A dilated inception network for visual saliency prediction

Yang, Sheng; Lin, Guosheng; Jiang, Qiuping; Lin, Weisi

2020

https://hdl.handle.net/10356/143872

https://doi.org/10.1109/TMM.2019.2947352

A Dilated Inception Network for Visual Saliency Prediction

Sheng Yang [ID], Guosheng Lin [ID], Qiuping Jiang [ID], and Weisi Lin [ID], *Fellow, IEEE*

Abstract—Recently, with the advent of deep convolutional neural networks (DCNN), the improvements in visual saliency prediction research are impressive. One possible direction to approach the next improvement is to fully characterize the multi-scale saliency-influential factors with a computationally-friendly module in DCNN architectures. In this work, we propose an end-to-end dilated inception network (DINet) for visual saliency prediction. It captures multi-scale contextual features effectively with very limited extra parameters. Instead of utilizing parallel standard convolutions with different kernel sizes as the existing inception module, our proposed dilated inception module (DIM) uses parallel dilated convolutions with different dilation rates which can significantly reduce the computation load while enriching the diversity of receptive fields in feature maps. Moreover, the performance of our saliency model is further improved by using a set of linear normalization-based probability distribution distance metrics as loss functions. As such, we can formulate saliency prediction as a global probability distribution prediction task for better saliency inference instead of a pixel-wise regression problem. Experimental results on several challenging saliency benchmark datasets demonstrate that our DINet with proposed loss functions can achieve state-of-the-art performance with shorter inference time.

Index Terms—Visual attention, saliency detection, eye fixation prediction, convolutional neural networks, dilated convolution, inception module.

I. INTRODUCTION

VISUAL attention mechanism refers to the ability of Human Vision System (HVS) to automatically select the most salient or interested regions from natural scenes by filtering out redundant and unimportant visual information for further processing. Around 10^8–10^9 bits per second of visual data enters into our eyes as reported in [1]. Without the help of visual attention mechanism, the HVS is impossible to handle and process this large volume of data in real-time. Therefore, it is important to understand and simulate the behavior of visual attention to advance a wide range of visual-oriented multimedia applications such as image retrieval [2], image retargeting [3], video summarization [4], image and video compression [5], [6], visual quality assessment [7]–[9], object detection [10]–[12], virtual reality content design [13], and more.

In general, visual attention is stimulated by two types of factors: bottom-up and top-down. Bottom-up saliency-driven attention, which is derived directly from the distinctiveness of visual stimuli, helps people to rapidly focus on conspicuous points/regions automatically. In contrast, top-down attention is task-driven and usually can help people to deal with specific visual tasks.

This paper focuses on modeling the task-free bottom-up visual attention mechanism by predicting human eye fixations on natural images. The study of this visual attention modeling, commonly referred as visual saliency prediction/detection, is an active problem in the field of computer vision and neuroscience. Typically, a saliency map, where a pixel with brighter intensity indicates a higher probability of attracting human attention, is generated as the output of the developed visual saliency detection models.

Most of classic bottom-up saliency prediction models [14]–[16] are biologically inspired. They mainly adopt multiple low-level hand-crafted features, such as intensity, color, and so on, and combine these features in a heuristics way (e.g. center-surround contrast [14], graph-based random walk [15], etc.). However, these low-level hand-crafted features and their heuristics combination are insufficient to represent the wide variety of factors that contribute to visual saliency [17]–[19].

With the advent of Deep Convolutional Neural Networks (DCNN), the feature extraction and combination could be formulated in a data-driven manner through fully end-to-end training. At present, DCNN-based saliency models have defeated the classical saliency prediction models in all challenging saliency datasets [20]–[22]. Within these DCNN-based models, the use of multi-scale contextual features [17], [18], [23], [24], which aims to characterize the diverse saliency-influential factors at different receptive field sizes, makes them stand out. However, these state-of-the-art saliency models suffer from the huge computation cost by fully exploiting these comprehensive feature representations.

In this work, we propose a DCNN architecture called Dilated Inception Network (DINet) for bottom-up visual saliency prediction. In order to fully exploit the multi-scale contextual features, an efficient yet effective dilated inception module (DIM) is involved. The original inception module [25] utilizes multiple convolutional layers with different kernel sizes to serve as multi-scale feature extractors with various receptive fields. In contrast, our DIM uses parallel dilated convolutions with different dilation rates [26] to capture more comprehensive and effective multi-scale contextual features with much less computation cost. In general, our DINet can be decomposed into two parts: encoder and decoder networks. The DCNN-based backbone network is paired with our DIM to serve as the encoder. Then, the multi-scale features are forwarded to a simple yet effective fully convolutional decoder network for saliency inference.

A recent study [27] shows that training a DCNN-based saliency model with their softmax normalization-based probability distribution (PD) distance metrics results in superior performance with respect to commonly-used pixel-wise regression loss functions. Instead, we further propose a set of linear normalization-based PD distance metrics as the new learning objectives to outperform both of them. As demonstrated in our ablation experiments, the saliency prediction model trained with our loss functions achieves better performance than the same architecture trained with either softmax normalization-based loss functions or standard regression loss functions.

The performance of our DINet is evaluated on various saliency benchmark datasets. The peer comparison results indicate that our DINet can achieve state-of-the-art performance in terms of both efficiency and efficacy. The source code of the DINet and its pre-trained model are publicly available.[1]

In summary, our main contributions are threefold:

- We propose an efficient and effective dilated inception module (DIM) to capture the multi-scale contextual features. The scale diversity is enriched by introducing paralleled dilated convolutions with various dilation ratios at lower computation cost. Moreover, the effectiveness of this DIM can be verified by our proposed visualization method and ablation analysis.
- A set of linear normalization-based probability distribution distance metrics are proposed as loss functions to optimize our DINet. They provide an additional linear regularization leading to a promising performance gain.
- The computation cost is further reduced by replacing the deconvolution layers with a fully convolutional decoder structure. As a result, the whole model is efficient to achieve real-time performance.

The rest of this paper is organized as follows. The related works on visual saliency prediction are summarized in Section II. The proposed DINet and optimization method are illustrated in Section III. The detail analysis and the peer comparison on public benchmarks will be provided in Section IV, and the conclusion is given in Section V.

[1][Online]. Available: https://github.com/ysyscool/DINet

II. RELATED WORK

In this section, we first review the previous saliency prediction models with deep learning architectures. Then, we particularly summarize the existing deep saliency models with multi-scale feature extraction modules.

A. Deep Learning-Based Visual Saliency Prediction

Nowadays, the advances in deep learning have already boosted the progress in saliency prediction. To the best of our knowledge, the first attempt to use convolutional neural networks to predict visual saliency was introduced by Vig *et al.* in 2014 [28]. Their model, called eDN, consists of three individual and different shallow networks (from 1 layer to 3 layers) for feature extraction. However, this model is inferior to some traditional unsupervised saliency models [15], [16] mainly due to the limited depth of their networks. After that, researchers seek to use deeper models (e.g. AlexNet [29] in [30], [31], VGGNet [32] in [17], [24], and ResNet [33] in [18], [19].) and utilize the fully convolutional network (FCN) [34] framework for fully leveraging the powerful capabilities of DCNN models in contextual feature extraction.

Currently, DCNN models utilize some down-sampling operations (e.g. max pooling and convolutions with strides) to reduce the computation cost and enlarge the receptive field in their subsequent layers. Here, we denote the ratio of the input image spatial resolution to the output resolution of DCNN by *output_stride* for simplification. The more usage of down-sampling operations, the higher *output_stride* is. However, higher *output_stride* also means the feature maps in the top layers have a relatively smaller spatial resolution. Such limited spatial information cannot support effective dense prediction of saliency [18], [19]. A naive approach, presented in ML-Net [35] and MxSalNet [23], to increase the spatial resolution in top layers is simply removing some down-sampling operations in some of the layers. But this approach will unavoidably reduce the receptive field size in subsequent layers. Since the size of the receptive field affects the amount of contextual information which is essential to the final saliency inference, such reduction in receptive field size is suboptimal. Therefore, a trade-off between the spatial resolution of feature maps and the computation cost should be guaranteed while maintaining suitable receptive field sizes. Therefore, several state-of-the-art deep saliency prediction models [17]–[19] adopt dilated convolution [26], [36], [37] strategy to increase the receptive field sizes of the top layers, compensating for the reduction in receptive field size induced by removing down-sampling operations.

Previous studies [17], [24] demonstrated that multi-scale contextual features are essential to the visual saliency prediction problem. In fact, the foundation for this conclusion is from the intuition that visual information is processed at various scales by human eyes [25], [40]. Table I provides a comparison of recent deep saliency models and our proposed model. The models with multi-scale inputs will integrate multi-scale contextual features while some models with single input still can capture these due to their architectures, as detailed in the next section.

