the bottom panel of Figure 2). Through visualizing the saliency explanation, the generator explicitly identifies the positive and negative factors of the given image that influence the saliency predictions. The positive factors mean the regions in the given image are heavily relied by the saliency models to make decision while the negative factors convey a message that removing these regions can bring a more accurate saliency result. That is, the presence of the negative regions may degrade the performance of saliency model. Moreover, the generator is compatible with other DNN based saliency models which can be adopted as a metric to compare the capability of different saliency models.

B. Saliency Encoder

The motivation behind the saliency encoder in SENet is to ensure the dense saliency classifier has access to both global and local perspective over the input image simultaneously when making saliency predictions. Both global and local information are crucial for saliency detection [18], [43] while a systematic method to comprehensively exploit these information is still absent. Therefore, we propose to integrate the information within the saliency detection models at multiple processing levels. A simple yet effective structure, called dense connections, is adopted to enrich the connection among intermediate features for fully utilizing the valuable information. In particular, two types of dense connections, i.e. the short- and long-range connections, are introduced.
to merge features at the same scale and transport features to other scales respectively. These dense connections avoid resorting to expensive ensemble of multiple networks for different features or contexts \cite{18, 38, 39, 43} and also effectively get rid of additional refinement stages, e.g. the superpixel-based filtration \cite{9, 19, 22} and the recurrent refinement \cite{14, 40}. When the saliency classifier makes decisions, the dense structure directly provides the classifier with various and necessary information instead of separating the decisions process of classifier into multi-networks or multi-stages. This end-to-end structure is thus more effective and efficient for salient object detection.

We now explain architectural details of the saliency encoder component. We call all the subsampling operations involved between two adjacent feature maps as forming one stage. As shown in the top panel of Figure 2, at the stage \(i\), the saliency encoder in SENet receives and concatenates the features from all the higher stages \(i, i + 1, \ldots, L\) into a new feature:

\[
x_{i-1} \leftarrow [x_{i-1}, H_i^{up}(x_i), H_{i+1}^{up}(x_{i+1}), \cdots, H_L^{up}(x_L)],
\]

where \(x_i\) is the feature map from the stage \(i\), \(H_i^{up}(\cdot)\) is an upsampling transformation and \(L\) is the total number of stages in the encoder.

These dense connections benefit the proposed encoder in the following desired aspects: (1) the saliency context can be reused at different stages which helps the encoder effectively collect information from both global and local perspectives and improve saliency predictions; (2) the features at each stage have direct access to the gradients from the top loss layer which can be regarded as implicit deep supervision \cite{15} and facilitates learning features for saliency detection; (3) the dense connections can explicitly widen flow of the gradient and make model training easier \cite{10}.

The dense connections receive features from a pre-trained DNN model for image classification. To make the extracted features more suitable for salient object detection, we fine-tune outputs of the classification network through a context block. As demonstrated in the top panel of Figure 2, before the dense connections, the proposed saliency encoder employs a residual-like block to augment the features. Concretely, after each stage \(i\), the feature maps \(f_i\) are passed through a context block and give the following output \(x_i\):

\[
x_i = G_i(f_i; W_i) + f_i,
\]

where \(W_i\) is the weight parameter of the context block at stage \(i\) and \(G_i(\cdot; \cdot)\) is a non-linear transformation over the input feature, as detailed below.

As shown in the top panel of Figure 2, the non-linear transformation \(G_i\) consists of two \(3 \times 3\) convolutional layers. Such architecture design is based on following considerations. Using a single trainable layer to build a model with sufficiently large capacity is not easy \cite{7}. Therefore, we stack two \(3 \times 3\) convolution layers in the context block for stronger learning capacity. Combining the context block and dense connections, the proposed saliency encoder within SENet can exploit the supportive clues from multi-level features effectively. In particular, high-level semantic features would help localize the salient object while the low-level fine-grained features would refine the details (e.g. boundaries).

Figure 3 illustrates how the dense connections benefit salient object detection through comparing results with the one from the model without dense connections. For getting the results without dense connections in Figure 3(c) we degenerate the architecture specified in Eqn. (1) into \(x_{i-1} \leftarrow [x_{i-1}, H_i^{up}(x_i)]\). Namely, only a single adjacent connection is used. Comparing results in Figure 3(c) and Figure 3(d), one can observe that the dense connections cannot only sharpen the object boundary (e.g., the legs of the deer) but also provide complete object regions (e.g., the plant pots). These desired benefits clearly show the advantages of dense connections for allowing direct access of classifier to low- and high-level features.

C. Self-explanatory Generator

Though we have carefully designed the encoder, it presents little transparency for its internal mechanism on saliency detection. It also lacks justification on how the saliency prediction is made. To address this issue, given an image of interest, we propose a generator to produce an explicit explanation map that quantizes sensitivity of the saliency encoder over each image location thoroughly. Through analyzing the generated explanation map, we can effectively reveal the important factors that influence the saliency predictions of the encoder, providing explanations on the encoder prediction and justifying the results.

