

Depthwise Non-local Module for Fast Salient Object Detection Using a Single Thread

Haofeng Li, Guanbin Li, Binbin Yang, Guanqi Chen, Liang Lin, Yizhou Yu

Abstract—Recently deep convolutional neural networks have achieved significant success in salient object detection. However, existing state-of-the-art methods require high-end GPUs to achieve real-time performance, which makes them hard to adapt to low-cost or portable devices. Although generic network architectures have been proposed to speed up inference on mobile devices, they are tailored to the task of image classification or semantic segmentation, and struggle to capture intra-channel and inter-channel correlations that are essential for contrast modeling in salient object detection. Motivated by the above observations, we design a new deep learning algorithm for fast salient object detection. The proposed algorithm for the first time achieves competitive accuracy and high inference efficiency simultaneously with a single CPU thread. Specifically, we propose a novel depthwise non-local module (DNL), which implicitly models contrast via harvesting intra-channel and inter-channel correlations in a self-attention manner. In addition, we introduce a depthwise non-local network architecture that incorporates both depthwise non-local modules and inverted residual blocks. Experimental results show that our proposed network attains very competitive accuracy on a wide range of salient object detection datasets while achieving state-of-the-art efficiency among all existing deep learning based algorithms.

Index Terms—salient object detection, deep neural network, non-local module.

I. INTRODUCTION

SALIENT object detection, which aims to identify the most visually distinctive objects within an image, has been well studied. Developing an accurate salient object detection model benefits a series of applications, such as person re-identification [1], robotic control [2], object detection [3], visual tracking [4] and content-aware image editing [5]. Salient object detection usually serves as a pre-processing component, which not only requires acceptable accuracy but also fast speed and small memory consumption on low-cost devices. Recent deep convolutional neural networks (CNNs) exhibit remarkable performance on many computer vision tasks including salient object detection, due to its strong fitting capacity.

This work was supported in part by the National Natural Science Foundation of China under Grant No.61976250, No.61702565 and No.U1811463, in part by the NSFC-Shenzhen Robotics Projects (U1613211), in part by the Fundamental Research Funds for the Central Universities under Grant No.18lgyy63, and in part by the National High Level Talents Special Support Plan (Ten Thousand Talents Program). (Corresponding authors: Guanbin Li and Yizhou Yu).

H. Li and Y. Yu are with the Department of Computer Science, The University of Hong Kong, Hong Kong. (e-mail: lhaof@foxmail.com; yizhouy@acm.org).

G. Li, B. Yang, G. Chen and L. Lin are with the school of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China (e-mail: liguanbin@mail.sysu.edu.cn; yangbb3@mail2.sysu.edu.cn; chengq26@mail2.sysu.edu.cn; linliang@ieee.org).

In particular, dense labeling methods, which make use of fully convolutional network (FCN) architecture, enjoy high accuracy and efficiency offered by end-to-end training and inference. However, the acceleration of convolution operations is highly dependent on high-performance GPUs, which are typically not supported by mobile devices, embedded devices and low-cost personal computers. Developing a deep learning based salient object detection algorithm, that achieves both fast inference and high-quality results using a single thread, remains a challenging task.

Generic low-cost deep network architectures have been proposed recently for mobile devices. Most of them replace conventional convolutional operators with a combination of depthwise separable convolutions and 1×1 convolutions. The inverted residual block [6] is one of such neural network modules based on depthwise separable convolutions. For example, an inverted residual block first expands the feature at each spatial position to a higher dimension, and then independently applies a convolution operation on each channel slice. Such methods demonstrate desired inference speed on CPUs but their prediction quality is far from satisfactory. The most essential reason behind is that these lightweight methods directly discard correlation modeling at different channels and spatial positions. Such correlation can be taken as context information, which plays an important role in modeling coherence, contrast and uniqueness for salient object detection. Simply borrowing existing generic lightweight network architectures to salient object detection does obtain high efficiency, but their prediction accuracies are far from competitive.

Driven by the above insights, this paper proposes a novel depthwise non-local module, and a fast salient object detection network framework based on the proposed module. This module aims at exploiting relationships among features located at different channel slices or positions. In contrast to traditional convolutional layers that take the vector across all channels at the same spatial position as a feature, the depthwise non-local module considers one column vector or row vector in the same channel as a feature unit, which is described as ‘depthwise’. The proposed module then learns an attention map via calculating pairwise similarity among non-local features within each sub-region. From the resulting attention map, a feature residual is computed in a self-attention way to update the feature map.

Our proposed module has the following strengths. First, the depthwise non-local module overcomes the limitations of inverted residual blocks by explicitly inferring correlations among features that are neither in the same channel nor in the same spatial position. Second, the DNL module can implicitly

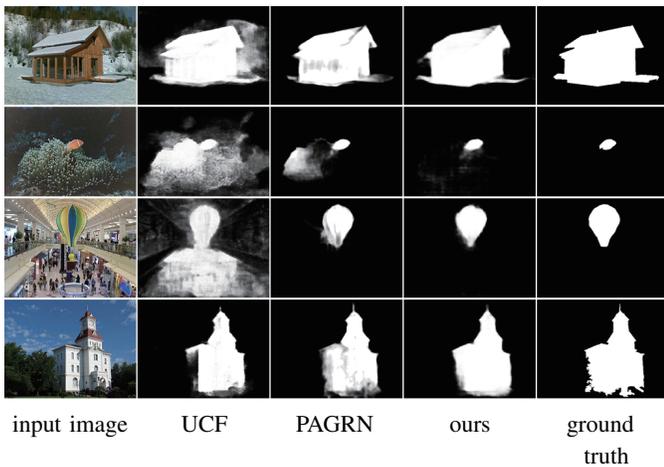


Fig. 1. Comparison among UCF [7], PAGRN [8] and our DNL algorithm. The proposed algorithm not only achieves practical inference time consumption using a single thread, but also presents saliency maps of competitive quality.

model connectivity, contrast and coherence for salient object detection. For example, if image parts are visually similar, their visual features have high attention scores. Thus, their salient features likely update each other and these image parts can be labeled with close saliency values. When a target feature is widely apart from other surrounding features in their latent space, the image region corresponding to the feature may have a different saliency value from other surrounding regions. Third, our proposed module segments an input feature map into sub-regions and only applies self-attention within each sub-region, which lowers its computational cost. As a result, the proposed module adds a very small amount of computation while it considerably enhances the accuracy of a baseline.

This paper has the following contributions.

- We propose a novel depthwise non-local module, which aims at mining intra-channel and inter-channel correlations in context. The proposed module enhances the fitting capacity of inverted residual blocks at the cost of negligible extra inference time.
- We present a fast depthwise non-local neural network, which not only demonstrates state-of-the-art inference speed with a single CPU thread but also attains competitive detection accuracy (as shown in Fig. 1) among deep learning methods.
- We have conducted extensive experiments, which verify the effectiveness and efficiency of the depthwise non-local module and the proposed network framework.

II. RELATED WORK

A. Salient Object Detection

Salient object detection can be solved by computing saliency map with prior knowledge and handcrafted features [9], [10], [11], [12], [13], [14], [15], [16], or training a deep learning model for prediction [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. MBS [12] exploits the cue that background regions are usually connected to the image boundaries, by computing an approximate minimum barrier distance transform. MST [13] employs a similar prior with MBS,

but computes an exact distance transform with a minimum spanning tree to measure boundary connectivity. MDC [14] suggests that background pixels display low contrast in at least one dimension and proposes minimum directional contrast as raw saliency for each pixel. Priors based methods enjoy real-time efficiency but cannot attain state-of-the-art results. Deep learning based salient object detection models can be roughly divided into two groups, including sparse and dense labeling. MDF [17] employs deep CNNs to extract multi-scale features and predict saliency values for image segments of different levels. Zeng *et al.* [30] formulate saliency detection as a non-cooperative game, where image regions as players choose to be foreground or background. Zhang *et al.* [31] convert an image into a sparsely-connected graph of regions, and compute saliency via an absorbing Markov chain. Qin *et al.* [26] develop Single-layer Cellular Automata (SCA) that can utilize the intrinsic correlations of similar image patches to locate salient objects, based on deep learning features. These sparse labeling methods require dividing an input image into hundreds of segments and estimating saliency value for each segment, which is not efficient for real-time applications. To name a few dense labeling methods, DSS [32] introduces a series of side output layers and short connections to combine the advantages of low-level and high-level features. PAGRN [8] is a progressive attention driven framework based on multi-path recurrent feedback. Wang *et al.* [33] propose a global recurrent localization network to locate salient objects, and a local boundary refinement network to capture pixel relations. Liu *et al.* [34] integrate a global guidance module and a feature aggregation module into a U-shape architecture.