Model	Backbone network	*output_stride*	Input/Inputs	Multi-scale	Loss function	pixel/PD	Center-bias
SALICON [31]	AlexNet/VGG16/GoogleNet	16	multi inputs	yes (IPN)	KLD	PD	implicit
DeepGazeII [38]	VGG19	16	single input	no	BCE (softmax)	PD	explicit
PDP [27]	VGGNet	N/A	single input	no	statistical distances (softmax)	PD	implicit
ML-Net [35]	VGG16	8	single input	yes (Skip-layer)	Euclidean distance	pixel	explicit
SAM [19]	VGG16/ResNet50	8	single input	no	KLD + NSS	PD + pixel	explicit
SALGAN [39]	VGG16	16	single input	no	BCE	pixel	implicit
DeepFix [17]	VGG16	8	single input	yes (Inception)	Euclidean distance	pixel	explicit
DSCLRCN [18]	VGG16/ResNet50 + Places-CNN	8	multi inputs	yes (Skip-layer)	NSS	pixel	implicit
DVA [24]	VGG16	16	single input	yes (Skip-layer)	BCE	pixel	implicit
MxSalNet [23]	VGG16	8	single input	yes (Skip-layer)	Euclidean distance + CCE	pixel	explicit
DINet (Ours)	ResNet50	8	single input	yes (Inception)	statistical distances (linear)	PD	implicit

KLD: Kullback-Leibler divergence, PD: probability distribution, BCE: binary cross entropy, N/A: not available, NSS: normalized scanpath saliency, CCE: categorical cross entropy.
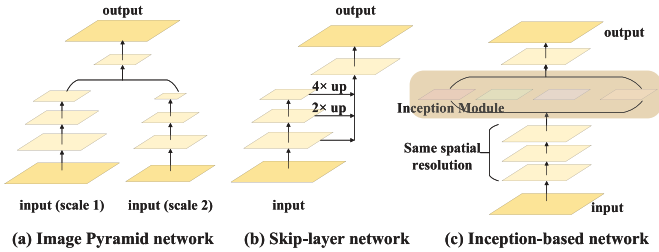

Fig. 1. The illustration of existing deep learning architectures to capture multi-scale information in saliency prediction.

As for loss function, most of the existing DCNN-based saliency models directly use the typical pixel-wise classification or regression loss functions whereas saliency prediction is evaluated on the whole saliency maps. In [27], Jetley *et al.* propose to use loss functions based on PD distances with softmax normalization for training saliency models. Their results demonstrate the improvement by considering saliency maps as probability distributions.

Regarding the center-bias phenomenon, some of the saliency models learn the center-bias explicitly by their designed modules, such as the location biased convolutional layer in DeepFix [17]. However, with the help of large-scale dataset–SALICON [20], DCNN-based saliency models can learn this bias implicitly and solely from the training data [27], [41].

B. Existing Multi-Scale Feature Extraction Deep Learning Architectures

In Fig. 1, we summarize the existing deep architectures aiming at capturing multi-scale contextual features in saliency prediction. These models can be roughly classified into three categories: i) Image Pyramid Network; ii) Skip-layer Network; and iii) Inception based Network.

1) Image Pyramid Network: The most straightforward way to learn multi-scale feature representations can be found in [31], [42]. Their idea is to apply duplicate or multiple feature extractor networks with the multi-scale inputs, as shown in Fig. 1(a). The outputs of this image pyramid network (IPN) are merged and fed into the following decoder network to generate the final saliency map. Such architectures with multi-scale inputs indeed can learn the multi-scale contextual features. Nevertheless, training and

testing these models are not economic in both computation cost and memory usage.

2) Skip-Layer Network: Due to the down-sampling operations in the common backbone networks, the output of each convolutional blocks is usually in different spatial resolution. The first several convolutional blocks learn the low-level image features while the features learned from the deeper blocks will contain semantic information and discriminative pattern with various receptive fields [43]. Based on this principle, architectures with skip-layers have been proposed in [24], [30], [35]. Skip-layer network captures multi-scale contextual features by concatenating the outputs of different layers with increasingly larger receptive fields and *output_stride*, as illustrated in Fig. 1(b). More importantly, the skip-layer network can efficiently utilize intermediate features while the conventional way only utilizes the topmost features. Despite the high efficiency, a main problem in the skip-layer network is that spatial information gradually reduced in the higher layers due to the useage of down-sampling operations. Direct up-sampling and concatenating these feature maps from different layers without feature adaptation will bring uncertainty and ambiguity into the saliency inference.

3) Inception-Based Network: As demonstrated in Fig. 1(c), inception-based network, as discussed in the DeepFix model [17], avoid the above problem by utilizing the dilated convolutions and removing some down-sampling operations in the backbone network. Therefore, its output still has sufficient spatial information to support the dense prediction. Inception modules, proposed in the well-known GoogleNet [25], are attached to the top of the backbone network to capture multi-scale contextual features. The main idea of inception module is to use convolutions with multiple kernel sizes. However, existing inception module is not very economic in both computation and optimization. Our work is based on this type of network. Specifically, we revise the original inception module to have more powerful multi-scale feature extraction capacity in a computationally-friendly manner, as will be presented in Section III-C. It should be noted that, in addition to the GoogleNet [25], our dilated inception module also take the advantage of the atrous spatial pyramid pooling (ASPP) module in the DeepLab model [26], which has succeeded in semantic segmentation. We apply those parallel dilated convolutional layers to form our dilated inception module and thus obtain the state-of-the-art performance in saliency prediction.
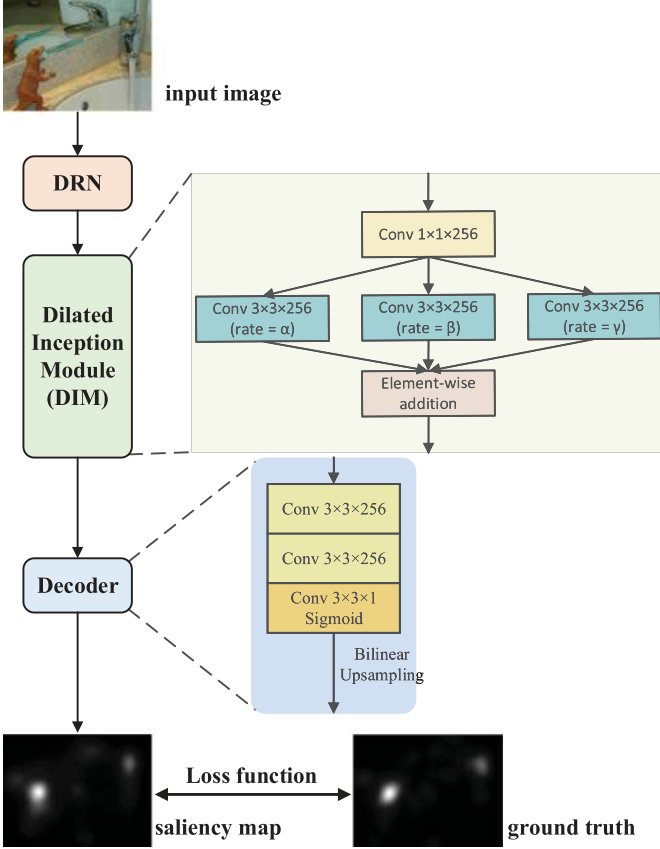
Fig. 2.   The architecture of our proposed DINet saliency prediction model.

Fig. 3.   A comparison between standard convolution (a) and dilated convolution (b).

III. OUR APPROACH

In this section, we illustrate the architecture of our DCNN-based saliency prediction model–DINet (Dilated Inception Network). The whole model is depicted in Fig. 2. Our model starts from the Dilated Residual Network (DRN) [37] which is used as the primary feature extractor to extract dense feature maps with relatively larger spatial resolution. We propose to attach an effective dilated inception module to the top of DRN for capturing the multi-scale features. A simple yet effective decoder network is employed at the end for converting these features into the saliency maps. Furthermore, since the saliency map can be viewed as a probability distribution, we propose a set of linear normalization-based probability distribution distance metrics for training our DINet to better measure the gaps between our saliency predictions and ground-truths.

A. Dilated Convolution and Dilated Residual Network

1) Dilated Convolution: The main idea of dilated convolution is to insert holes(zeros) in convolutional kernels to increase the receptive field, thus enabling dense feature extraction in DCNN. Since the usage of dilated convolutions is the core of our model, we simply revisit its concept and properties here.

In general, for each spatial location i, dilated convolution is defined as:
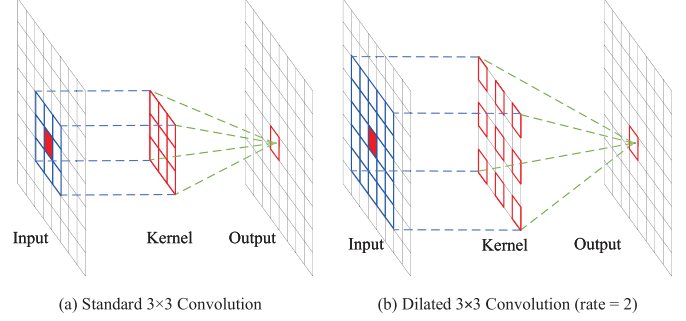
$$y[i] = \sum_l x[i + r \cdot l]w[l], \qquad (1)$$

where $y[i]$ and $x[i]$ denote the output and input on location i, respectively. w is the convolutional filter and r is the dilation rate to sample the input. Dilated convolution is implemented by inserting $r - 1$ zeros between two consecutive spatial positions in the original filter w along each spatial dimension. For a $k \times k$ convolutional kernel, the actual size of the dilated convolutional kernel is $k_d \times k_d$, where $k_d = k + (k - 1) \cdot (r - 1)$. It should be noted that dilated convolution still only have $k \times k$ meaningful kernel parameter. The standard convolution is a special case of dilated convolution with $r = 1$. A comparison between standard convolution and dilated convolution is illustrated in Fig. 3. It is obvious that a dilated 3×3 convolutional kernel with $r = 2$ sample the feature maps like a 5×5 standard convolutional kernel, which means the receptive field of the outputs after these two kernels is roughly the same. With this observation, we can arbitrarily change the field-of-view of dilated convolutional kernels via choosing different dilation rate under the same number of parameters. By incorporating dilated convolutions into the encoder network, the dilated encoder network is capable of preserving the spatial resolution and compensate the receptive field reduction caused by removing some pooling or stride convolutional layers in the original encoder network.