The purpose of the self-explanatory generator is to explicitly quantify how each element of the input image (or feature) \(y = \{y_1, \ldots, y_M\}\) contributes to the final saliency prediction. Let \(y_m\) denote the investigated feature element, e.g. one patch of the input image. Here \(m\) is the feature element index. Inspired by the work of \cite{32} and \cite{44}, for analyzing the effect of feature \(y_m\), the generator purposely keeps it from contributing to the encoder and only passes the other features \(y_{\sim m}\) (e.g. the other patches of the input image) to the encoder. The output is taken as prediction of the generator. Compared to the encoder prediction using all features \(y\), the prediction of the generator
be simplified as follows:

For a given pixel \( y \), the generator approximately estimates importance \( P(y_m \mid Y_m) \) for each patch \( y_m \), by sampling within the neighborhood \( y_m \) (highlighted in purple in the left). Then the features \( y \) and \( Y_m \) are passed to the encoder, giving the saliency prediction \( p(i \mid y) \) and \( p(i \mid Y_m) \) respectively. The final explanatory evidence is generated by the function of saliency explanation (FSE). See main text for more details. Best viewed in color.

Fig. 4. Diagram of the proposed self-explanatory generator within SENet. For the given image \( y \), the generator approximately estimates importance \( p(y_m \mid Y_m) \) for each patch \( y_m \), by sampling within the neighborhood \( y_m \) (highlighted in purple in the left). Then the features \( y \) and \( Y_m \) are passed to the encoder, giving the saliency prediction \( p(i \mid y) \) and \( p(i \mid Y_m) \) respectively. The final explanatory evidence is generated by the function of saliency explanation (FSE). See main text for more details. Best viewed in color.

reflects the sensitteness of encoder over feature \( y_m \). After all the features within \( y \) (that denotes the whole image) are examined by the generator, the integral results would provide interpretation on how the encoder utilizes input \( y \) for saliency detection.

In particular, we build the generator as a probabilistic model that specifies a saliency prediction \( p \) with the absence of an investigated feature \( y_m \):

\[
\text{gen}(i \mid y_m) = p(i \mid Y_m),
\]

where \( p(i \mid y) \) is the predicted saliency probability with features \( y \) at the location \( i \), and \( Y_m \) denotes the set of all features \( y \) excluding \( y_m \).

To leave \( y_m \) out cleanly from \( y \) (note it is not equivalent to naively zeroing \( y_m \)), we need to give an unknown label to \( y_m \) and retrain the saliency classifier to ignore the contribution of \( y_m \) over the saliency prediction. However, the cost of retraining a DNN model is too expensive and clearly infeasible for explaining the DNN based saliency detection models. Another approach is to simulate the absence of \( y_m \) by applying the following Bayesian rules [32]:

\[
p(i \mid Y_m) = \sum_{y_m} p(y_m \mid Y_m) p(i \mid Y_m, y_m),
\]

where \( p(i \mid Y_m, y_m) = p(i \mid y) \) is the actual saliency prediction of the encoder at the location \( i \). Because \( y_m \) is assumed to be unknown, all possible values of \( y_m \) need to be considered.

How to estimate \( p(y_m \mid Y_m) \) in Eqn. (4) is crucial for the generator to interpret the contribution of \( y_m \). During the image saliency detection, each image pixel can be regarded as a feature \( y_m \), which however makes the estimation of \( p(y_m \mid Y_m) \) impossible due to huge search space. In [32], \( y_m \) is directly assumed to be independent of \( Y_m \), and the \( p(y_m \mid Y_m) \) can be simplified as follows:

\[
p(y_m \mid Y_m) \approx p(y_m).
\]

The above approximation is simple but not accurate as it ignores the strong dependence among spatially neighboring pixels within an image. As suggested by [44], for a given pixel \( y_m \), multiple image patches \( y^h \) around it can be sampled to approximate \( y \) where \( h \) denotes the neighboring regions of \( y_m \). Hence, as shown in the left of Figure 4 we approximate \( p(y_m \mid Y_m) \) by sampling patches within \( y_m \) neighborhood:

\[
p(y_m \mid Y_m) \approx p(y_m \mid Y_m^h).
\]

Once obtaining estimate of \( p(y_m \mid Y_m) \), the generator prediction \( \text{gen}(i \mid y_m) \) is computed as follows:

\[
\text{gen}(i \mid y_m) \approx \sum_{y_m} p(y_m \mid Y_m^h) p(i \mid y).
\]

We can compare outputs of the generator and the encoder to explain reaction of the encoder at feature \( y_m \). A comparison function, called saliency explanation, is defined to measure the sensitiveness of the proposed saliency encoder on investigated feature \( y_m \):

\[
\text{FSE}(i \mid y_m) = \text{diff}(\text{enc}(i \mid y), \text{gen}(i \mid y_m)) = \log \frac{\text{enc}(i \mid y)}{1 - \text{enc}(i \mid y)} - \log \frac{\text{gen}(i \mid y_m)}{1 - \text{gen}(i \mid y_m)}.
\]

where \( \text{enc}(i \mid y) \) means the encoder prediction at location \( i \) over feature \( y \).

The \( \text{FSE}(i \mid y_m) \) in Eqn. (8) reflects how the feature \( y_m \) influences the saliency prediction of the encoder on the \( i \)-th location. \( \text{FSE}(i \mid y_m) > 0 \) means removing \( y_m \) causes the decreasing of the saliency score and \( y_m \) thus has a positive influence on the saliency prediction of the encoder. \( \text{FSE}(i \mid y_m) < 0 \) means removing \( y_m \) brings a increasing saliency score. That is, the presence of \( y_m \) may suppress the saliency prediction of the encoder. Using Eqn. (8), we get the explanation map \( \text{FSE}(y_m) \) of feature \( y_m \) at each position \( i \). After going through all features \( y \), we can get explanation maps for all the used features. To integrate them together, we sum and average all the evidence maps into the explanatory \( \text{FSE}(y) \) for \( y \).