B. Fast Convolutional Neural Network

Designing efficient and lightweight neural networks [35], [36], [37], [38], [39], [40], [41], [42], [43] has recently become popular in the community. Han *et al.* [36] propose a network pruning pipeline that is first trained to learn which connections are important, and then discards the unimportant connections. Factorized Convolutional Neural Networks [41] unravel the 3D convolution operation in a convolution layer as spatial convolutions in each channel and a linear projection across channels, to reduce the computation. He *et al.* [42] introduce a channel pruning method which alternatively select the most representative channels based on a LASSO regression, and reconstruct the output feature maps with linear least squares. ShuffleNet [43] proposes a pointwise group convolution that separates convolution filters into groups, and ‘channel shuffle’ that permutes the channels in a group. MobileNetV2 [6] utilizes an inverted residual structure that is composed of two pointwise convolutions and a depthwise separable convolution. Some existing state-of-the-art fast deep neural networks employ depthwise separable convolutions that are lightweight but lack intra-channel and inter-channel correlations mining.

C. Self-Attention and Non-local Modeling

Computational models that exploit pairwise similarities as self-attention scores among dense positions or nodes within some non-local regions, have been widely studied in the field

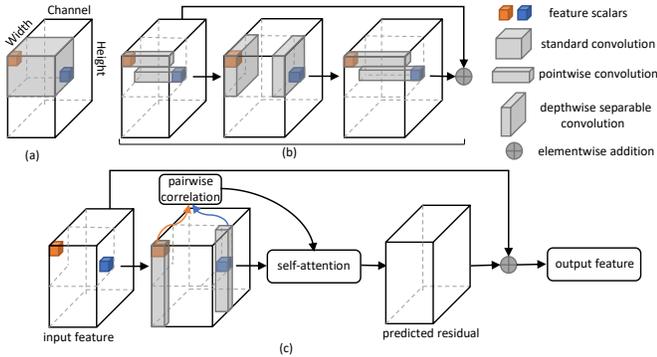


Fig. 2. (a) Standard Convolution. (b) Inverted Residual. Before the element-wise addition in inverted residual, the shapes of the two feature maps are aligned with each other. Notice that the correlation of two spatially close scalars (in orange and blue colors respectively) is not directly captured in an inverted residual, while a standard convolution can predict outputs based on them together. (c) Our proposed depthwise non-local module. The proposed module can capture the correlation between two feature vectors, which are located at different channels and spatial positions.

of natural language processing and computer vision [44], [45], [46], [47], [48], [49], [50], [51], [52]. Self-attention model for machine translation [50] learns a feature for some node by attending to all other nodes in the same sequence and taking weighted summation of their embedded features in some latent space. Non-local means algorithm [45] denoises an image by replacing a pixel with a non-local averaging of all pixels in the image. The non-local averaging utilizes similarity between pixels as weights. Block-matching 3D (BM3D) [53] applies collaborative filtering on a group of similar non-local image patches and achieves competitive image denoising results, even compared to deep learning based methods. Efros and Leung [47] synthesize texture by growing one pixel at a time. They determine pixel value by locating all image patches matching with the target position to fill. Dense conditional random field (CRF) [48] models long-range dependencies by introducing a pairwise energy term that is weighted by the similarity of two nodes. Li et al. [54] propose a non-locally enhanced encoder-decoder network which can learn more accurate feature for rain streaks and preserve better image details during de-raining. Besides from low-level tasks, Wang *et al.* [51] propose a non-local neural network that can harvest long-range spatiotemporal relations for high-level problems, such as video classifications. Such models can learn features from long-range dependencies that are potential to model contrast and coherency in salient object detection. Most existing non-local models work in the spatial dimensions of images or the spatiotemporal dimensions of videos.

III. METHOD

A. Depthwise Non-local Module

In this section, we introduce a novel depthwise non-local module, that efficiently enhances inverted residual blocks with channel-wise coherence and contrast learning. Inverted residual is an efficient neural network block built on top of depthwise separable convolutions. In the following, we briefly review inverted residual blocks and depthwise separable

convolutions as preliminaries. Let I be a $C \times H \times W$ input feature map. C , H and W denotes the channel number (depth), the height and the width of I respectively. k , i and j are used as the indices of depth, height and width respectively. For example, $I_{i,j}$ is a vector of length C and $I_{k,i}$ has W elements. Consider a regular $K \times K$ convolution layer with C output channels. Its total number of weights is $CK^2 \times C$. The time complexity of applying convolution at one position is C^2K^2 . As for a depthwise separable convolution layer with C output channels, it has C independent convolution kernels, each of which has a total of $K^2 \times 1$ weights. Performing depthwise separable convolution at one position costs CK^2 . As shown in Fig. 2(b), an inverted residual block is composed of a 1×1 convolution, a depthwise convolution and another 1×1 convolution. These two 1×1 convolutions aim at aggregating features across channels. However, for a pair of positions (the orange and blue feature scalars in Fig. 2), $I_{k,i,j}$ and $I_{k',i',j'}$ ($k \neq k'$ and $(i,j) \neq (i',j')$), their correlation cannot be directly captured by the depthwise separable convolution or the pointwise 1×1 convolutions even when they are located within each other's neighborhood.

To efficiently harvest intra-channel correlations, the depthwise non-local module considers $I_{k,i}$ or $I_{k,j}$ (shown in Fig. 4(b)) rather than $I_{i,j}$ (shown in Fig. 4(a)) as a feature vector. $I_{k,i}$ and $I_{k,j}$ represent some feature of their corresponding horizontal and vertical image region respectively. The proposed module is a residual block with two possible types of residual layers. One type of layers is called vertical or *vertical-split* layer and the other is horizontal or *horizontal-split* layer. In a vertical-split layer, we take $I_{k,j}$ as a feature vector. In a horizontal-split layer, $I_{k,i}$ is taken as a feature vector. These two types of layers are designed in a similar and symmetric way.