2) Dilated Residual Network: There are two commonly used pre-trained backbone networks for saliency prediction: VGG-16 and ResNet-50. In addition, both of these two backbone networks have their corresponding dilated versions. Thanks to the residual learning introduced by He *et al.* in [33], the ResNet can be trained very deeply for more comprehensive feature extraction. Existing works also support that (dilated/plain) ResNet-50 based saliency models perform better than those based on (dilated/plain) VGG-16. In this work, we employ the commonly used ResNet-50 as our backbone network.

ResNet-50 backbone network has five blocks of convolutional layers. The *output_stride* of the plain ResNet-50 network is 32 which will lead to some ambiguities in dense predictions. In dilated ResNet-50 [18], [19], to obtain relatively larger spatial resolution without too much computation cost increase, the original three convolutional blocks are kept fixed while the Conv4 and Conv5 blocks are modified by removing down-sampling operations and replacing the standard convolutions inside these blocks by dilated convolutions with dilation rate of 2 and 4, respectively. As a result, the *output_stride* of dilated ResNet-50

is 8 which results in a good compromise between the spatial resolution and computation cost.

B. Decoder Network

In our framework, the DRN acts as a basic encoder network. Note that a decoder network is needed to generate the saliency map from the encoded features in DRN. One conventional decoder network is built by stacking deconvolutional layers which can also help in up-sampling the coarse feature maps into dense ones. However, up-sampling these non-dense feature maps by deconvolutions inevitably need extra heavy computations and also bring some non-smoothing patterns inside them [44]. Thanks to the DRN backbone network, the encoded feature maps have relatively denser spatial information. Therefore, the deconvolutional layers are no longer used in our decoder network.

Instead, our decoder network is very simple since it only consists of three stacked standard convolutional layers with one bilinear up-sampling operation in the end. This number of convolutional layers is determined by our experiments in Section IV-F2. The first two layers have 256 3×3 convolutional kernels with the ReLU activation. The last convolutional layer is the prediction layer. It has only one 3×3 convolutional kernel with the sigmoid activation to generate the down-sampled version of the saliency map. The reason for using sigmoid activation function in this layer is related to the range of saliency value where each pixel belongs to [0, 1]. The outputs can be rescaled into this target interval by this function. After these three convolutions, the resolution of the outputs is still lower than the inputs since no up-sampling operations are involved. We simply apply a bilinear up-sampling operation in the end to reduce the computation cost. Compared to the existing efforts, our decoder network is simple yet effective. The baseline model for this paper is the combination of DRN and this decoder network. In fact, we insert a $2048 \times 1 \times 1 \times 256$ convolutional layer between the DRN and decoder network to reduce the number of parameters in this baseline model. To our surprise, the performance of our baseline model has no visible change with such modification. When constructing the DINet, we replace this newly inserted layer by our dilated inception module, as presented in the next section.

C. Proposed Dilated Inception Module

The proposed module is derived from the inception module which intends to capture the multi-scale contextual information from the inputs [25]. The principal idea of the original inception module is to utilize multiple convolutional layers with different kernel sizes working as multi-scale feature extractors with various receptive field sizes, as shown in Fig. 4(a). Unlike the well-known GoogLeNet [25] which is stacked by several customized inception modules with carefully designed topologies, inception module acts as a single plug-in module in our model to diversify the receptive fields of those encoded features from the output of DRN.

The filter numbers in our inception modules are all fixed to 256. By inserting the inception module between the DRN and decoder network, the performance of our new model is improved
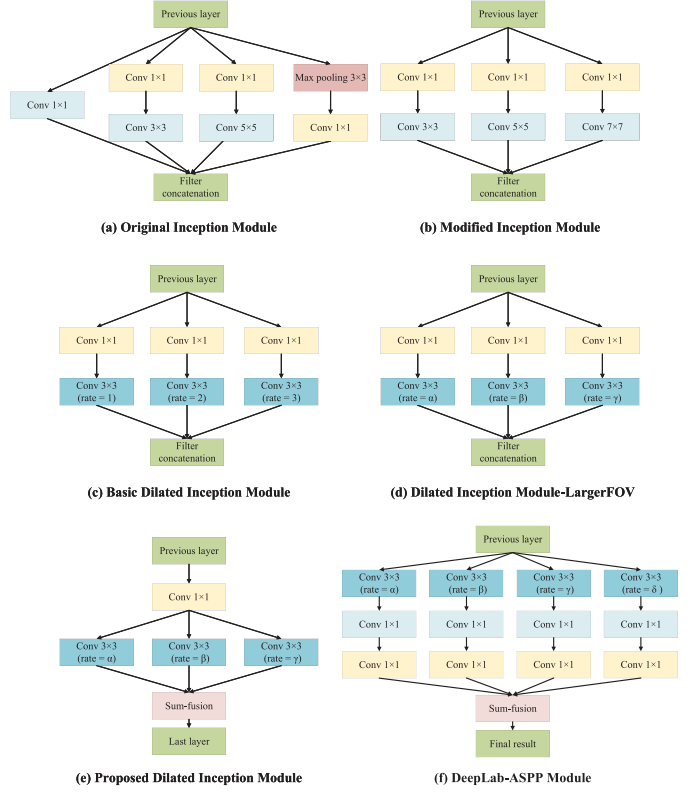

Fig. 4. The inception module with its variations and the ASPP module. Module (a) is the original inception module [25]. Module (b), (c), and (d) are three variants. Module (e) is our final proposed dilated inception module (DIM). Module (f) is the DeepLab-ASPP module [26]. The yellow 1×1 convolutional blocks have the ability of dimensionality reduction.

obviously with acceptable extra parameters and computations. However, we find that the branch of 1×1 convolutional block has limited influence on final results. Besides, we replace the max-pooling branch by one 7×7 convolutional layer after one 1×1 convolutional layer to only investigate and explore the convolutional layers within inception module, as shown in Fig. 4(b). With the help of 7×7 convolutional block, the modified inception module can extract more diverse and wider field-of-view (FOV) features. For simplification, we denote the parameters number of a $256 \times 1 \times 1 \times 256$ convolutional layer (without bias term) as W. Therefore, 7×7 convolutional layer in inception module (b) has $7^2 W = 49 W$ parameters to be determined, which is much more than 5×5 convolution ($25 W$ parameters) and 3×3 convolution ($9 W$ parameters). The total number of parameters in the modified inception model needs an additional $32 W$ parameters compared to the original inception model, which result in larger computation cost and longer inference time.

Recall the dilated convolutions introduced in Section III-A, dilated convolutions can be used to replace the large kernel standard convolutions under the same receptive field, as shown in Fig. 4(c). 7×7 and 5×5 convolutions in the modified inception module can be replaced by 3×3 dilated convolutions with dilation rate of 3 and 2, respectively. After this replacement, the dilated inception module (DIM) can perform the similar

TABLE II
THE COMPARISON OF THE BASELINE MODEL AND OTHER MODELS WITH DIFFERENT MULTI-SCALE CONTEXT FEATURE EXTRACTION MODULES.
THE MODEL (BASELINE + INCEPTION(E)) IN BOLD IS OUR FINAL PROPOSED DINET MODEL

Model	Total #params	Extra params (%)	Best validation loss	Average Inference time
Baseline (DRN + Decoder)	25.27M	0	0.2776	72.40s
ResNet + Skip-layer + Decoder	26.84M	6.21	0.2793	48.52s
Baseline + Skip-layer			0.2739	76.38s
Baseline + IPN	26.38M	4.39	0.2732	100.54s
Baseline + Inception(a)	30.84M	22.04	0.2720	89.50s
Baseline + Inception(a) - 1×1 branch	29.72M	17.61	0.2721	87.91s
Baseline + Inception(b)	32.94M	30.35	0.2701	90.22s
Baseline + Inception(c)	29.27M	15.83	0.2696	81.08s
Baseline + Inception(d)			0.2673	81.22s
Baseline + Inception(e)	27.04M	7.00	0.2679	77.39s
DRN + ASPP-S	42.70M	67.98	0.2679	116.92s
DRN + ASPP-L			0.2684	141.38s

or even better results as the modified inception module with $(7^2 + 5^2 - 2 \times 3^2)W = 56\,W$ parameters less. It is worth noting that dilated convolutions in DRN are used in a cascaded way to preserve the spatial resolution and compensate the reduction in receptive fields. While in DIM, dilated convolutions are used in a parallel way to enhance the encoded features with diverse and comprehensive field-of-views.