In this paper, we focus on explaining dependence of the saliency model on the input image \( I \). Then \( \text{FSE}(I) \) is adopted for explanation. Through visualizing \( \text{FSE}(I) \), we get supportive saliency evidence map for the proposed saliency encoder. Note the self-explanatory generator can be applied for explaining any DNN based saliency detection model besides the proposed saliency encoder model in this paper. Hence, \( \text{FSE}(I) \) can be regarded as a metric to compare the capability of different models. More powerful the saliency detection model is, more reasonable the \( \text{FSE}(I) \) should be. We believe such an explanation model is of independent interest to the computer vision community to better understand predictions of “black-box” DNN models.

### IV. Experiments

#### A. Datasets

In this section, we evaluate the performance of the proposed SENet model for salient object detection on five widely used benchmark datasets: MSRA-B [25], ECSSD [33], HKU-IS [18], PASCAL-S [23] and SOD [29]. The MSRA-B dataset [25] contains 5,000 images from hundreds of different categories and most images only have one object instance.
ECSSD [33] contains 1,000 semantically meaningful but structurally complex images. HKU-IS [18] is one of the latest datasets and contains more than 4,000 challenging images, including low contrast or multiple salient objects. PASCAL-S [23] is built on the validation set of PASCAL VOC 2012 segmentation challenge which contains 850 natural images. Due to complex object context and cluttered background, PASCAL-S [23] is one of the most challenging salient object detection datasets. SOD [29] contains 300 images, most of which include multiple salient instances. All the datasets provide pixel-level human annotations on salient object.

B. Evaluation Metrics

Three widely used and standard metrics are adopted to evaluate our model, i.e., precision-recall (PR) curves, F-measure, and the mean absolute error (MAE). The PR curves depict the mean precision and recall of all testing images against the ground truth. The mean precision and recall of an image are computed by polarizing the saliency map with a threshold $T_f$ varying within the range of $[0:1:255]$ and comparing results with the ground truth. Similar to [1], an adaptive threshold $T_a$ is employed to compute the weighted F-measure. The adaptive threshold $T_a$ is defined as proportional to the mean of the saliency map $S$ as follows:

$$T_a = \frac{t}{W \times H} \sum_i \sum_j S(i,j),$$

where $t$ is typically set to be 1.5 [21]. Here $H$ and $W$ are the height and width of the image respectively.

The binary map generated by using the threshold of $T_a$ is employed to compute the precision, recall and weighted F-measure:

$$F_\omega = \frac{(1 + \omega^2) \text{Precision} \times \text{Recall}}{\omega^2 \times \text{Precision} + \text{Recall}},$$

where Precision and Recall denote the proportion of detected true positive saliency pixels as compared to the number of detected saliency pixels and the number of saliency pixels in the ground truth, respectively. The parameter $\omega^2$ is set to 0.3 for emphasizing the precision [1].

Finally, MAE is defined as the mean absolute difference between the saliency map $S$ and the ground truth $G$:

$$\text{MAE} = \frac{1}{H \times W} \sum_i \sum_j |S(i,j) - G(i,j)|.$$  

MAE considers the true negative rates while the $F_\omega$ focuses on the successful detected regions [3].

C. Implementation Details

Our proposed SENet is implemented by the publicly available Caffe library [11]. We choose ResNet-101 [6] and VGG-16 [36] pre-trained on ImageNet [4] as our basic models. For ResNet-101, the entire model is trained end-to-end. The feature maps from res2c, res3b3, res4b22 and res5c are selected and passed to the context block. Details of layer configuration for the context block are provided in Table I.

Due to the capability limitation, the end-to-end training is infeasible for VGG-16 based model. We split the training of VGG-16 based saliency model in two steps. Firstly, we train the VGG-16 like FCNs [26] with abandoning the layers of pool5, fc6 and fc7. Then the context blocks and the dense short- and long- connections are added into the pretrained model. The feature maps from conv1_2, conv2_2, conv3_3, conv4_3 and conv5_3 are fed forward to the context block. The learning rate of pre-trained layers is set to 1/100 of the newly added layers.

To ensure a fair comparison with existing approaches, we adopt a training set of 5000 images following the same setting in [18]: 2500 images from the MSRA-B dataset and 2500 images from the HKU-IS. All the model hyperparameters are tuned over a separate validation set containing 1000 images: 500 images from the MSRA-B dataset and the other 500 images from the HKU-IS. We evaluate the trained model over the datasets mentioned in Sec. IV-A. Momentum and weight decay parameters are fixed as 0.9 and 0.0005 respectively. We use the following data augmentation: random mirror and random resize between 0.5 and 1.5. Augmenting the data reduces the over-fitting risk for the network and improves the performance. For the ResNet101-based model, we train the model with a mini-batch size of 2 for 25 epochs, and set the learning rate as $1 \times 10^{-8}$. For the VGG16-based model, we first train the model with a mini-batch size of 15 for 10 epochs, and set the learning rate to $5 \times 10^{-9}$. Then we fine-tune the pre-trained model for another 20 epochs.