In the vertical-split layer shown in Fig. 3, the input feature map can be seen as $C \times W$ feature vectors and each vector has H elements. To exploit cross-channel features, we first compute an attention map by measuring pairwise similarities among these CW vectors. Thus the attention map A is of size $CW \times CW$. Consider two arbitrary features, $I_{k,j}$ and $I_{k',j'}$, whose indices in the attention map are p and q . Their attention score can be calculated in a bilinear form.

$$A_{p,q} = I_{k,j}^T U_\theta^T V_\phi I_{k',j'}, \quad (1)$$

where U_θ and V_ϕ are two learnable matrices of two embedding layers θ and ϕ . The size of U_θ and V_ϕ is $H^A \times H$. The bilinear form can be seen as mapping $I_{k,j}$ and $I_{k',j'}$ into a latent space R^{H^A} by linear transformations, and then computing the dot product between the transformed features as similarity. Importantly, we discuss whether all $C \times W$ feature vectors share the same transform matrix U_θ in the following. $I_{k,j}$ with the same k but different j , which denotes different spatial region within the same latent space, can share the same transform matrix. $I_{k,j}$ with different k should not utilize the same transform matrix, since different k implies different latent space. Thus, only feature vectors with the same channel share the same parameter U_θ . Since the layers θ and ϕ embed different channels of an input feature with different weights, we name θ and ϕ as ‘Channel-wise Embedding Layer’ as

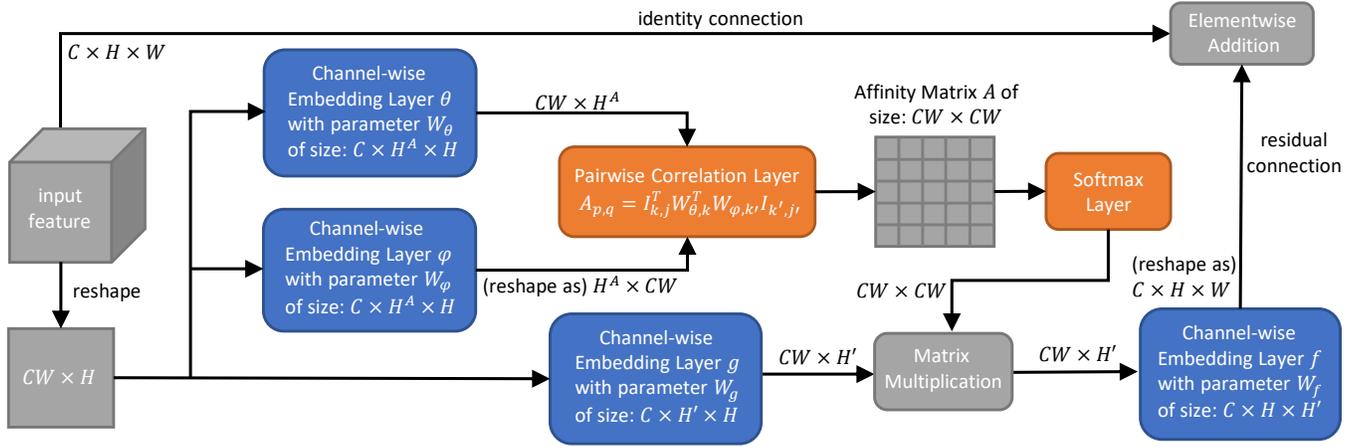


Fig. 3. Vanilla Depthwise Non-Local Module (vertical-split). The proposed module contains four channel-wise embedding layers, a pairwise correlation layer and a softmax layer. The softmax layer normalizes each row of the affinity matrix A . The matrix multiplication serves as taking weighted summation of the output of Layer g , with the softmax output as attention weights.

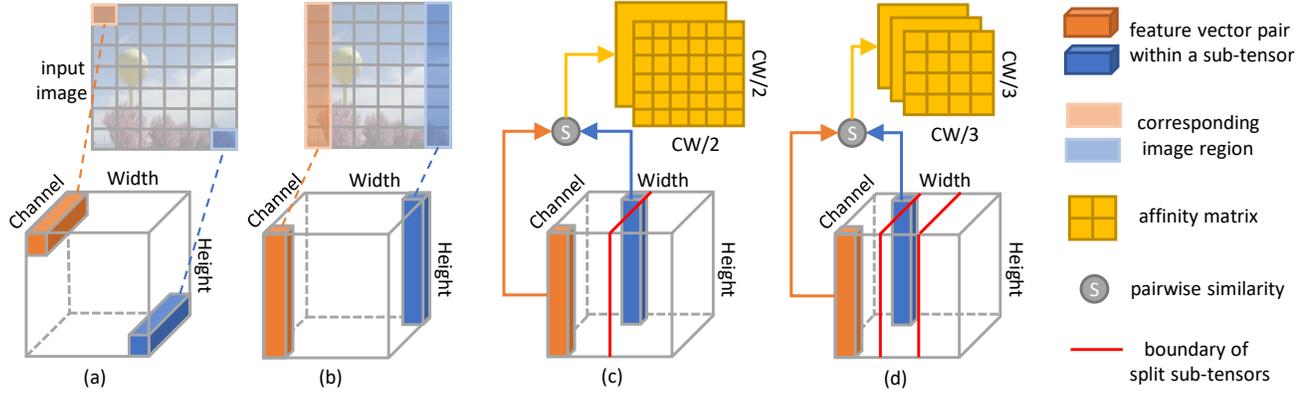


Fig. 4. (a) Vanilla (Spatial) Non-local Module; (b) Vanilla Depthwise Non-local Module (vertical-split); (c) Divide-and-Conquer Depthwise Non-local Module (vertical-split, split number, $s=2$); (d) split number, $s=3$. Dense pairwise similarity is computed between each pair of feature vectors (the orange bar and the blue bar) within the same region/sub-region. In (c), the $C \times W$ plane is split into 2 sub-regions.

displayed in Fig. 3. We reformulate the bilinear form (shown as ‘pairwise correlation’ in Fig. 3) in the below.

$$A_{p,q} = I_{k,j}^T W_{\theta,k}^T W_{\phi,k'} I_{k',j'} \quad (2)$$

where W_{θ} and W_{ϕ} are $C \times H^A \times H$ matrices. $W_{\theta,k}$ and $W_{\phi,k'}$ are $H^A \times H$ matrices corresponding to U_{θ} and V_{ϕ} . We claim that the above attention model is promising in modeling contrast and connectivity for salient object detection. Small $A_{p,q}$ indicates high contrast between $I_{k,j}$ and $I_{k',j'}$. If $I_{k,j}$ and $I_{k',j'}$ are spatially close to each other, large $A_{p,q}$ suggests that their corresponding regions are connected in the saliency map.

To leverage these cues, we conceive a strategy to propagate contrast information or saliency value from $I_{k',j'}$ to $I_{k,j}$, according to their correlation. The strategy learns a feature residual by taking attention weighted transformation of the feature map.

$$\tilde{I} = f(\text{softmax}(A)g(I)), \quad (3)$$

where \tilde{I} denotes the above-mentioned feature residual. $\text{softmax}(\cdot)$ normalizes each row of matrix A . $g(\cdot)$ denotes a transformation of the input feature map I . As de-

finied below, $g(\cdot)$ adopts linear transformations with different parameters for different channels of I . Let $g(I) = [I_1^T W_{g,1}, \dots, I_k^T W_{g,k}, \dots, I_C^T W_{g,C}]$, where I_k is a $H \times W$ matrix representing the k -th channel of feature map I , $W_{g,k}$ is a $H \times H'$ matrix representing the linear transformation for the k -th channel of the feature map, $[\cdot]$ denotes the concatenation along the first dimension. Thus $g(I)$ is a $CW \times H'$ matrix. $\text{softmax}(A)g(I)$ is an attention weighted linear transformation of $g(I)$, shown as ‘Matrix Multiplication’ in Fig. 3. Since the size of $\text{softmax}(A)g(I)$ is $CW \times H'$, another transformation f is required to map it into the R^H space. Let $y = \text{softmax}(A)g(I)$. Then $f(y) = [(y_1 W_{f,1})^T, \dots, (y_k W_{f,k})^T, \dots, (y_C W_{f,C})^T]$, where y is a $CW \times H'$ matrix, and y is reshaped into a $C \times W \times H'$ tensor before $f(\cdot)$ is applied, W_f is a $C \times H' \times H$ tensor, and $f(y)$ is a $CH \times W$ matrix. Finally, $f(y)$ is reshaped into a $C \times H \times W$ residual tensor \tilde{I} , and I is updated by adding the residual tensor and I together. The output of DNL modules is calculated as $O = I + \tilde{I} = I + f(y)$, where O represents the output feature map, and the reshaping operators are omitted.