Furthermore, the dilation rate of these three parallel dilated convolutions can be arbitrarily changed, as denoted by $[\alpha, \beta, \gamma]$. Considering that the last convolutional block of the DRN has set the dilation rate equal to 4, our DIM can be viewed as an extended convolutional block of the DRN with a combination of three parallel dilated convolutions inside. In our experiments, we set $[\alpha, \beta, \gamma] = [4, 8, 16]$ which show a great improvement from the primary dilated or original inception module. The receptive fields of the outputs after our dilated inception module are diverse and relatively large which contribute to incorporate various contextual information at different scales. This DIM with larger FOV is depicted in Fig. 4(d). We further reduce the computational complexity of our model by building a bottleneck type of DIM, as shown in Fig. 4(e). On the one hand, we use one single 1×1 convolutional layer in the top to replace the existing individual ones in the different branches for dimensionality reduction. On the other hand, the filter concatenation is replaced by sum-fusion (element-wise addition) which can also help in dimensionality reduction and efficient computation. As a result, this final DIM only brings an additional $27\,W$ parameters which indicate only three extra 3×3 convolutional layers are added, compared to the baseline model. Furthermore, with the help of this computationally-friendly module, our proposed DINet can reach more than 50 FPS inference time for input images of size 240×320.

In the literature, the atrous spatial pyramid pooling (ASPP) module [26] also utilize parallel dilated convolutions for learning multi-scale feature representations, as shown in Fig. 4(f). In this module, the features extracted at different dilation rates are further processed in separate branches and sum-fused to generate the final results. In contrast, our DIM is just a single plug-in module and its outputs are still features, rather than the final results. Since these two modules share the same idea of using the parallel dilated convolutions, it is also reasonable to use ASPP module to replace our DIM and its followed decoder network for saliency prediction. Directly insert this ASPP module on

the top of DRN cannot guarantee that every pixel in the final results is in the range of [0,1]. We add an extra linear scaling operation after sum-fusion to solve this. ASPP module has two variants: ASPP-S and ASPP-L. The only difference in these two is the setting of dilation rates. ASPP-S has smaller dilation rates ($[\alpha, \beta, \gamma, \theta] = [2, 4, 8, 12]$) while ASPP-L has larger rates ($[6,12,18,24]$). The information of these two ASPP-based saliency models is reported in the last two rows in Table II. As observed from this table, with the help of huge extra parameters, model (DRN + ASPP-S) can obtain a similar performance to our DINet. Compared to the ASPP module, our DIM only need one decoder network to generate the saliency predictions since we have the sum-fusion before the decoder rather than after it. Another reason for longer inference time in the ASPP-based model is that our DIM performs the 1×1 convolution before the dilated convolutions for dimension reduction while ASPP directly uses dilated convolutions to process these features from DRN. Specifically, the difference between the dilated convolutions part of ASPP and our DIM in #parameters is $8 \times 3^2 \times 4 = 288\,W$ versus $8 + 3^2 \times 3 = 35\,W$.

Besides, we also investigate other existing multi-scale context feature extraction frameworks, such as image pyramid network (IPN) with shared backbone network and skip-layer network, into our baseline model. The overall comparison among these models is listed in Table II. Extra params (%) term indicates the percentage of the number of additional parameters involved when using this model compared to the baseline model. The best validation loss term means that the smallest loss results of the models on SALICON validation dataset [20]. The loss function used in here is the linear normalization-based total variation distance, as discussed in the next section. The detailed evaluation results corresponding to these loss values are reported in Table V. Average inference time term is the average time of these models for predicting 5,000 validation images with 5 repeats under the same experimental conditions. Among these models in Table II, our DINet achieves a relatively good trade-off between the validation performance and inference speed.

D. Loss Function

Most saliency models directly predict saliency maps via optimizing loss functions designed for pixel-wise regression/ classification. However, saliency map can be viewed as a

probability distribution (PD) of human fixations over the whole image [27]. Pixel-wise prediction, where each pixel is predicted individually, may suffer from the global inconsistency problem as it ignores the inter-pixel relationship. Therefore, it is reasonable to use off-the-shelf PD distance metrics as loss functions. In order to convert the predicted saliency map and its corresponding ground-truth into probability distributions, a normalization method should be applied first. Here, we improve the existing method [27] by replacing their softmax normalization with a simple linear regularization.

Base on the validation experimental results, we select the total variation distance as the loss function. Besides, the unnormalized version of total variation distance is the $\ell1$-norm which is a commonly used regression loss. Due to these two factors, we use this loss function as an example to illustrate the differences between our proposed linear normalization-based loss function and the existing two types. The total variation distance or $\ell1$-norm can be broadly formulated by the following equation:

$$L(\mathbf{p}, \mathbf{g}) = \sum_i |p_i - g_i|, \qquad (2)$$

where $\mathbf{p}$ is the predicted result and $\mathbf{g}$ is the ground-truth. The definitions of these two terms are different in each loss function, as listed in the following:

In $\ell1$-norm (unnormalized loss function),

$$p_i = x_i^p, \quad g_i = x_i^g. \qquad (3)$$

In softmax normalization-based loss function,

$$p_i = \frac{exp(x_i^p)}{\sum_{i=1}^N exp(x_i^p)}, \quad g_i = \frac{exp(x_i^g)}{\sum_{i=1}^N exp(x_i^g)}. \qquad (4)$$

In linear normalization-based loss function,

$$p_i = \frac{x_i^p}{\sum_{i=1}^N x_i^p}, \quad g_i = \frac{x_i^g}{\sum_{i=1}^N x_i^g}, \qquad (5)$$

where $\boldsymbol{x} = (x_1, \ldots, x_i, \ldots, x_N)$ is the set of unnormalized saliency response values for either the predicted saliency map (x^p) and the ground-truth saliency map (x^g).

The experiments in Section IV-D illustrate that proposed linear normalization-based loss functions perform better than both softmax normalization-based and unnormalized ones. The target output in saliency prediction is an array $\boldsymbol{x^g} \in [0, 1]^N$. According to the following theorem, for an array whose values between 0 and 1, the softmax will de-emphasize the maximum values among them [45] while the linear normalization still maintains their initial proportion. Therefore, the existing loss functions coupled with softmax normalization cannot measure the gaps between the predicted probability distribution and its corresponding ground-truth very well.

Theorem 1: Given an array $\boldsymbol{x} \in [0, 1]^N$, using Equation (4) and Equation (5) to normalize this array separately, denote the range of the elements of this two normalized arrays as $[a_s, b_s]$ and $[a_l, b_l]$, respectively. Then, we have:

$$[a_s, b_s] \subset [a_l, b_l].$$

Proof: It is obvious that both these normalization functions are monotonic increasing functions. We also note that

$\boldsymbol{x} \in [0, 1]^N$. So, we get the minimum normalized response when $x_i = 0$ and get the maximum when $x_i = 1$. Considering that we have $a_s = \frac{exp(0)}{\sum_i exp(x_i)} = \frac{1}{\sum_i e^{x_i}} > 0 = \frac{0}{\sum_i x_i} = a_l$. Now we only need to prove $b_l \geq b_s$. In fact, we have:

$$b_l - b_s = \frac{1}{\sum_i x_i} - \frac{e}{\sum_i e^{x_i}} = \frac{\sum_i(e^{x_i} - ex_i)}{\sum_i x_i \sum_i e^{x_i}}.$$

Recall that $x_i \in [0, 1]$, it is easy to prove that $e^{x_i} - ex_i \geq 0$ for every $x_i \in [0, 1]$. So we have $b_l \geq b_s$. ∎

IV. Experiments

In this section, we apply our proposed DINet for saliency prediction and report its experimental results on several public saliency benchmark datasets. The effectiveness and efficiency of our model is validated qualitatively and quantitatively.

A. Saliency Benchmark Datasets

For evaluating the saliency prediction model, we adopt three popular saliency benchmark datasets with different image contents and experimental settings.

1) SALICON [20]: It contains 10,000 training images, 5,000 validation images, and 5,000 testing images, taken from the Microsoft COCO dataset [46]. The spatial resolution of each image in this dataset is 480×640. At present, it is the largest public dataset for visual saliency prediction. The ground-truths of training and validation datasets are available while the ground-truths of test dataset are held out. For evaluation on its test dataset, researchers need to submit their results on the SALICON challenge website.[2] Besides, the evaluation protocols and codes are available in the website.[3]

2) MIT1003 [21]: It contains 1,003 images collected from Flickr and LabelMe. The ground-truths for this dataset are created from eye-tracking data of 15 users. The evaluation codes for this dataset are available in the MIT Saliency Benchmark website.[4]

3) MIT300 [22]: It contains 300 images, including both indoor and outdoor scenarios. The ground-truths for this entire dataset are held out. Researchers can only submit the results of their models to the MIT Saliency Benchmark website[4] for evaluation. Currently, the MIT1003 dataset is usually used as the training and validation sets for this dataset.

B. Evaluation Metrics for Saliency Prediction

There exists a large variety of metrics to measure the agreement between model predictions and human eye fixations. Following existing works [47], [48], we conduct our quantitative experiments by adopting four widely used saliency evaluation metrics, including AUC, shuffled AUC (sAUC), Normalized Scanpath Saliency (NSS), and Linear Correlation Coefficient (CC). For the sake of simplification, we denote the predicted saliency map as P, the ground-truth saliency map as G, and the

Metrics	Category	Ground-truth
AUC (area under the ROC curve)	Location-based	Fixation Map (Q)
sAUC (shuffled AUC)	Location-based	Fixation Map (Q)
Normalized Scanpath Saliency (NSS)	Value-based	Fixation Map (Q)
Linear Correlation Coefficient (CC)	Distribution-based	Saliency Map (G)

ground-truth fixation map as Q. The saliency evaluation metrics are listed in Table III according to their characteristics.