For saliency explanation, we employ a crop size of 224 because small crop size can reduce sampling complexity and speed up the computation in Eqn. (7). For a testing image $I$, we compute FSE($I$) in Eqn. (9) with a $16 \times 16$ image patch each time. For a testing patch, we sample its neighbors for 5 times within a $24 \times 24$ surrounding region and slide the sampling window over the whole image with a stride of 2.

D. Comparison with State-of-the-arts

We compare the proposed SENet with 11 existing DNN based salient object detection methods, including MDF [18], MC [43], DS [22], DCL [19], ELD [16], DHS [24], RFCN [40], CRPSD [38], DLS [9], NLDF [27] and DSS [8]. Besides,

| Stage | ResNet-101 | VGG-16 |
|-------|------------|--------|
| Layer | Kernel     | Layer | Kernel |
| 1     | -          | conv1_2 | 3x3, 32 |
| 2     | res2c 3x3, 256 | res2c | 3x3, 64 |
| 3     | res3b3 3x3, 256 | conv3_3 | 3x3, 128 |
| 4     | res4b22 3x3, 256 | conv4_3 | 3x3, 256 |
| 5     | res5c 3x3, 512 | conv5_3 | 3x3, 256 |
TABLE II
QUANTITATIVE COMPARISON WITH STATE-OF-THE-ARTS. THE UP-ARROW ↑ MEANS LARGER IS BETTER WHILE THE DOWN-ARROW ↓ MEANS SMALLER IS BETTER. SYMBOL † INDICATES THAT THE NETWORK IS PRE-TRAINED ON ResNet-101. For the VGG-16 based models, the top three results are highlighted in RED, GREEN and BLUE.

| Methods | Datasets | MSRA-B [25] | ECSSD [33] | HKU-IS [18] | PASCAL-S [23] | SOD [29] |
|---------|----------|-------------|-------------|-------------|-------------|---------|
|         |         | $F_F$ ↑ MAE ↓ | $F_F$ ↑ MAE ↓ | $F_F$ ↑ MAE ↓ | $F_F$ ↑ MAE ↓ | $F_F$ ↑ MAE ↓ |
| MDF [18] | 0.885 | 0.104 | 0.833 | 0.108 | 0.860 | 0.129 | 0.764 | 0.145 | 0.785 | 0.155 |
| MC [43] | 0.872 | 0.062 | 0.822 | 0.107 | 0.781 | 0.098 | 0.721 | 0.147 | 0.708 | 0.184 |
| DS [22] | - | - | 0.810 | 0.160 | - | - | 0.818 | 0.170 | 0.781 | 0.150 |
| DCL [19] | 0.916 | 0.047 | 0.898 | 0.071 | 0.907 | 0.048 | 0.822 | 0.108 | 0.832 | 0.126 |
| ELD [16] | 0.914 | 0.042 | 0.865 | 0.098 | 0.844 | 0.071 | 0.767 | 0.121 | 0.760 | 0.154 |
| DHS [21] | - | - | 0.905 | 0.061 | 0.892 | 0.052 | 0.820 | 0.091 | 0.823 | 0.127 |
| RFCN [40] | 0.926 | 0.082 | 0.889 | 0.097 | 0.895 | 0.079 | 0.827 | 0.118 | 0.805 | 0.161 |
| CRPSD [38] | - | - | 0.919 | 0.098 | 0.911 | 0.091 | 0.830 | 0.114 | - | - |
| DLS [6] | - | - | 0.852 | 0.086 | 0.835 | 0.091 | 0.752 | 0.120 | - | - |
| NLDF [27] | 0.911 | 0.069 | 0.905 | 0.093 | 0.902 | 0.075 | 0.824 | 0.120 | 0.840 | 0.168 |
| DSS [8] | 0.926 | 0.029 | 0.915 | 0.055 | 0.912 | 0.040 | 0.831 | 0.083 | 0.342 | 0.121 |
| SENet | 0.931 | 0.026 | 0.915 | 0.052 | 0.924 | 0.032 | 0.840 | 0.071 | 0.851 | 0.119 |
| DeepLab-v2† | 0.951 | 0.021 | 0.935 | 0.040 | 0.928 | 0.038 | 0.857 | 0.074 | 0.869 | 0.101 |
| SENet† | 0.957 | 0.020 | 0.945 | 0.036 | 0.935 | 0.032 | 0.865 | 0.066 | 0.877 | 0.095 |

Fig. 5. Comparison of precision-recall curves of 11 DNN based salient object detection approaches on 4 datasets. Best viewed in color.

Overall, the proposed SENet provides new state-of-the-art for DNN based salient object detection in terms of MAE, F-measure and PR curves consistently. For various benchmark datasets, such as relative simple MSRA-B or challenging PASCAL-S and SOD, the performance of the proposed SENet is superior to the previous methods, demonstrating that SENet can better detect the non-salient regions and offer better true negative rates. Besides, the proposed SENet improves the F-measure of existing best-performing approaches by 0.5%, 1.3%, 1.1% and 1.1% respectively on MSRA-B, HKU-IS, PASCAL-S and SOD. CRPSD [38] achieved the best F-measure of ECSSD which outperforms SENet by 0.4%. Based on the pre-trained model ResNet-101 [6], the proposed method outperforms DeepLab-v2 [2] by an average margin of 9.5% and 0.8% for MAE and F-measure respectively. It is because our model densely connects low- and high-level context which makes the classifier have global and local connections with the computed features. Though the dilated convolution [2] is proposed to maintain high-resolution feature maps, the connection between classifier and feature maps is diluted by the operation of dilation [31]. The local connected classifier loses its global perspective which could hurt performance for detecting salient object out of the receptive field.