The horizontal-split layer is a symmetric form of the vertical-split layer. We simply summarize its process and

describe the differences from the vertical one.

$$A_{p,q} = I_{k,i}^T W_{\theta}^T W_{\phi} W_{\phi,k'} I_{k',i'}, \quad (4)$$

$$g(I) = [I_1 W_{g,1}, \dots, I_k W_{g,k}, \dots, I_C W_{g,C}], \quad (5)$$

$$O = I + [y_1 W_{f,1}, \dots, y_k W_{f,k}, \dots, y_C W_{f,C}], \quad (6)$$

where the size of attention map A is $CH \times CH$, and $A_{p,q}$ denotes pairwise similarity between two arbitrary features $I_{k,i}$ and $I_{k',i'}$ ($1 \leq k, k' \leq C, 1 \leq i, i' \leq H$), whose indices in the attention map A are p and q . $I_{k,i}$ is a feature vector of length W . W_{θ} , W_{ϕ} , W_g and W_f are $C \times W^A \times W$, $C \times W^A \times W$, $C \times W \times W'$ and $C \times W' \times W$ tensors, respectively. y is computed in the same way as in the vertical-split layer, and is reshaped into a $C \times H \times W'$ tensor before f is applied. $f(y)$ is converted to a $C \times H \times W$ tensor before I is updated. Note that all channel-wise embedding layers, θ , ϕ , $g(\cdot)$ and $f(\cdot)$, have bias parameters. For example, $y_k W_{f,k}$ should actually be $[y_k^{*T}, 1]^T [W_{f,k}^*, B_{f,k}]$. For simplicity, all bias terms have been omitted in the above formulations.

B. Divide-and-Conquer

In this section, we accelerate the naive depthwise non-local module by dividing an input feature map into multiple sub-tensors. A few rationales support the divide-and-conquer strategy. First, the naive DNL module computes dense pairwise similarities, which is too computationally expensive for a fast neural network module. Second, the divide-and-conquer strategy still maintains spatial coherence in the resulting saliency map. If there is a strong similarity between spatially adjacent features, it is most likely that these two features or segments belong to the same object. Propagating saliency values between such pairs of features can likely improve the accuracy of saliency prediction. In the naive vertical-split DNL module, all pairs of feature vectors on the $C \times W$ plane are used to measure similarity, as shown in Fig. 4(b). For the divide-and-conquer DNL shown in Fig. 4(c), the feature tensor is divided into 2 sub-tensors, and vector pairs are only sampled from the same sub-tensor. For each sub-tensor, a smaller affinity matrix is obtained by calculating the pairwise correlation. The softmax operation is separately applied for each affinity matrix. Different sub-tensors still share the same W_{θ} and W_{ϕ} . The number of sub-tensors is controlled by split number s .

C. Complexity Analysis

In this section, we analyze the space and time complexities of the vanilla depthwise non-local module and its divide-and-conquer version. This analysis can help us determine the values of hyper-parameters and the location of the proposed module in our network architecture.

Let us first discuss the space complexity of a vanilla depthwise non-local module. We assume that all variables are released after inference. Only parameters and intermediate variables are considered. The size of W_{θ} , W_{ϕ} , W_g and W_f in a vertical-split layer is respectively $C \times H^A \times H$, $C \times H^A \times H$, $C \times H \times H'$ and $C \times H' \times H$ while their size is $C \times W^A \times W$, $C \times W^A \times W$, $C \times W \times W'$ and $C \times W' \times W$ in a horizontal-split

TABLE I
COMPLEXITY OF VANILLA DEPTHWISE NON-LOCAL MODULE AND ITS
DIVIDE-AND-CONQUER VARIANT.

Complexity	Vanilla	Divide-and-Conquer
Space	$\mathcal{O}(C^2(H^2 + W^2))$	$\mathcal{O}(\frac{1}{s}C^2(H^2 + W^2))$
Time	$\mathcal{O}(C^2HW(H + W))$	$\mathcal{O}(\frac{1}{s}C^2HW(H + W))$

layer. Without taking bias terms into account, the total number of parameters is $2C(H(H^A + H') + W(W^A + W'))$. The size of intermediate variables A , $g(I)$, y and \tilde{I} is respectively $CW \times CW$, $CW \times W'$, $CW \times W'$, and $C \times H \times W$ in a vertical layer. In a horizontal one, their size is $CH \times CH$, $CH \times H'$, $CH \times H'$, and $C \times H \times W$. The space complexity of intermediate variables is $C^2(H^2 + W^2) + 2C(HH' + HW + WW')$. The space complexity of a depthwise non-local module is $\mathcal{O}(C^2(H^2 + W^2))$.

For time complexity, we count the number of multiplications and additions (MAdds). In a vertical-split layer, applying transformations W_{θ} and W_{ϕ} costs $CWHH^A$ while computing pairwise similarity costs $C^2W^2H^A$. The time complexity of $\text{softmax}(\cdot)$, $g(\cdot)$, $f(\cdot)$ and $\text{softmax}(A)g(I)$ is respectively C^2W^2 , $CWHH'$, $CWHH'$ and C^2W^2H' . In a horizontal-split layer, computing A costs $2CHWW^A + C^2H^2W^A$. The time complexity of $\text{softmax}(\cdot)$, $g(\cdot)$, $f(\cdot)$ and $\text{softmax}(A)g(I)$ is respectively C^2H^2 , $CHWW'$, $CHWW'$ and C^2H^2W' . The total number of multiplications and additions is $C^2W^2(H^A + H' + 1) + C^2H^2(W^A + W' + 1) + 2CHW(H^A + H' + W^A + W')$. The time complexity of the proposed module is $\mathcal{O}(C^2HW(H + W))$.

Next, we analyze the computational cost of a divide-and-conquer depthwise non-local module. Its space complexity is reduced by a factor of s since the size of A becomes $\frac{1}{s}C^2W^2$ in a vertical layer and $\frac{1}{s}C^2H^2$ in a horizontal layer. As for time complexity, computing attention scores in a vertical-split layer costs $2CWHH^A + s(\frac{CW}{s})^2H^A$. The time complexity of $\text{softmax}(\cdot)$, $g(\cdot)$, $f(\cdot)$ and $\text{softmax}(A)g(I)$ is respectively $\frac{C^2W^2}{s}$, $CWHH'$, $CWHH'$ and $s(\frac{CW}{s})^2H'$. The computation in a vertical-split layer costs $\frac{1}{s}C^2W^2(H^A + H' + 1) + 2CHW(H^A + H')$ and a horizontal-split layer $\frac{1}{s}C^2H^2(W^A + W' + 1) + 2CHW(W^A + W')$. The time complexity of the accelerated variant is $\mathcal{O}(\frac{1}{s}C^2HW(H + W))$. Notice that H^A/H' and W^A/W' are set as $H/2$ and $W/2$ respectively in our implementation.

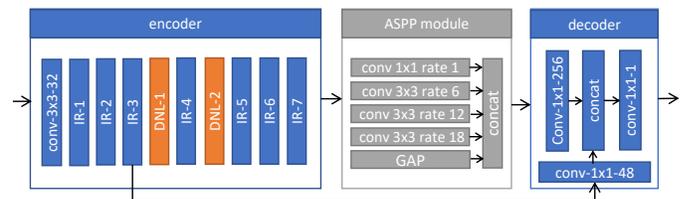


Fig. 5. Depthwise Non-Local Deep Neural Network. As shown in the above, IR denotes inverted residual module while DNL denotes depthwise non-local module. ‘rate’ refers to the dilation rate of a convolution. GAP denotes global average pooling.

D. Model Architecture

In this section, we develop a depthwise non-local neural network based on our proposed module. The proposed network consists of an encoder, an atrous spatial pyramid pooling (ASPP) module [55] and a decoder, as shown in Figure 5. The encoder contains 7 inverted residual modules, 2 depthwise non-local modules and several regular convolution layers. Each inverted residual (IR) module is composed of one or multiple inverted residual blocks. The hyper-parameters of these above-mentioned IR modules follow the setting in [6].