1) AUC and sAUC: AUC means the Area Under the ROC curve. This metric evaluates the binary classification performance of the predicted saliency map P, where fixation and non-fixation points in its corresponding Q are divided into the positive set and negative set, respectively. By using a threshold, P can be binary classified into the salient and non-salient regions. ROC curve will be obtained by varying this threshold from 0 to 1. Finally, the AUC metric can be calculated by using this ROC curve. Shuffled AUC (sAUC) is introduced to alleviate the influence of center-bias. Differ in AUC, the fixation points of other images in this dataset is used as the negative set in computing sAUC values. However, these two AUC-based metrics have the limitation in penalizing false positives, as reported in [17]–[19].

2) NSS: Normalized Scanpath Saliency (NSS) is a specific value-based saliency evaluation metric. This metric is computed by taking the mean of $\bar{P}$ at the human eye fixations Q:

$$NSS = \frac{1}{N} \sum_{i=1}^{N} \bar{P}(i) \times Q(i), \tag{6}$$

where N is the total number of human eye fixations, $\bar{P}$ is the unit normalized saliency map P.

3) CC: The Linear Correlation Coefficient (CC) is a statistical metric for measuring the linear correlation between two random variables. For saliency prediction evaluation, the predicted saliency maps (P) and ground-truth density maps (G) are treated as two random variables. Then, CC is calculated by the following equation:

$$CC = \frac{cov(P, G)}{\sigma(P) \times \sigma(G)}, \tag{7}$$

where $cov(\cdot, \cdot)$ and $\sigma(\cdot)$ refer to the covariance and standard deviation, respectively.

C. Implementation Details

Our proposed DINet is implemented by Keras with Tensor-Flow backend [49], [50]. During training, the weights in Dilated ResNet-50 Network (DRN) are initialized from the ImageNet-pretrained ResNet-50 Network. The weights of remaining layers are initialized by the default setting of Keras. The whole model is trained with widely used Adam optimizer [51] with an initial learning rate of 10^{-4}. This learning rate will be scaled down by a factor of 0.1 after every two epochs. A mini-batch of 10 images is used in each iteration.

We train our model on the training set of SALICON [20] with 10,000 training images and use its validation datasets (5,000 validation images) to validate the model. For the MIT1003

Loss function	CC	sAUC	AUC	NSS
Total Variation distance (linear)	**0.843**	0.788	0.885	3.077
Total Variation distance (softmax)	0.826	0.786	**0.888**	2.906
ℓ1-norm	0.810	0.783	0.874	2.960
Bhattacharyya distance (linear)	0.839	0.786	0.881	3.077
Bhattacharyya distance (softmax)	0.828	0.785	0.884	2.992
KLD (linear)	0.842	0.788	0.886	3.070
KLD (softmax)	0.827	0.785	0.884	2.968
χ^2 divergence (linear)	0.826	**0.790**	0.886	2.994
χ^2 divergence (softmax)	0.826	0.786	0.883	2.968
Cosine distance (linear)	0.835	0.789	0.885	3.048
Cosine distance (softmax)	0.828	0.786	0.887	2.999
BCE	0.826	0.785	0.885	2.963
Euclidean (ℓ2-norm)	0.824	0.781	0.884	2.958
NSS	0.733	0.782	0.860	**3.411**

dataset [21], we directly use the model trained on the SALICON dataset to evaluate the generalization performance of our model on this dataset. For testing on the MIT300 dataset [22], we fine-tune our model in the MIT1003 dataset with the same evaluation protocol in [18], [19]. The fine-tuned results of the MIT1003 dataset are also presented. For the latter two datasets, the input images are all resized to 320×480 with zero padding to keep the original content aspect ratio. This input image size is decided by our validation experiments on SALICON dataset. It is worth mentioning that our model can achieve processing speed as little as 0.02 s and 0.03 s for one input image of size 240×320 and 320×480, respectively, by using one single GTX 1080 Ti GPU.

D. Loss Function Analysis

We compare the performance of our baseline models trained by our proposed probability distribution (PD) distance metrics with linear normalization to those trained on standard regression loss functions and existing softmax normalization based statistical distances.

Table IV presents the experimental results for each loss function, as measured by the overall performance with respect to four aforementioned evaluation metrics on SALICON validation dataset. These results support that: (i) generally, the loss functions based on PD distance metrics perform better than standard regression loss functions, such as BCE, ℓ1-norm, and ℓ2-norm in our experiments; (ii) for a specific statistical distance based loss function, our proposed linear normalization method is more compatible than the softmax normalization as it can measure the distance between the predicted PD and its target in a more proper way; (iii) Using NSS loss function alone can obtain an extremely high NSS score while this loss function is not very good at other three evaluation metrics.

The first two conclusions have been discussed in section III-D. The reason for (iii) can be illustrated by Table III. NSS is a value-based saliency evaluation metric since it is computed by the average of the normalized saliency values at eye fixation locations. In other words, a saliency map with a higher NSS score is more like a fixation map which is not similar to the fixation

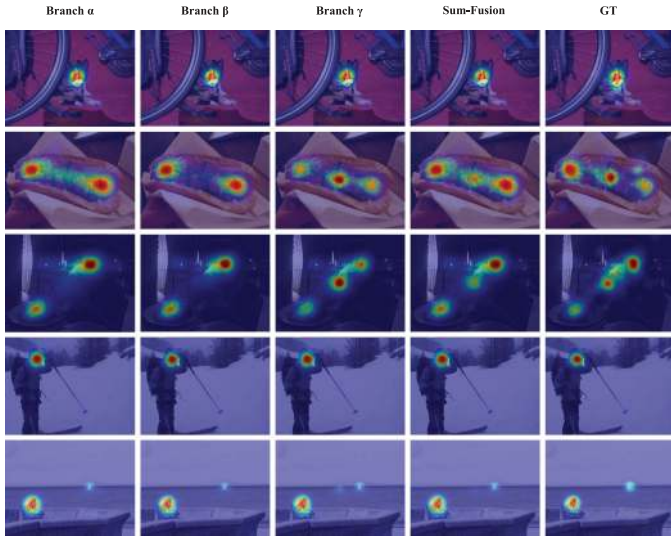| Branch α | Branch β | Branch γ | Sum-Fusion | GT |

Fig. 5. The influence of each dilated convolutional branch in the DIM to visual saliency. In each col, images are the saliency prediction results by using the features captured from the above indicated branch. GT: Ground Truth.

Model	CC	sAUC	AUC	NSS
Influence of backbone network				
ResNet + Decoder	0.776	0.762	0.879	2.456
Baseline (DRN + Decoder)	0.843	0.788	0.885	3.077
Influence of decoder network				
DRN + Decoder(1 conv layer)	0.838	0.785	0.883	3.052
DRN + Decoder(2 conv layers)	0.841	0.787	0.884	3.067
DRN + Decoder(4 conv layers)	0.843	0.788	0.885	3.072
DRN + Decoder(1 deconv + 1 conv layers)	0.841	0.787	0.884	3.064
DRN + Decoder(2 deconv + 1 conv layers)	0.841	0.788	0.884	3.067
DRN + Decoder(3 deconv + 1 conv layers)	0.841	0.787	0.885	3.061
Effectiveness of multi-scale features				
ResNet + Skip-layer + Decoder	0.841	0.786	0.885	3.053
Baseline + IPN	0.849	0.787	0.885	3.086
Baseline + Skip-layer	0.847	0.788	0.886	3.084
Baseline + Inception(a)	0.850	0.788	0.886	3.094
Baseline + Inception(a) - 1×1 branch	0.849	0.789	0.886	3.091
Baseline + Inception(b)	0.852	0.790	0.886	3.107
Baseline + Inception(c)	0.852	0.790	0.886	3.111
Baseline + Inception(d)	0.854	0.790	0.887	3.114
DINet (Baseline + Inception(e))	0.853	0.789	0.887	3.117
DRN + ASPP-S	0.853	0.789	0.887	3.112
DRN + ASPP-L	0.852	0.789	0.887	3.102
Influence of training image size				
DINet (240×320)	0.853	0.789	0.887	3.117
DINet (320×480)	0.858	0.790	0.887	3.143
DINet (480×640)	0.854	0.789	0.886	3.128
DINet (ensemble)	**0.867**	**0.792**	**0.889**	**3.168**

Type	b_α	b_β	b_γ	CC	sAUC	AUC	NSS
The results on additional decoder network							
0 branch				0.752	0.794	0.858	2.729
1 branch	✓			0.833	0.799	0.882	3.012
		✓		0.811	0.793	0.864	3.025
			✓	0.801	0.759	0.873	2.967
2 branches-sum	✓	✓		0.831	0.799	0.879	3.035
	✓		✓	0.804	0.799	0.869	3.036
		✓	✓	0.813	**0.800**	0.872	3.032
3 branches-sum	✓	✓	✓	**0.853**	0.789	0.886	3.098
The results on original decoder network							
3 branches-sum	✓	✓	✓	**0.853**	0.789	**0.887**	**3.107**

density map, i.e. saliency map. Conversely, another three evaluation metrics (CC, AUC, sAUC) prefer the latter one. Therefore, it is difficult to use one single loss function to train the DCNN model for obtaining a promising result on both NSS and other evaluation metrics.

E. Model Visualization

We verify the effectiveness of DIM by individually visualizing the responses of each dilated convolutional branch. This visualization experiment is realized by adding an additional decoder network without non-linear activation at the end of our DIM. Both of this additional decoder and the original decoder are jointly trained with the same loss and the same inputs from the DIM. Since the additional decoder is a linear operator applied to input feature maps, the joint decoded output in this decoder can be decoupled into a linear combination of the outputs coming from individual branches. Moreover, the input dimension of our decoder is the same as the output dimension of every branch in our DIM (all are equal to 256). The responses of each branch can be easily obtained by feeding this additional decoder with the features learned in this specific branch. By visualizing both joint and individual saliency prediction results, we can analyze the contribution of these dilated convolutional branches in our DIM.