We show the results of PR curves in Figure 5. The PR curves depict the rates of true positive. Usually, higher precision and slower attenuation of the curve indicate better capability of the salient object detection model. The ResNet-101 based SENet† outperforms other compared methods significantly on all the benchmark datasets. For the VGG-16 based models, SENet preserves the precision at the highest level with increased recall. For the HKU-IS and PASCAL-S datasets, at high level of recall (recall>0.9), DCL [19] and NLDF [27] can provide better precision. Though DSS [8] achieves the second best performance in terms of the MAE and F-measure, NLDF [27] goes beyond DSS [8] under the evaluation of PR curves.

we train a ResNet-101 based DeepLab-v2 [2] model for salient object detection to compare with our ResNet-101 based SENet. We replace the softmax layer of DeepLab-v2 [2] with a sigmoid cross entropy layer for saliency prediction. Besides, fully connected conditional random field (CRF) [13] is employed for post-processing the results.

1) Quantitative comparison: In Table II we quantitatively compare state-of-the-arts in term of maximum MAE and F-measure. For VGG-16 [36] based models, the proposed SENet outperforms other compared approaches in terms of MAE, improving existing best results by 10.3%, 5.4%, 20.0%, 14.5% and 1.7% respectively on MSRA-B, ECSSD, HKU-IS, PASCAL-S and SOD. The much smaller MAE of SENet demonstrates that SENet can better detect the non-salient regions and offer better true negative rates. Besides, the proposed SENet improves the F-measure of existing best-performing approaches by 0.5%, 1.3%, 1.1% and 1.1% respectively on MSRA-B, HKU-IS, PASCAL-S and SOD. CRPSD [38] achieved the best F-measure of ECSSD which outperforms SENet by 0.4%. Based on the pre-trained model ResNet-101 [6], the proposed method outperforms DeepLab-v2 [2] by an average margin of 9.5% and 0.8% for MAE and F-measure respectively. It is because our model densely connects low- and high-level context which makes the classifier have global and local connections with the computed features. Though the dilated convolution [2] is proposed to maintain high-resolution feature maps, the connection between classifier and feature maps is diluted by the operation of dilation [31]. The local connected classifier loses its global perspective which could hurt performance for detecting salient object out of the receptive field.

We show the results of PR curves in Figure 5. The PR curves depict the rates of true positive. Usually, higher precision and slower attenuation of the curve indicate better capability of the salient object detection model. The ResNet-101 based SENet† outperforms other compared methods significantly on all the benchmark datasets. For the VGG-16 based models, SENet preserves the precision at the highest level with increased recall. For the HKU-IS and PASCAL-S datasets, at high level of recall (recall>0.9), DCL [19] and NLDF [27] can provide better precision. Though DSS [8] achieves the second best performance in terms of the MAE and F-measure, NLDF [27] goes beyond DSS [8] under the evaluation of PR curves.

Overall, the proposed SENet provides new state-of-the-art for DNN based salient object detection in terms of MAE, F-measure and PR curves consistently. For various benchmark datasets, such as relative simple MSRA-B or challenging PASCAL-S and SOD, the performance of the proposed SENet
is impressive.

2) Visual comparison: In Figure 6, we present three typical challenging cases in salient object detection, including: low contrast, complex background and multiple salient objects. The low contrast case presents a scenario where the salient object has similar color or texture to the background. As shown in the top two rows of Figure 6, the target object has similar visual features as the background which makes it hard to be segmented accurately. For this challenging case, most compared methods except CRPSD [38] fail to detect the integrated salient objects. The proposed SENet detects the salient object entirely and provides sharp boundaries along the salient object. The second challenging case we consider is from complex background, as shown in the middle of Figure 6. The background presents complex structure which easily leads to false positive detection. For example, the flower in the third row of Figure 6 has bright color which increases error for all the compared methods. For this example, the low-level features cannot work well. However, our proposed SENet can address this challenging case well. Its superior performance benefits from its internal dense connections of multi-level features and exploiting the high-level feature—the context information of the bee—to detect the salient object. The last challenging scenario presents multiple salient objects. Some models, such as DHS [24], DCL [19], NLDF [27] and DSS [8] severely miss parts of the salient object while others (e.g., MDF [18]) incorrectly include the background regions into the detection results. By contrast, the proposed SENet detects each object instance accurately.

In summary, the visual results in Figure 6 illustrate that the proposed SENet can handle various challenges for salient object detection better than multi-networks architecture [18], [38] or multi-stages refinement [19], [40]. We attribute this superiority to the dense connections of classifier on low- and high-level feature maps as well as the specifically designed context blocks.