Our proposed depthwise non-local modules are located between some of the inverted residual modules to strengthen non-local correlations among all feature map channels. Since the computational complexity of the proposed module grows quadratically with respect to the number of channels and cubically w.r.t the spatial resolution of input feature map, we place the proposed module at middle levels of the network. The first depthwise non-local module is placed behind the third inverted residual module. The second one is inserted after the fourth inverted residual module. After the third inverted residual module, the feature map has shrunk to the smallest spatial size, $1/8$ of the original image size, while the number of channels is still reasonably small. Thus positioning the proposed modules at middle levels helps lower the computational cost incurred by these modules. Theoretically DNL modules can be placed at any position of a backbone network. How the position of a DNL module affect its performance and efficiency is investigated in Section IV-C.

The atrous spatial pyramid pooling module in Fig. 5 concatenates the feature maps produced from the five parallel layers along the depth dimension. These five parallel layers are respectively a 1×1 pointwise convolution, three dilated 3×3 convolutions and an image pooling layer. The dilation rates of the three atrous convolutions are 6, 12, and 18 respectively. All of these five parallel layers produce 256-channel feature maps. The image pooling layer consists of a global spatial averaging sub-layer and a pointwise convolution that converts the number of output channels to 256. The ASPP module produces a 1280-channel feature map. The decoder takes the output of both the ASPP module and the third inverted residual module as a combination of high-level and low-level inputs. The decoder reduces the number of channels in the high-level input to 256 while increasing the number of channels in the low-level input to 48 using pointwise convolutions following a 2D batch normalization layer. Finally, the low-level and high-level features are concatenated to predict a dense saliency map via a 1×1 convolution.

IV. EXPERIMENTS

In the experiment section, salient object detection methods are tested on DUT-OMRON [56], ECSSD [57], HKU-IS [58] test set, PASCAL-S [59] and DUTS [60]. All the above datasets provide dense pixel-level annotations. DUT-OMRON contains 5168 challenging images which has one or more salient objects. ECSSD has 1000 images. HKU-IS includes a train set of 2500 images, a validation set of 500 images and a test set of 1447 images. PASCAL-S consists of 850 images.

Threshold is chosen as 0.5 to binarize masks of PASCAL-S, as suggested in [59]. Notice that all salient object detection models are not trained on any subsets of DUT-OMRON and all 5168 images are utilized as testing samples. Thus DUT-OMRON is a challenging benchmark which can reveal the generalization capability of a salient object detection model. HKU-IS is another challenging dataset in which many images contain multiple ground-truth objects. Our proposed method is trained with 5000 images from MSRA-B train set and HKU-IS train set. The optimization algorithm is SGD with weight decay $5e-4$, momentum 0.9 and initial learning rate $1e-7$. We adopt poly policy with power 0.9 to tune the learning rate. The proposed network is trained for 300 epochs. Pytorch 0.4.1 with MKL backend is used for all deep learning methods. For the methods whose released model is not trained with Pytorch, their model weights are copied to an implementation of Pytorch. For fair comparisons, the efficiency of the mentioned salient object detection algorithms are evaluated on the same personal computer, which has an Intel i7-6850k CPU with 3.60 GHz base frequency, a GeForce GTX 1080 Ti GPU and 32GB memory.

A. Comparison on Quality of Saliency Maps

To evaluate the quality of saliency maps, we adopt maximum F-measure (maxF) [61], mean absolute error (MAE) and structure-measure [62] (S-m) as criteria. To compute maximum F-measure, we first sample a list of thresholds. Given a threshold, the average of precision and recall is computed for all saliency predictions in a dataset. Then F_β is defined as:

$$F_\beta = \frac{(1 + \beta^2) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \quad (7)$$

where β controls the relative importance between precision and recall. β^2 is selected as 0.3, according to [61]. MAE is computed as the average of pixel-level absolute difference between predictions and ground-truth annotations, as shown in:

$$MAE = \frac{1}{HW} \sum_{h=1}^H \sum_{w=1}^W |P_{h,w} - G_{h,w}| \quad (8)$$

where P denotes a binarized saliency prediction and G denotes its corresponding binary ground-truth. H and W are height and width of images. h and w are the corresponding indices. Different from estimating pixel-wise errors, S-measure, recently proposed in [62] is adopted to estimate structural similarity between predictions and ground-truth. It is defined as a weighted sum of an object-aware measure and a region-aware measure. Formal definition of structure-measure can be found in [62].

As shown in TABLE II and Fig. 6, the proposed DNL network is compared with existing salient object detection models, RAS [27], PAGRN [8], UCF [7], NLDF [22], DSS [32], RFCN [23], DCL [63], DS [20], MDF [58] and two generic lightweight architectures originally proposed for image classification and semantic segmentation, including MobileNet-V2 [6] and ShuffleNet [43]. As suggested in [6], MobileNet-V2 serves as an encoder integrated with a DeepLab-V3+ architecture [55] to solve semantic segmentation tasks. To detect

TABLE II
COMPARISON AMONG THE STATE-OF-THE-ART AND OURS. * METHODS ARE ORIGINALLY PROPOSED FOR IMAGE CLASSIFICATIONS AND SEGMENTATIONS.

	DUT-OMRON			ECSSD			HKU-IS			PASCAL-S			DUTS		
	maxF	MAE	S-m	maxF	MAE	S-m	maxF	MAE	S-m	maxF	MAE	S-m	maxF	MAE	S-m
RAS	0.7848	0.0633	0.8119	0.9203	0.0551	0.8935	0.9116	0.0449	0.8875	0.8319	0.1021	0.7978	0.8310	0.0591	0.8281
PAGR _N	0.7709	0.0709	0.7751	0.9268	0.0609	0.8892	0.9187	0.0475	0.8891	0.8531	0.0921	0.8190	0.8541	0.0549	0.8254
UCF	0.7365	0.1318	0.7578	0.9097	0.0790	0.8816	0.8866	0.0749	0.8643	0.8217	0.1292	0.7999	0.7700	0.1178	0.7710
NLDF	0.7532	0.0796	0.7704	0.9050	0.0626	0.8747	0.9017	0.0480	0.8782	0.8278	0.0990	0.8036	0.8156	0.0649	0.8052
DSS	0.7604	0.0744	0.7892	0.9078	0.0620	0.8836	0.9005	0.0499	0.8805	0.8262	0.1029	0.8025	0.8130	0.0647	0.8168
RFCN	0.7332	0.0782	0.7503	0.8867	0.0765	0.8368	0.8832	0.0572	0.8361	0.8284	0.0996	0.7895	0.7702	0.0744	0.7506
DCL	0.7260	0.0944	0.7498	0.8884	0.0717	0.8672	0.8823	0.0584	0.8650	0.8053	0.1092	0.7930	0.7756	0.0787	0.7859
DS	0.7449	0.1204	0.7502	0.8824	0.1217	0.8206	0.8661	0.0791	0.8531	0.8109	0.1472	0.7715	0.7756	0.0894	0.7916
MDF	0.6944	0.0916	0.7208	0.8316	0.1050	0.7761	0.8605	0.1291	0.8101	0.7655	0.1451	0.6935	0.7285	0.0995	0.7232
*MobileNetV2	0.7446	0.0871	0.7595	0.8901	0.0737	0.8614	0.8979	0.0537	0.8756	0.7716	0.1318	0.7592	0.7613	0.0806	0.7720
*ShuffleNet	0.7300	0.0946	0.7506	0.8763	0.0830	0.8443	0.8914	0.0558	0.8701	0.7856	0.1216	0.7709	0.7698	0.0845	0.7740
ours	0.7795	0.0779	0.7981	0.9096	0.0646	0.8851	0.9133	0.0451	0.8974	0.8229	0.1065	0.8031	0.8081	0.0731	0.8071

salient objects, the output channels of the last convolution in DeepLab-V3+ is adjusted to 1. Similar to MobileNet-V2, ShuffleNet is also modified as an encoder with DeepLab-V3+. We fine-tune these two generic frameworks with our training data for saliency prediction. The proposed method significantly outperforms MobileNet-V2 and ShuffleNet on all four benchmarks and three criteria since they fail to capture the contrast as well as the channel-wise coherence information which is essential for saliency inference. Our proposed model obtains the second best maximum F-measure of 0.7795 and the best S-measure of 0.7981 on the large and challenging dataset DUT-OMRON. The DNL network outperforms the third best PAGR_N by 0.9% maxF and DSS by 0.9% S-measure. Noted that our method is not trained on any subsets of DUT-OMRON but tested on the all 5168 images, which suggests that the DNL network possesses strong generalization capability to achieve stable performance in real applications. The proposed DNL network presents the second best maxF, the second smallest MAE and the best S-measure on the HKU-IS dataset. Particularly, the proposed method surpasses the second best PAGR_N by 0.8% S-measure. Our proposed method show the best results on two challenging benchmarks DUT-OMRON and HKU-IS, which indicates that the proposed network enjoys superior generalization and is comparable to the state-of-the-arts.