Fig. 5 demonstrates the saliency prediction results of five validation images. The first three columns show the saliency maps independently predicted by branch $-\alpha$, $-\beta$ and $-\gamma$, and the fourth column shows the final saliency maps by sum-fusing the outputs produced by mentioned branches. All of these predicted saliency maps are generally consistent with the ground-truth. As demonstrated in the second and the third rows, branches with different receptive fields learn to focus on different parts of an input image. Specifically, the branch γ, i.e. b_γ, with the largest dilation rate, learns the center-bias implicitly without any additional

supervision. These learned center-bias patterns compensate the negligence on the center salient regions from other two branches, b_α and b_β, and produce a more accurate saliency prediction result. On the other hand, b_γ sometimes generates false alarms in the center regions with low confidence. In this case, as shown in the last two rows of Fig. 5, the previous two branches b_α and b_β can help in reducing this unwanted side-effect on the final fusion results. These three branches in our DIM work in a collaborative manner. The results by using the features from a single branch are no need to be perfect for all possible cases. These incomplete predictions will be ensembled by the sum-fusion to become more comprehensive and reliable final results, which can be also supported by our ablation analysis in Table VI.

F. Model Ablation Analysis

In this section, we conduct ablation analysis for our DINet on the SALICON validation dataset. The complete ablation results are presented in Table V. It should be noted that all of models in

this table are trained by the proposed linear normalization-based total variation distance loss function.

1) Influence of the Backbone Network: Our baseline model is built on DRN where the *output_stride* is equal to 8. As mentioned in Section II-A, the *output_stride* of original ResNet is 32 which means that less spatial information are included in the output of this backbone network and thus leads to the unsatisfactory performance. To verify this statement, we compare our baseline model (DRN + decoder) with a more basic model (ResNet + decoder). From the first part of Table V, we can conclude that *output_stride* is one of the key elements for the dense prediction tasks. There is a significant performance gain by replacing the original ResNet with DRN.

2) Influence of the Decoder Network: In our baseline model, our designed decoder network is just three convolutional layers plus sigmoid activation in the end. The reason for using three layers is determined by the experiments. We have tried to use different number of convolutional or deconvolutional layers before the prediction layer (one convolutional layer followed by a sigmoid activation) to form other decoder networks. Their results are reported in the second part of Table V. As we can see that the models with these decoders cannot get good results as our original decoder, i.e. Decoder(3 conv layers).

3) Effectiveness of Multi-Scale Features: DINet uses the proposed DIM to capture multi-scale contextual features. To support the conclusions in [17], [18], [31] that integrating multi-scale features can further improve saliency detection performance, we incorporate existing alternative multi-scale feature extraction modules, including IPN, skip-layer, inception and ASPP, into our baseline or backbone network. From the third part of Table V, we can observe that the saliency prediction performance indeed boosted by incorporating the multi-scale features. Especially, when the backbone network is not DRN, the multi-scale features can compensate the performance drop significantly, by comparing two models with the plain ResNet backbone network. In all these multi-scale saliency prediction framework, our proposed inception(d) and (e) obtain the optimal results among them. For the reason that inception(e) is more efficient in terms of #parameters and inference time, as illustrated by Table II, we pick this DIM to form our DINet.

4) Ablation Analysis on DIM: We further verify the effectiveness of our DIM by conducting two quantitative experiments. In the first experiment, we evaluate the performance of a trained DINet with two decoders mentioned in the visualization experiment to investigate the contribution of each dilated convolutional branch in our DIM respectively. In the second experiment, we make a comparison among a set of variants of DINet to explore the impact of the number of parallel dilated convolutional layers.

Table VI shows the results of the first experiment. Each row in this table represents the evaluation results by using the outputs from the indicated branch(es) as the input to a trained decoder. As we can see that, 1 branch type of DIM will learn different bias under its specific receptive fields to help in predicting visual saliency. Specifically, b_α prefers the results with higher sAUC score, while b_β is more interested in the NSS metric. By comparing the results between the row of 3 branches-sum and the rows in 2 branches-sum type on the first part of this table, we can

TABLE VII
DILATED INCEPTION MODULE ABLATION ANALYSIS WITH INDIVIDUAL
TRAINED VARIANTS OF DINET ON SALICON VALIDATION DATASET [20]

Type	b_α	b_β	b_γ	CC	sAUC	AUC	NSS
0 branch				0.843	0.788	0.885	3.077
1 branch	✓			0.847	**0.790**	0.886	3.080
		✓		0.849	0.788	**0.887**	3.086
			✓	0.851	0.788	**0.887**	3.095
2 branches-sum	✓	✓		0.852	0.788	**0.887**	3.099
	✓		✓	0.853	0.788	0.886	3.098
		✓	✓	0.852	0.788	**0.887**	3.103
3 branches-sum	✓	✓	✓	0.853	0.789	**0.887**	**3.117**
3 branches-concat	✓	✓	✓	**0.854**	0.789	**0.887**	3.116

observe that the performance drop dramatically with the absence of any one branch, which means every branch in our DIM has its irreplaceable impact on the final results. These three branches in our DIM work in a collaborative manner. Even if the performance by using any individual branch is not comparable to the performance of our baseline model, their fused results can deal with the diverse images with different patterns of salient regions. Moreover, the results on the last row show that the features used in the additional decoder can still be decoded by our original decoder with only a little bit performance drop in the NSS metric. It can guarantee the generality of the above conclusions.

Table VII compares the performance of several variants of DINet. Each row in this table means the evaluation results by testing the individual trained variant which has the indicated branch(es). Especially, the model in 3 branches-sum type is the proposed DINet, while the model in 0 branch type is our baseline model. This table shows that using more branches (from 0 to 3), which means using more comprehensive features, will lead to a higher performance on evaluation metrics. Besides, in the 1 branch type of DINet, using dilated convolution with larger dilation rate before the decoder network can achieve a better performance than using a smaller one. It can be credited to the larger size of receptive fields which represent the longer range of dependencies in captured features. Moreover, using concatenation to replace our element-wise addition has a limited impact on the final results, as presented in the last two rows in this table. Mathematically, element-wise addition followed by a convolution layer is a special case of concatenation followed by another convolution layer [52], which can be used to explain this limited difference on evaluation results. In summary, both of these two experiments can verify that the performance gain of our DIM is realized by the corporation of these three parallel dilated convolutional branches.

5) Influence of Training Image Size: The previous experimental results on SALICON validation dataset are all obtained from 240×320 images, whose size is the half resolution of the original SALICON images. Here we want to see the performance of our DINet models which are trained by images with different spatial resolution. From Table V, we find that the DINet trained by input images of size 320×480 can obtain the best performance among these three models. This model will be directly fine-tuned in the MIT1003 dataset for the evaluation of the MIT300 dataset. Note that these evaluation results are the average scores, there are some validation images which perform better in other DINets (240×320 or 480×640). In order to

TABLE VIII
PERFORMANCE COMPARISON OF OUR DINET MODELS WITH DIFFERENT LOSS
FUNCTIONS ON SALICON VALIDATION DATASET [20]

Model	CC	sAUC	AUC	NSS
DINet (TV distance)	**0.867**	**0.792**	**0.889**	3.168
DSCLSTM [18]	0.835	0.788	0.887	3.221
SAM-ResNet [19]	0.844	0.787	0.886	3.260
DINet (NSS)	0.724	0.782	0.861	**3.600**
DINet (ensemble NSS and TV distance))	0.862	**0.792**	0.886	3.310

TABLE IX
COMPARISON RESULTS ON THE SALICON TEST DATASET [20]

Models	CC	sAUC	AUC	NSS
DINet (Ours)	**0.860**	0.782	**0.884**	**3.249**
SAM-ResNet [19]	0.842	0.779	0.883	3.204
DSCLRCN [18]	0.831	0.776	**0.884**	3.157
SAM-VGG [19]	0.825	0.774	0.881	3.143
SalGAN [39]	0.781	0.772	0.781	2.459
SU [53]	0.780	0.760	0.880	2.610
PDP [27]	0.765	0.781	0.882	-
ML-Net [35]	0.743	0.768	0.866	2.789
MxSalNet [23]	0.730	0.771	0.861	2.767
Deep Convnet [54]	0.622	0.724	0.858	1.859
Shallow Convnet [54]	0.562	0.658	0.821	1.663
DeepGazeII [38]	0.479	**0.787**	0.867	1.271

TABLE X
COMPARISON RESULTS ON THE MIT1003 DATASET [21]

Model	CC	sAUC	AUC	NSS
DINet (w/o finetune)	**0.67**	0.70	**0.88**	**2.40**
DVA [24]	0.64	**0.77**	0.87	2.38
GBVS [15]	0.42	0.66	0.83	1.38
eDN [28]	0.41	0.66	0.85	1.29
Mr-CNN [55]	0.38	0.73	0.80	1.36
BMS [16]	0.36	0.69	0.79	1.25
ITTI [14]	0.33	0.66	0.77	1.10

TABLE XI
COMPARISON RESULTS ON THE MIT1003 VALIDATION DATASET [21]