3) Comparison on object boundary detection: To demonstrate the proposed SENet effectively exploits features from low-level layers to better localize boundaries, we quantitatively evaluate the performance of different models for the salient object boundary detection. The evaluation setting is similar to [2], [13]. We add a narrow band (called trimap in Figure 7(a)) surrounding actual object boundaries which are inferred from the given ground truth. Then we compute the F-measure for these pixels lying within the narrow band. Based on the results in Table II we select the best-performing DHS [24], DCL [19], CRPSD [38], NLDF [27], DSS [8] and DeepLab-v2 [2] as the baselines for comparison. To fairly compare the effect along object boundaries, we do not apply any post-processing over all the results. As shown in Figure 7, exploiting powerful pre-trained network and dense connections with multi-levels features enables SENet†
to outperform all compared methods across all the trimap widths and datasets. The VGG16-based SENet has similar performance as DCL [19] and CPRSD [38] which both benefit from the superpixel-based local network component. The proposed dense structure performs competitively on localizing the object boundaries without introducing additional network. Such efficiency makes the SENet architecture very appealing in practice. In the next subsection, we will further explain the effectiveness of SENet by visualizing the saliency explanation extracted by the self-explanatory generator.

E. Explanation on Saliency Detection

For a given image \(I\), we compute and visualize the saliency explanation \(FSE(I)\) in Eqn. (8) to explain the behavior of proposed saliency encoder. We also compare capability of different DNN based saliency detection models using such evidence. A testing dataset of 200 images is built by extracting 50 images randomly from ECSSD, HKU-IS, PASCAL-S and SOD respectively. MSRA-B is excluded due to the training set of DHS [24] including the entire MSRA-B. Three DNN based approaches, i.e., DHS [24], DCL [19] and DSS [8], are chosen to compare with SENet over the built datasets. All methods are evaluated without any post-processing for fair comparison. The F-measure of DHS [24], DCL [19], DSS [8] and SENet over the datasets is 0.842, 0.850, 0.865 and 0.877 respectively.

Figure 8 collects and shows statistics of \(FSE(I)\) for different approaches. For each given image \(I\), we calculate the mean and standard deviation (STD) of \(FSE(I)\) and plot its distribution in descending order of the mean values from SENet. For clear illustration, we compare SENet with DHS [24], DCL [19] and DSS [8] respectively in Figure 8. Comparing results shown in the three figures, one can observe that the better the saliency model is (based on the F-measure), the larger variance of \(FSE(I)\) the model intends to provide, especially in Figure 8(a). The average STD of positive and negative evidence by SENet is 0.0647 and 0.0681 while these number by DHS [24] is 0.0376 and 0.0410. The gap of saliency detection performance between DSS [8] and SENet is smaller than the one of DHS [24]. Hence DSS [8] also has a larger average STD of \(FSE(I)\) (0.0522 and 0.0581) than DHS [24]. The larger variance of \(FSE(I)\) means the saliency explanation of interesting model is more sensitive. A better saliency model intends to offer more information in its corresponding saliency explanation.

Figure 9 visualizes the \(FSE(I)\) of DHS [24], DCL [19], DSS [8] and the proposed SENet. The red areas indicate the regions that provide positive support for salient object detection while the blue ones represent the regions whose presence could suppress and distract saliency detection. The white color areas mean no significant sensitivity for salient object detection. For the example, in the top tow rows, the \textit{boat} is easy to detect for all the models. However, the saliency explanation varies across different models. DSS [8] and the proposed SENet give heavy negative credits on surroundings of the boat. In contrast, DCL [19] focuses equally on the salient and non-salient regions. For the second example, DHS [24] and DCL [19] regard the \textit{wall} as positive evidence for saliency detection. This well explains why these two methods detect the wall as the saliency regions in the final saliency maps. In the last example, the \textit{bird} in the image has similar visual contrast as the background where all the models label the region around the bird as negative saliency evidence. It indicates that removing this region would help saliency detection. Compared to other three models, the proposed SENet concentrates more on the correct saliency evidence. This leads to more accurate saliency map for SENet. Observing these explanation results in Figure 1 and Figure 9, one can find that a good saliency detection model prefers giving positive emphasis to the context of salient object as useful saliency clues, and regards surroundings of the salient object as negative factors that restrains the range of the salient regions.

Some failure cases of the proposed SENet are shown in Figure 10. We visualize the saliency explanation to interpret why SENet fails to detect these salient objects. For the example, in the first row, the true salient region is regarded as highly negative evidence for saliency detection. SENet regards the meaningless sky as the supportive clues for saliency predictions. Without other distinct object context, only the sky evidence is not enough to detect salient object. In the second example, SENet shows to focus more on the objects that have higher contrast with the background, such as the \textit{black mouse}. The real saliency object, i.e., the monitor, is regarded as negative for detection. We attribute such failure to the fact that the objects have more attractive and higher contrast in the testing image.

F. Ablation Study

To analyze the contributions of each components in the proposed method, we evaluate several variants of SENet.
Fig. 8. Statistical distribution of $FSE(I)$ by different methods. Over the new-built datasets of 200 images from ECSSD, HKU-IS, PASCAL-S and SOD respectively, we calculate the mean and standard deviation (STD) of $FSE(I)$ over each examples and plot the distribution in descending order of the mean values of SENet. Three DNN based approaches, i.e., DHS [24], DCL [19] and DSS [8], are chosen to compare with the proposed SENet. All methods are evaluated without any post-processing which aims to fairly compare the capability of the saliency detection model. Best viewed in color.