B. Comparison on Efficiency

To evaluate the efficiency of the proposed methods and existing neural network models, this section utilizes CPU time, GPU time, memory usage (denoted as Mem), number of parameters (denoted as Params), MAdds [6] and time complexity as criteria. CPU time is computed using a single CPU thread while GPU time is measured with a single GPU. Batch size is set as 1 for all neural models. Time cost by file input/output is not included but time-consuming preprocessings such as computing prior maps and superpixel segmentation are taken into accounts. Each model sequentially infers 50 randomly selected images from HKU-IS. The peak memory cost during inference is logged as the memory usage. Params is the number of learnable parameters in a neural

model and it determines the disk space consumed. MAdds is the number of multiplications and additions, calculated by setting the input size of each method as its default size. Time complexity denoted as ‘Complexity’ in TABLE III is the number of multiplications and additions with respect to input size that is viewed as variables H and W .

TABLE III
EFFICIENCY OF THE STATE-OF-THE-ART AND OURS.

	CPUTime /secs	GPUPTime /secs	Mem /MB	Params /M	MAdds /B	Complexity /HW
RAS	2.0457	0.0355	5023	20.23	54.56	421.0K
PAGR _N	—	—	—	23.63	100.4	805.8K
UCF	3.4696	0.0886	5307	29.43	123.3	614.4K
NLDF	2.6051	0.0340	1709	35.48	354.6	2863K
DSS	3.4451	0.0339	1587	62.22	127.5	984.0K
RFCN	2.7190	0.0691	4833	53.00	113.9	455.7K
DCL	2.5820	0.0867	4069	66.31	797.6	3031K
DS	2.4588	0.0609	4799	50.37	106.7	426.7K
MDF	897.68	24.996	1591	75.68	7533	149.9M
ours	0.3993	0.0113	605	5.320	9.567	73.82K

As shown in TABLE III, MB denotes million bytes. K, M and B denote thousands, millions and billions respectively. HW represents the product of input height and width. Since the implementation of PAGR_N is not available, we only present its theoretical efficiency including parameters, MAdds and time complexity. Most existing CNN based methods predict a saliency map with more than 2.5 seconds on CPU while our proposed network takes less than 0.4 seconds to infer an image. Achieving the fastest CPU inference, our method is $5\times$ faster than the second best RAS. The proposed method also demonstrates the most efficient inference with GPU, and it is $3\times$ faster than the second best DSS. Most methods use 1500-5000 MB memory during inference, which is too expensive for a preprocessing component. Meanwhile our proposed method costs 600 MB running memory, less than 40% of the second best. Besides, our proposed method has the least parameters, less than 25% of the second least. Most models consume more than 100 MB storage while our method only costs about 20 MB. Our proposed method obtains the minimum MAdds less than 20% of the second best. The time complexity of the

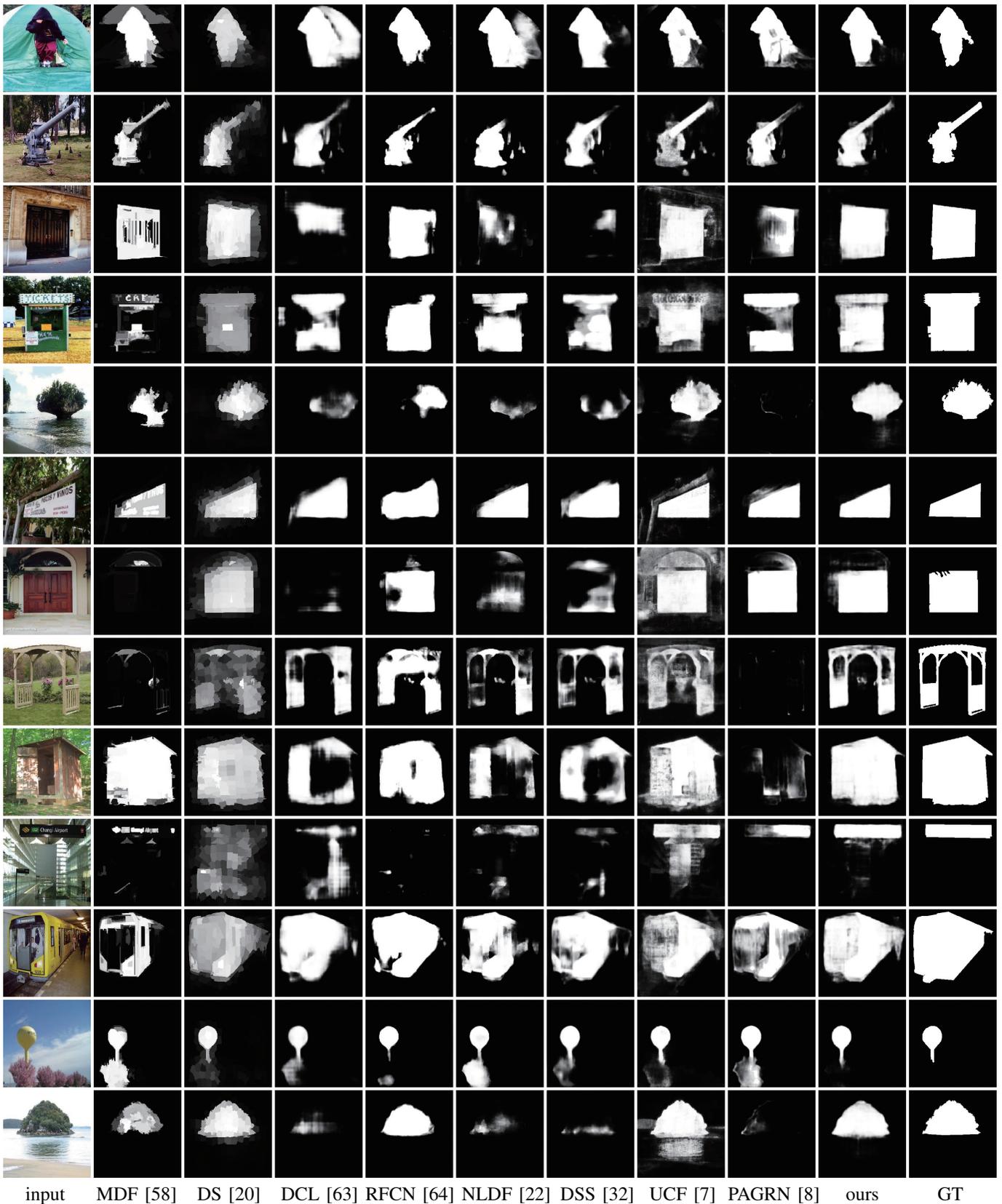


Fig. 6. Qualitative comparison among the state-of-the-art and ours. As shown in the above, the proposed method is compared with MDF, DS, DCL, RFCN, NLDF, DSS, UCF and PAGRN on the DUT-OMRON benchmark. Our proposed method successfully segments complete foreground objects with consistent saliency value and sharp boundaries.