Model	CC	sAUC	AUC	NSS
DINet (w finetune)	**0.87**	**0.77**	**0.91**	**3.27**
SAM-ResNet [19]	0.77	0.62	**0.91**	2.89
SAM-VGG [19]	0.76	0.61	**0.91**	2.85
DeepFix [17]	0.72	0.74	0.90	2.58
DINet (w/o finetune)	0.67	0.72	0.89	2.50

TABLE XII
COMPARISON RESULTS ON THE MIT300 DATASET [22]

Model	CC	sAUC	AUC	NSS
DSCLRCN [18]	**0.80**	0.72	0.87	**2.35**
DINet (Ours)	0.79	0.71	0.86	2.33
SAM-ResNet [19]	0.78	0.70	0.87	2.34
DeepFix [17]	0.78	0.71	0.87	2.26
SAM-VGG [19]	0.77	0.71	0.87	2.30
SALICON [31]	0.74	**0.74**	0.87	2.12
SalGAN [39]	0.73	0.72	0.86	2.04
PDP [27]	0.70	0.73	0.85	2.05
DVA [24]	0.68	0.71	0.85	1.98
ML-Net [35]	0.67	0.70	0.85	2.05
SalNet [54]	0.58	0.69	0.83	1.51
BMS [16]	0.55	0.65	0.83	1.41
DeepGazeII [38]	0.52	0.72	**0.88**	1.29
GBVS [15]	0.48	0.63	0.81	1.24
Mr-CNN [55]	0.48	0.69	0.79	1.37
eDN [28]	0.45	0.62	0.82	1.14
ITTI [14]	0.37	0.63	0.75	0.97

characterize this phenomenon, we adopt a simple ensemble learning metric, i.e. average voting, to further improve the performance of our model. By using the average results from these three different models, this ensemble model obtain the best scores in our model ablation analysis.

6) Ensemble Learning for Improving NSS: However, our best model, which is trained by a single total variation distance loss function, still cannot beat two state-of-the-art models [18], [19] in NSS metrics, as shown in Table VIII. These two models use the NSS itself as one of the loss functions for training. To further improve our performance on NSS metrics, we use the same ensemble learning method as above to combine the results of two DINet models which are trained by using two different loss function (total variation distance with linear normalization and NSS) separately. The last ensemble model in this table is our final submission to the SALICON test dataset which results in a good comprise between NSS and another three evaluation metrics.

G. Comparison With State-of-the-Arts

To demonstrate the effectiveness of our proposed DINet model in predicting visual saliency, we quantitatively compare our method with state-of-the-art models on SALICON, MIT1003, and MIT300 datasets.

Table IX shows the evaluation results on the SALICON dataset. The results of other models come from their papers or the leaderboard of this dataset. In this table, the results in bold indicate the best performance method on each evaluation metric. As it can be observed, our DINet outperforms all competitors on CC, AUC, and NSS three metrics. The DeepGazeII [38] model get the best sAUC score and relatively lower scores on other metrics. The saliency maps generated by this model actually are very blurred/hazy and visually different from the ground-truth, as shown in the left part of Fig. 6x. This is because AUC-based metrics mainly relied on true positives without significantly penalizing false positives [17], [19].

The results on MIT1003 are reported in Table X. We directly use the DINet trained on the SALICON dataset to evaluate the generalization performance of our model on the whole MIT1003 dataset, as the DVA model [24]. Our model also achieves promising results on this dataset which verifies its robustness and generality. Qualitative comparison results of our model with other state-of-the-art saliency models on SALICON validation and MIT1003 datasets can be found in Fig. 6. This figure can also support that our results match the ground-truth saliency maps best among all the compared models in both two datasets.

In order to evaluate the MIT300 dataset, we fine-tune our DINet on the MIT1003 dataset. The fine-tuned results are shown in Table XI, As we can see that, the performance of our model improves significantly after fine-tuning which can also outperform other existing fine-tuned models. The results on MIT300 dataset are presented in Table XII. Different in the previous two datasets, our DINet can not outperform the DSCLRCN model [18]. Our model may over-fitted on the MIT1003 dataset which leads to lower generalization performance on MIT300 dataset. Both DSCLRCN model and our DINet use multi-scale
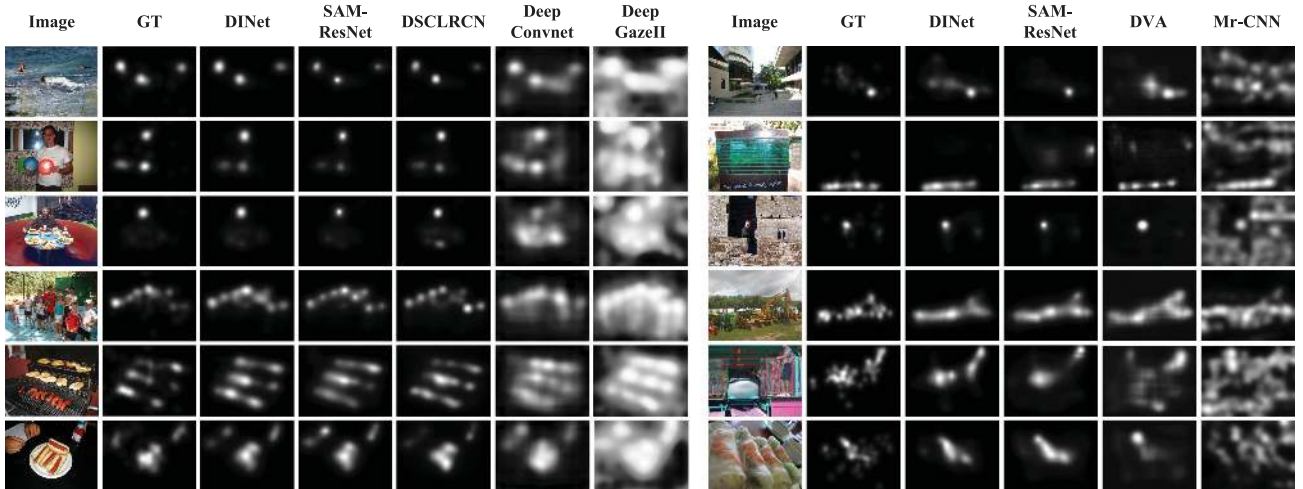
Fig. 6. Qualitative comparison results on two datasets. Left images are from SALICON validation dataset [20], while right images are from MIT1003 dataset [21]. GT: Ground Truth.

TABLE XIII
COMPREHENSIVE COMPARISON WITH THE STATE-OF-THE-ARTS

Model	Backbone network	#parameters	key module	Input image size	inference time (per image)
DINet (Ours)	Dilated ResNet50	27.04M	Dilated inception module for multi-scale	240×320	0.02s
				320×480	0.03s
				480×640	0.06s
SAM^a [19]	Dilated ResNet50	70.09M	Conv-LSTMs [56] for iterative refinement	240×320	0.09s
	Dilated VGG16	51.84M			0.07s
DSCLRCN^b [18]	Dilated ResNet50 + Places-CNN	>33.71M	Spatial LSTMs [57] for context incorporation	$480 \times 640 + 227 \times 227$	0.27s
DVA^c [24]	VGG16	25.07M	Skip-layers for multi-scale	224×224	0.02s

a: The codes for these models are from the authors' github website. We test its inference time in our experimental environment.
b: This model has many customized operations which is hard to count their trainable parameters completely and reimplement in the Keras framework. 0.27 s is adopted from their paper. It is certain that this model need more parameters and longer inference time than our method.
c: This model is reimplemented by ours and tested in our experimental environment. Its deconvolutions-based decoder network slow down the whole model and thus it has similar inference time as our model.

features to further improve saliency prediction performance. Besides, DSCLRCN model incorporates the global context and scene context by using spatial LSTM [57] method and additional Places-CNN [58] backbone network to achieve this performance. Consequently, their model is more complex and much slower than our method. When testing one image with size 480×640, the DSCLRCN model needs 0.27 s while our DINet needs only 0.06 s.

A comprehensive summary of our model and other three state-of-the-art competitors, i.e. SAM [19], DSCLRCN [18], and DVA [24], are listed in Table XIII. Apart from achieving superior performance on the SALICON and MIT1003 datasets, our model also has the obvious advantages in terms of both #parameters and inference time compared to the SAM and DSCLRCN models.

However, despite the good results, there are still a small number of failure cases, as shown in Fig. 7. These bad cases are caused by the fact that so many objects are cumulated in a single image. Within them, the relative importance of these objects cannot be fully learned by simply utilizing the multi-scale contextual features without higher level visual understanding. Therefore, some non-salient regions are highlighted (like the first row) or some salient regions are missed, as shown in the second row. Note that SAM and DSCLRCN models suffer from
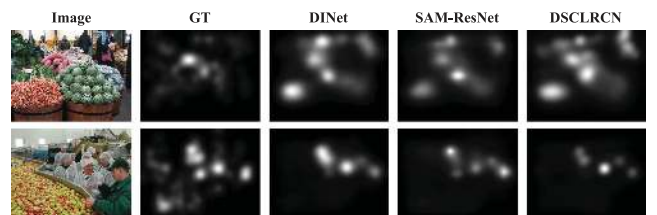

Fig. 7. Some failure cases of our DINet and two state-of-the-arts. Images are from SALICON validation dataset [20].

the same problem as ours. It can be concluded that even the state-of-the-art saliency models still cannot fully understand the relative importance of image regions in such semantically rich scenes. To further approach human-level performance, saliency models will need to discover increasingly higher-level concepts in images for determining an appropriate amount of visual attention on a certain image region.