(a) DHS [24] vs. SENet  (b) DCL [19] vs. SENet  (c) DSS [8] vs. SENet

Fig. 9. Visual comparison of saliency results and explanation by different methods. In the first, third and fifth rows, we visualize the saliency explanation that interprets how the saliency prediction made by (b) DHS [24], (c) DCL [19], (d) DSS [8], and (e) proposed SENet. The red areas mean these regions provide positive support for salient object detection while the blue ones represent that the existing of this region could suppress and distract saliency detection. The white color areas mean no significant sensitivity for salient object detection. The second, forth and last rows show the saliency detection results without post-processing. Best viewed in color.

(a) Source  (b) DHS [24]  (c) DCL [19]  (d) DSS [8]  (e) SENet

we separately evaluate performance of the models with only short or long range dense connections. The baseline for short connections (baseline$_s$ in Table III) is a network with architecture similar to FCN-32s [26]. Then we up-sample the feature maps from res5c by bilinear interpolation. Before combining with res4b22 by short connection, a $3 \times 3$ convolutional layer is adopted to fine-tune the features. Similarly, different feature maps from res3b3, res2c and conv1 are gradually combined what are denoted as No.3, No.4 and No.5 in Table III. We find that combining with res4b22 via short connections can boost the baseline most significantly (bringing 1.34% improvement). Further combination with conv1 can

Fig. 10. Some failure cases of SENet. (a) Testing images. (b) Ground truth. (c) Saliency maps. (d) Saliency explanation. Best viewed in color.

(a)  (b)  (c)  (d)

TABLE III

| No. | settings | $F_{\omega}$ |
|-----|----------|--------------|
| (a) Comparison of short connections: |
| 1  | baseline$_s$ | 0.830 |
| 2  | + res4b22 | 0.843 |
| 3  | + res4b22 + res3b3 | 0.849 |
| 4  | + res4b22 + res3b3 + res2c | 0.852 |
| 5  | + res4b22 + res3b3 + res2c + conv1 | 0.853 |
| (b) Comparison of long connections: |
| 6  | baseline$_L$ (No. 4 setting) | 0.852 |
| 7  | dense connections only on res2c | 0.857 |
| 8  | dense connections on all stages | 0.861 |
| (c) Comparison of context block: |
| 9  | baseline$_C$ (No. 4 setting) | 0.852 |
| 10 | + context block | 0.856 |
| 11 | + context block + No. 8 setting | 0.864 |

with various design options. All the ablation experiments are conducted based on the ResNet-101 [6] that can be easily trained end-to-end.

To investigate the effectiveness of the dense connections,
only improve the performance incrementally. Therefore, we select the setting of No.4 in Table III as our basic architecture.

In Table III(b), we compare two different settings of the long-range connections. For the No. 7 setting in Table III(b), we exploit sparse long connections by concatenating the up-sampled res5c, res4b22 and res3b3 with res2c. This strategy can improve the \( F_{\text{v}} \) by 0.5%. Such results confirm that increasing connection of classifier to the feature map is beneficial for salient object detection. We further improve the connections by densely concatenating all computed features as in shown in Figure 2. We observe 0.9% improvement on PASCAL-S [29] by exploiting dense connections.

In Table III(c), we demonstrate the effectiveness of the context block. We replace the 3 × 3 convolutional layer after each stage with the proposed context block as shown in Figure 2. Before the dense connections, all the features are encoded again by the context block. This can improve performance of the baselineC to 0.858. Overall, the proposed dense connections and context can improve the baseline by 3.4%.

G. Running Time

It takes about 20 hours (14 hours) to train the VGG-16 (ResNet-101) based SENet on a single NVIDIA TITAN X GPU and a 3.4 GHz Intel processor. In the phase of inference, it takes about 0.17 second (0.25 second) for the VGG-16 (ResNet-101) based SENet to process an image of size \( 417 \times 417 \). Under the same setting to process an image, the running time of state-of-the-art DSS [8] and DCL [19] is about 0.21 second and 1.1 second respectively. Besides, the CRF-based post-processing [13] requires extra 0.5 second per image.

V. CONCLUSION

We proposed the first Self-Explanatory salient object detection Network (SENet) to provide improved saliency detection method and interpretable evidence. The improved saliency detection method, i.e., the saliency encoder, widens the access of classifier to multi-level clues by reusing multi-level features through introducing dense short- and long-range connections. These dense connections facilitate the encoder to make saliency decisions globally and locally. The self-explanatory generator offers supportive explanation to interpret the saliency decisions by purposely preventing the features of interest from contributing to the saliency encoder. Through visualization, the generator can also compare the capability of different DNN based saliency detection models. Experimental results on five benchmark datasets and the visualized saliency explanation demonstrate the superior performance of the proposed SENet.

REFERENCES

[1] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, “Frequency-tuned salient region detection,” in IEEE CVPR, 2009, pp. 1597–1604.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” arXiv preprint arXiv:1606.00915, 2016.

[3] M.-M. Cheng, J. Warrell, W.-Y. Lin, S. Zheng, V. Vineet, and N. Crook, “Efficient salient region detection with soft image abstraction,” in IEEE ICCV, 2013, pp. 1529–1536.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in IEEE CVPR, 2009, pp. 248–255.

[5] L. Gao, M. Wang, D. Tao, R. Ji, and Q. Dai, “3-d object retrieval and recognition with hypergraph analysis,” IEEE TIP, vol. 21, no. 9, pp. 4290–4303, 2012.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in IEEE CVPR, 2016, pp. 770–778.