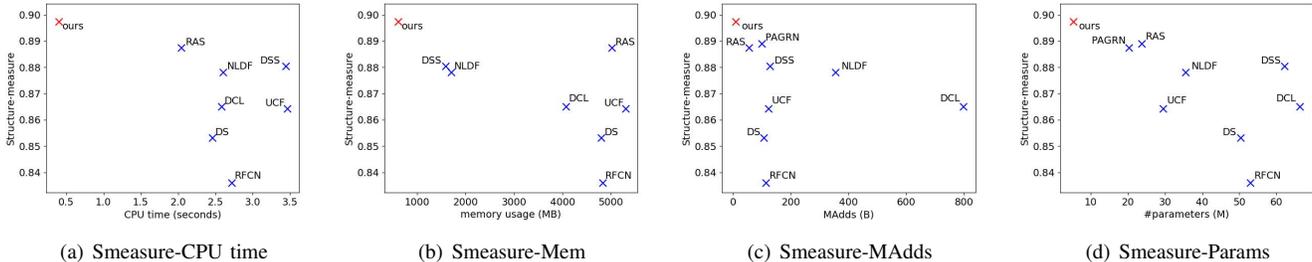


Fig. 7. Comparisons on Efficiency and Quality. To measure the efficiency and quality of salient object detection models at the same time, the above scatter diagrams take an efficiency metric as horizontal axis and a quality metric as vertical axis. Our proposed method is always located at the upper left of these diagrams, which indicate the best trade-off between efficiency and accuracy.

DNL network is also the lowest and 6 times less than the second lowest RAS. Note that the time complexity of DNL modules actually contains terms with respect to $HW(H+W)$. For convenient comparison we simplify the formula by fixing input size $H \times W$ as default size 360×360 . To sum up, our proposed network enjoys the fastest inference speed on both CPU and GPU, consumes the least memory and disk storage, and shows the lowest theoretical complexity, in comparison to existing deep learning models.

To simultaneously evaluate the efficiency and quality of our proposed method, we plot an efficiency metric and a quality metric on the same scatter diagram (shown in Fig. 7), with the efficiency metric as horizontal axis and the quality metric as vertical axis. For quality metric, larger S-measure means more accurate predictions while smaller value means lower cost for efficiency metric. Thus the best method balancing accuracy and efficiency should be located at the upper-left in an Efficiency-Quality scatter diagram. As shown in Fig. 7, our proposed method achieves the best trade-off between efficiency and quality, on the scatter diagrams of Smeasure-CPU time, Smeasure-Mem, Smeasure-MAdds and Smeasure-Params.

C. Ablation Study

This section verifies the effectiveness of DNL module and investigates how the number of splits affects the performance of the proposed network. The baseline is built by removing all DNL modules from the proposed network. The baseline has exactly the same architecture as MobileNetV2 but has different input size. We use Precision-Recall curves and Threshold- F_β measure curves to compare our proposed method with the baseline. To draw these curves, a list of evenly spaced thresholds is sampled. For each threshold, a tuple of precision, recall and F_β measure is calculated, with $\beta^2 = 0.3$. Then we plot the pairs of (recall, precision) in Fig. 8, and the pairs of (threshold, F_β measure) in Fig. 9. As shown in Fig. 8 and Fig. 9, the proposed module effectively improve the prediction accuracy of the baseline on all three benchmarks.

As TABLE IV displays, Split-9 denotes an accelerated DNL module that divides the input feature tensor into 9 sub-tensors. For Split- s ($s = 1, 3, 5, 9$), DNL modules are located after IR-3 and IR-4 shown in Fig. 5. For IR6-splits ($s = 1, 5, 10$), a DNL module is inserted after IR-6. As shown in TABLE IV,

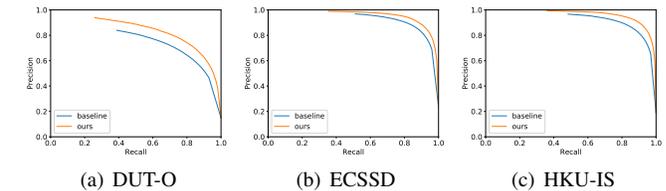


Fig. 8. Ablation Study on Precision-Recall Curves. In the above graphs, the blue curve denotes the baseline model without deploying any proposed modules, while the orange one denotes our proposed DNL network. Our proposed method obtains higher precision than the baseline, for the same recall value.

Split-9 surpasses the baseline by 3.9% maxF, 1.2% MAE and 4.1% S-m on the HKU-IS dataset. The performance of Split-5 is quite close to that of Split-9. Split-9 marginally outperforms Split-1 by 0.11% maxF and 0.22% S-measure. For IR6-splits, similarly, IR6-split10 exceeds IR6-split1 by 0.9% maxF, 0.3% MAE and 0.7% S-m. The above results suggest that DNL modules effectively improve the baseline and the splits number does not affect much of the prediction quality. Larger splits number could lead to slight improvement, since it helps to maintain better spatial coherence as discussed in Section III-B. If the splits number is larger, then the spatial size of each sub-region becomes smaller. Feature vectors within the same sub-region are more likely to belong to background or the same object. In such cases, non-local pairwise correlations help propagate saliency score of a feature to its adjacent features, which models spatial coherence. Compared with IR6-split1 whose DNL module is at high level, Split-1, whose DNL modules are at middle level, achieves better maxF, MAE and S-m. It suggests that placing the proposed module at middle level better improves the performance.

The baseline obtains the fastest CPU inference and the lowest MAdds. As the splits number decreases, both CPU time cost and MAdds of the corresponding DNL network become larger. Because smaller splits number results in computing more pairwise similarities. Since CPU time is measured for a whole network, we need to compute *difference* between some network and the baseline to obtain the time cost of DNL modules. For examples, DNL modules in Split-9 additionally costs $399.3 - 390.6 \sim 9$ ms and $9.567 - 9.325 \sim 0.24$ B MAdds while DNL modules in Split-1 additionally takes

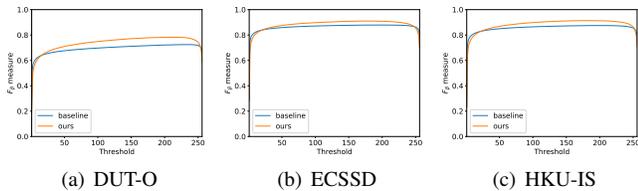


Fig. 9. Ablation Study on Threshold- F_{β} measure Curves. As shown in the above, the F_{β} measures of our proposed method are considerably higher than those of the baseline, for most thresholds within the range of [50, 200].

TABLE IV
THE EFFECTIVENESS OF DNL MODULE.

	HKU-IS			CPUTime /secs	MAdds
	maxF	MAE	S-m		
Baseline w/o DNL	0.8745	0.0572	0.8563	0.3906	9.325B
Split-9	0.9133	0.0451	0.8974	0.3993	9.567B
Split-5	0.9133	0.0451	0.8969	0.4019	9.657B
Split-3	0.9131	0.0452	0.8967	0.4103	9.781B
Split-1	0.9121	0.0458	0.8952	0.4549	10.42B
IR6-split10	0.9125	0.0459	0.8922	0.5138	12.25B
IR6-split5	0.9069	0.0479	0.8873	0.5447	12.90B
IR6-split1	0.9036	0.0489	0.8851	1.1384	18.13B

454.9 – 390.6 ~ 64 ms and 10.42 – 9.325 ~ 1.1B MAdds when compared with the baseline model. Compared with Split-1, DNL modules in Split-9 speed up 7× CPU time from 64 ms to 9 ms, and reduce 5× MAdds from 1.1B to 0.24B. For IR6-splits, the difference is larger. The DNL module in IR6-split10 reduces 1.1384 – 0.5138 ~ 0.6 s CPU time and 18.13 – 12.25 ~ 6B MAdds, in comparison to IR6-split1. The above results indicate that the divide-and-conquer variant of DNL module indeed accelerates the inference. To understand the difference between positioning DNL modules at middle level and high level, we compare Split-1 with IR6-split1. DNL modules are located at the middle level of Split-1 and the high level of IR6-split1. IR6-split1 spends more CPU time (0.7 s) and larger MAdds (7.7B) than Split-1. Thus positioning DNL modules at high level causes more computational cost. Because the time complexity of DNL modules is in proportion to squared number of input channels.