V. CONCLUSION

We have proposed a dilated inception network for visual saliency prediction. The multi-scale saliency-influential factors are captured by an efficient and effective dilated inception

module. The whole model works in a fully convolutional encoder-decoder architecture, which is trained end-to-end and lightweight for time-efficiency. Furthermore, we adopted a set of linear normalization-based probability distribution distance metrics as loss functions to formulate the saliency prediction problem as a probability distribution prediction task. With such loss functions, our models can perform better than those trained by using either standard regression loss functions or existing softmax normalization-based probability distribution distance metrics. Experimental results on the challenging saliency benchmark datasets have demonstrated the outstanding performance of our model with respect to other relevant saliency prediction methods.

REFERENCES

[1] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 185–207, Jan. 2013.

[2] Y. Gao, M. Shi, D. Tao, and C. Xu, "Database saliency for fast image retrieval," *IEEE Trans. Multimedia*, vol. 17, no. 3, pp. 359–369, Mar. 2015.

[3] S.-S. Lin, I.-C. Yeh, C.-H. Lin, and T.-Y. Lee, "Patch-based image warping for content-aware retargeting," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 359–368, Feb. 2013.

[4] G. Evangelopoulos *et al.*, "Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention," *IEEE Trans. Multimedia*, vol. 15, no. 7, pp. 1553–1568, Nov. 2013.

[5] S. Li, M. Xu, Y. Ren, and Z. Wang, "Closed-form optimization on saliency-guided image compression for HEVC-MSP," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 155–170, Jan. 2018.

[6] H. Hadizadeh and I. V. Bajic, "Saliency-aware video compression," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 19–33, Jan. 2014.

[7] K. Gu *et al.*, "Saliency-guided quality assessment of screen content images," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 1098–1110, Jun. 2016.

[8] Q. Jiang *et al.*, "Optimizing multistage discriminative dictionaries for blind image quality assessment," *IEEE Trans. Multimedia*, vol. 20, no. 8, pp. 2035–2048, Aug. 2018.

[9] H. Kim and S. Lee, "Transition of visual attention assessment in stereoscopic images with evaluation of subjective visual quality and discomfort," *IEEE Trans. Multimedia*, vol. 17, no. 12, pp. 2198–2209, Dec. 2015.

[10] L. Ye *et al.*, "Salient object segmentation via effective integration of saliency and objectness," *IEEE Trans. Multimedia*, vol. 19, no. 8, pp. 1742–1756, Aug. 2017.

[11] A. Borji, M.-M. Cheng, Q. Hou, H. Jiang, and J. Li, "Salient object detection: A survey," *Comput. Vis. Media*, vol. 5, pp. 117–150, 2019.

[12] Z. Liu *et al.*, "Learning Markov clustering networks for scene text detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2018, pp. 6936–6944.

[13] V. Sitzmann *et al.*, "Saliency in VR: How do people explore virtual environments?" *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 4, pp. 1633–1642, Apr. 2018.

[14] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.

[15] J. Harel, C. Koch, and P. Perona, "Graph-based visual saliency," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 545–552.

[16] J. Zhang and S. Sclaroff, "Saliency detection: A Boolean map approach," in *Proc. IEEE Int. Conf. Comput. Vision*, 2013, pp. 153–160.

[17] S. S. Kruthiventi, K. Ayush, and R. V. Babu, "DeepFix: A fully convolutional neural network for predicting human eye fixations," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4446–4456, Sep. 2017.

[18] N. Liu and J. Han, "A deep spatial contextual long-term recurrent convolutional network for saliency detection," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3264–3274, Jul. 2018.

[19] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting human eye fixations via an LSTM-based saliency attentive model," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5142–5154, Oct. 2018.

[20] M. Jiang, S. Huang, J. Duan, and Q. Zhao, "Salicon: Saliency in context," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 1072–1080.

[21] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *Proc. IEEE 12th Int. Conf. Comput. Vision*, 2009, pp. 2106–2113.

[22] T. Judd, F. Durand, and A. Torralba, "A benchmark of computational models of saliency to predict human fixations," Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. MIT-CSAIL-TR-2012-001, 2012.

[23] S. Dodge and L. Karam, "Visual saliency prediction using a mixture of deep neural networks," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 4080–4090, Aug. 2018.

[24] W. Wang and J. Shen, "Deep visual attention prediction," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2368–2378, May 2018.

[25] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 1–9.

[26] 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," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[27] S. Jetley, N. Murray, and E. Vig, "End-to-end saliency mapping via probability distribution prediction," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 5753–5761.

[28] E. Vig, M. Dorr, and D. Cox, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2014, pp. 2798–2805.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[30] M. Kümmerer, L. Theis, and M. Bethge, "DeepGaze I: Boosting saliency prediction with feature maps trained on ImageNet," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015. [Online]. Available: http://arxiv.org/abs/1411.1045

[31] X. Huang, C. Shen, X. Boix, and Q. Zhao, "SALICON: Reducing the semantic gap in saliency prediction by adapting deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 262–270.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 770–778.

[34] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 3431–3440.

[35] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "A deep multi-level network for saliency prediction," in *Proc. 23rd Int. Conf. Pattern Recognit.*, 2016, pp. 3488–3493.

[36] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. 4th Int. Conf. Learn. Represent.*, 2016. [Online]. Available: http://arxiv.org/abs/1511.07122

[37] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 636–644.

[38] M. Kümmerer, T. S. Wallis, L. A. Gatys, and M. Bethge, "Understanding low-and high-level contributions to fixation prediction," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 4789–4798.

[39] J. Pan *et al.*, "SalGAN: Visual saliency prediction with generative adversarial networks," 2017. [Online]. Available: http://arxiv.org/abs/1701.01081

[40] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2013, pp. 1155–1162.

[41] S. He, N. Pugeault, Y. Mi, and A. Borji, "What catches the eye? Visualizing and understanding deep saliency models," 2018. [Online]. Available: http://arxiv.org/abs/1803.05753

[42] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu, "Predicting eye fixations using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 362–370.

[43] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 818–833.

[44] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. [Online]. Available: http://distill.pub/2016/deconv-checkerboard

[45] Wikipedia Contributors, "Softmax function—Wikipedia, the free encyclopedia," 2018. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Softmax_function&oldid=85021%3870. Accessed on: Jul. 18, 2018.

[46] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 740–755.

[47] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 3, pp. 740–757, Mar. 2019.

[48] N. Riche, M. Duvinage, M. Mancas, B. Gosselin, and T. Dutoit, "Saliency and human fixations: State-of-the-art and study of comparison metrics," in *Proc. IEEE Int. Conf. Comput. Vision*, 2013, pp. 1153–1160.

[49] F. Chollet *et al.*, "Keras," 2015. [Online]. Available: https://github.com/keras-team/keras

[50] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI 16)*, 2016, pp. 265–283.

[51] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*. 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[52] Y. Chen *et al.*, "Dual path networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4467–4475.

[53] S. S. Kruthiventi, V. Gudisa, J. H. Dholakiya, and R. Venkatesh Babu, "Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 5781–5790.

[54] J. Pan, E. Sayrol, X. Giro-i Nieto, K. McGuinness, and N. E. O'Connor, "Shallow and deep convolutional networks for saliency prediction," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 598–606.

[55] N. Liu, J. Han, T. Liu, and X. Li, "Learning to predict eye fixations via multiresolution convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 2, pp. 392–404, Feb. 2018.

[56] S. Xingjian *et al.*, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[57] F. Visin *et al.*, "ReNet: A recurrent neural network based alternative to convolutional networks," 2015. [Online]. Available: http://arxiv.org/abs/1505.00393

[58] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 487–495.

Guosheng Lin received the bachelor's and master's degrees in computer science from South China University of Technology, Guangzhou, China, in 2007 and 2010, respectively, and the Ph.D. degree from The University of Adelaide, Adelaide, SA, Australia, in 2015. He is currently an Assistant Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests are in computer vision and machine learning.

Qiuping Jiang received the Ph.D. degree from the School of Information Science and Engineering, Ningbo University, Ningbo, China, in June 2018. He is currently an Associate Professor with the School of Information Science and Engineering, Ningbo University. From January 2017 to May 2018, he was a Visiting Student with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include image processing, visual perception modeling, and computer vision. He is a Reviewer for several prestigious journals such as the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON MULTIMEDIA, etc. He was the recipient of the JVCI 2017 Best Paper Award Honorable Mention.

Sheng Yang received the B.E. degree in information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015. He is currently working toward the Ph.D. degree with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include image quality assessment, visual saliency prediction, and point cloud analysis.

Weisi Lin (F'16) received the Ph.D. degree from King's College London, London, U.K. He is currently a Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include image processing, visual quality evaluation, and perception-inspired signal modeling, with more than 340 refereed papers published in international journals and conferences. He has been on the Editorial Board of the IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE SIGNAL PROCESSING LETTERS, and *Journal of Visual Communication and Image Representation*. He has been elected as APSIPA (2012/2013) Distinguished Lecturer. He was a Technical-Program Chair for Pacific-Rim Conference on Multimedia 2012, the IEEE International Conference on Multimedia and Expo 2013, and the International Workshop on Quality of Multimedia Experience 2014. He is a Fellow of the Institution of Engineering Technology, and an Honorary Fellow of the Singapore Institute of Engineering Technologists.