[7] ——, “Identity mappings in deep residual networks,” in ECCV, 2016, pp. 630–645.

[8] O. Hou, M.-M. Cheng, X.-W. Hu, A. Borji, Z. Tu, and P. Torr, “Deeply supervised salient object detection with short connections,” in IEEE CVPR, 2017.

[9] P. Hu, B. Shuai, J. Liu, and G. Wang, “Deep level sets for salient object detection,” in IEEE CVPR, 2017.

[10] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in IEEE CVPR, 2017.

[11] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in ACM Multimedia, 2014, pp. 675–678.

[12] J. Kim and V. Pavlovic, “A shape-based approach for salient object detection using deep learning,” in ECCV, 2016, pp. 455–470.

[13] V. Koltun, “Efficient inference in fully connected crfs with gaussian edge potentials,” in NIPS, 2011.

[14] J. Kuen, Z. Wang, and W. Wang, “Recurrent attentional networks for salient object detection,” in IEEE CVPR, 2016, pp. 3668–3677.

[15] C.-Y. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu, “Deeply-supervised nets.” in AISTATS, vol. 2, no. 3, 2015, p. 5.

[16] G. Lee, Y.-W. Tai, and J. Kim, “Deep saliency with encoded low level distance map and high level features,” in IEEE CVPR, 2016, pp. 660–668.

[17] G. Li, Y. Xie, L. Lin, and Y. Yu, “Instance-level salient object segmentation,” in IEEE CVPR, 2017.

[18] G. Li and Y. Yu, “Visual saliency based on multiscale deep features,” in IEEE CVPR, 2015, pp. 5455–5463.

[19] ——, “Deep contrast learning for salient object detection,” in IEEE CVPR, 2016, pp. 478–487.

[20] ——, “Visual saliency detection based on multiscale deep cnn features,” IEEE TIP, vol. 25, no. 11, pp. 5012–5024, 2016.

[21] H. Li, H. Lu, Z. Lin, X. Shen, and B. Price, “Inner and inter label propagation: Salient object detection in the wild,” IEEE TIP, vol. 24, no. 10, pp. 3176–3186, 2015.

[22] X. Li, L. Zhao, L. Wei, M.-H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang, “Deepsaliency: Multi-task deep neural network model for salient object detection,” IEEE TIP, vol. 25, no. 8, pp. 3919–3930, 2016.

[23] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, “The secrets of salient object segmentation,” in IEEE CVPR, 2014, pp. 280–287.

[24] N. Liu and J. Han, “Dhsnet: Deep hierarchical saliency network for salient object detection,” in IEEE CVPR, 2016, pp. 678–686.

[25] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, “Learning to detect a salient object,” IEEE TPAMI, vol. 33, no. 2, pp. 353–367, 2011.

[26] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in IEEE CVPR, 2015, pp. 3431–3440.

[27] Z. Luo, A. Mishra, A. Achkar, J. Eichel, S. Li, and P.-M. Jodoin, “Non-local deep features for salient object detection,” in IEEE CVPR, 2017.

[28] V. Mahadevan and N. Vasconcelos, “Saliency-based discriminant track- ing,” in AISTATS, 2009, pp. 1007–1013.

[29] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in IEEE ICCV, 2001, pp. 439–446.

[30] S. J. Oh, R. Benenson, A. Khoreva, Z. Akata, M. Fritz, and B. Schiele, “Exploiting saliency for object segmentation from image level labels,” in IEEE CVPR, 2017.

[31] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, “Large kernel matters—improve semantic segmentation by global convolutional network,” in IEEE CVPR, 2017.

[32] M. Robnik-ˇSikonja and I. Kononenko, “Explaining classifications for individual instances,” IEEE TKDE, vol. 20, no. 5, pp. 589–600, 2008.

[33] J. Shi, Q. Yan, L. Xu, and J. Jia, “Hierarchical image saliency detection on extended cbsd,” IEEE TPAMI, vol. 38, no. 4, pp. 717–729, 2016.
[34] A. Shrikumar, P. Greenside, and A. Kundaje, “Learning important features through propagating activation differences,” *arXiv preprint arXiv:1704.02685*, 2017.

[35] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint arXiv:1312.6034*, 2013.

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

[37] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, “Striving for simplicity: The all convolutional net,” *arXiv preprint arXiv:1412.6806*, 2014.

[38] Y. Tang and X. Wu, “Saliency detection via combining region-level and pixel-level predictions with cnns,” in *ECCV*, 2016, pp. 809–825.

[39] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, “Deep networks for saliency detection via local estimation and global search,” in *IEEE CVPR*, 2015, pp. 3183–3192.

[40] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, “Saliency detection with recurrent fully convolutional networks,” in *ECCV*, 2016, pp. 825–841.

[41] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan, “Object region mining with adversarial erasing: A simple classification to semantic segmentation approach,” in *IEEE CVPR*, 2017.

[42] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *ECCV*, 2014, pp. 818–833.

[43] R. Zhao, W. Ouyang, H. Li, and X. Wang, “Saliency detection by multi-context deep learning,” in *IEEE CVPR*, 2015, pp. 1265–1274.

[44] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, “Visualizing deep neural network decisions: Prediction difference analysis,” *arXiv preprint arXiv:1702.04595*, 2017.