V. CONCLUSIONS

This paper introduces a novel DNL module that effectively enhances the accuracy of an inverted residual block. A divide-and-conquer variant of DNL is proposed to further accelerate inference. Moreover, we develop a light-weight DNL based network architecture with low memory cost, high inference speed and competitive accuracy. Numeric results declare that our method achieves not only competitive accuracy but also state-of-the-art efficiency among deep CNN based methods.

REFERENCES

[1] R. Zhao, W. Ouyang, and X. Wang, “Unsupervised salience learning for person re-identification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3586–3593.

[2] A. P. Shon, D. B. Grimes, C. L. Baker, M. W. Hoffman, S. Zhou, and R. P. Rao, “Probabilistic gaze imitation and saliency learning in a robotic head,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005, pp. 2865–2870.

[3] V. Navalpakkam and L. Itti, “An integrated model of top-down and bottom-up attention for optimal object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 2049–2056.

[4] H. Wu, G. Li, and X. Luo, “Weighted attentional blocks for probabilistic object tracking,” *The Visual Computer*, vol. 30, no. 2, pp. 229–243, 2014.

[5] S. Avidan and A. Shamir, “Seam carving for content-aware image resizing,” in *ACM Transaction on Graphics*, vol. 26, no. 3, 2007, p. 10.

[6] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[7] P. Zhang, D. Wang, H. Lu, H. Wang, and B. Yin, “Learning uncertain convolutional features for accurate saliency detection,” in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2017, pp. 212–221.

[8] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang, “Progressive attention guided recurrent network for salient object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 714–722.

[9] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, “Global contrast based salient region detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 569–582, 2015.

[10] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, “Saliency filters: Contrast based filtering for salient region detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 733–740.

[11] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, “Salient object detection: A discriminative regional feature integration approach,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2083–2090.

[12] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech, “Minimum barrier salient object detection at 80 fps,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1404–1412.

[13] W.-C. Tu, S. He, Q. Yang, and S.-Y. Chien, “Real-time salient object detection with a minimum spanning tree,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2334–2342.

[14] X. Huang and Y.-J. Zhang, “300-fps salient object detection via minimum directional contrast,” *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4243–4254, 2017.

[15] Q. Peng, Y. Cheung, X. You, and Y. Y. Tang, “A hybrid of local and global saliencies for detecting image salient region and appearance,” *IEEE Transactions on Systems Man and Cybernetics: Systems*, vol. 47, no. 1, pp. 86–97, 2017.

[16] L. Zhang, C. Yang, H. Lu, X. Ruan, and M.-H. Yang, “Ranking saliency,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1892–1904, 2017.

[17] G. Li and Y. Yu, “Visual saliency based on multiscale deep features,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5455–5463.

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

[19] G. Li and Y. Yu, “Contrast-oriented deep neural networks for salient object detection,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6038–6051, 2018.

[20] 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 Transactions on Image Processing*, vol. 25, no. 8, pp. 3919–3930, 2016.

[21] N. Liu and J. Han, “Dhsnet: Deep hierarchical saliency network for salient object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 678–686.

[22] Z. Luo, A. Mishra, A. Achkar, J. Eichel, S. Li, and P.-M. Jodoin, “Non-local deep features for salient object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6609–6617.

[23] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, “Salient object detection with recurrent fully convolutional networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

- [24] M. Sun, Z. Zhou, Q. Hu, Z. Wang, and J. Jiang, "Sg-fcn: A motion and memory-based deep learning model for video saliency detection," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 2900–2911, 2018.
- [25] H. Li, G. Li, and Y. Yu, "Rosa: robust salient object detection against adversarial attacks," *IEEE transactions on cybernetics*, 2019.
- [26] Y. Qin, M. Feng, H. Lu, and G. W. Cottrell, "Hierarchical cellular automata for visual saliency," *International Journal of Computer Vision*, vol. 126, no. 7, pp. 751–770, 2018.
- [27] S. Chen, X. Tan, B. Wang, and X. Hu, "Reverse attention for salient object detection," in *European Conference on Computer Vision*, 2018.
- [28] G. Li, Y. Xie, T. Wei, K. Wang, and L. Lin, "Flow guided recurrent neural encoder for video salient object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [29] H. Li, G. Chen, G. Li, and Y. Yu, "Motion guided attention for video salient object detection," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [30] Y. Zeng, M. Feng, H. Lu, G. Yang, and A. Borji, "An unsupervised game-theoretic approach to saliency detection," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4545–4554, 2018.
- [31] L. Zhang, J. Ai, B. Jiang, H. Lu, and X. Li, "Saliency detection via absorbing markov chain with learnt transition probability," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 987–998, 2018.
- [32] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr, "Deeply supervised salient object detection with short connections," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 4, pp. 815–828, 2019.
- [33] T. Wang, L. Zhang, S. Wang, H. Lu, G. Yang, X. Ruan, and A. Borji, "Detect globally, refine locally: A novel approach to saliency detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3127–3135.
- [34] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, and J. Jiang, "A simple pooling-based design for real-time salient object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.
- [35] J. Ngiam, Z. Chen, D. Chia, P. W. Koh, Q. V. Le, and A. Y. Ng, "Tiled convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2010, pp. 1279–1287.
- [36] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems*, 2015, pp. 1135–1143.
- [37] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, "Sparse convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 806–814.
- [38] V. Sindhwani, T. Sainath, and S. Kumar, "Structured transforms for small-footprint deep learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 3088–3096.
- [39] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 525–542.
- [40] X. Zhang, J. Zou, K. He, and J. Sun, "Accelerating very deep convolutional networks for classification and detection," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 1943–1955, 2016.
- [41] M. Wang, B. Liu, and H. Foroosh, "Factorized convolutional neural networks," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 545–553.
- [42] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, no. 6, 2017.
- [43] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.
- [44] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende *et al.*, "Interaction networks for learning about objects, relations and physics," in *Advances in Neural Information Processing Systems*, 2016, pp. 4502–4510.
- [45] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 60–65.
- [46] X. He, S. Yang, G. Li, H. Li, H. Chang, and Y. Yu, "Non-local context encoder: Robust biomedical image segmentation against adversarial attacks," in *AAAI Conference on Artificial Intelligence (AAAI)*, January 2019.
- [47] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the IEEE International Conference on Computer Vision*, 1999, p. 1033.
- [48] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in Neural Information Processing Systems*, 2011, pp. 109–117.
- [49] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4967–4976.
- [50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.
- [51] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7794–7803.
- [52] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti, "Visual interaction networks: Learning a physics simulator from video," in *Advances in Neural Information Processing Systems*, 2017, pp. 4539–4547.
- [53] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.
- [54] G. Li, X. He, W. Zhang, H. Chang, L. Dong, and L. Lin, "Non-locally enhanced encoder-decoder network for single image de-raining," in *ACM Multimedia Conference on Multimedia Conference*. ACM, 2018, pp. 1056–1064.
- [55] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 801–818.
- [56] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3166–3173.
- [57] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1155–1162.
- [58] G. Li and Y. Yu, "Visual saliency detection based on multiscale deep cnn features," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5012–5024, 2016.
- [59] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 280–287.
- [60] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan, "Learning to detect salient objects with image-level supervision," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3796–3805.
- [61] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1597–1604.
- [62] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4558–4567.
- [63] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 478–487.
- [64] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan, "Saliency detection with recurrent fully convolutional networks," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 825–841.