IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 2, 2019, accepted March 19, 2019, date of publication April 1, 2019, date of current version April 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2908685

Dense Convolutional Networks for Semantic Segmentation

CHAOYI HAN, YIPING DUAN, XIAOMING TAO, (Member, IEEE), AND JIANHUA LU, (Fellow, IEEE)

Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China
Beijing Innovation Center for Future Chip, Beijing 100084, China

Corresponding author: Xiaoming Tao (taoxm@mail.tsinghua.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61622110 and Grant 61801260, and in part by the Chinese Postdoctoral Science Foundation under Grant 2017M620044 and Grant 2018T110098.

ABSTRACT Recent studies have greatly promoted the development of semantic segmentation. Most state-of-the-art methods adopt fully convolutional networks (FCNs) to accomplish this task, in which the fully connected layer is replaced with the convolution layer for dense prediction. However, standard convolution has limited ability in maintaining continuity between predicted labels as well as forcing local smooth. In this paper, we propose the dense convolution unit (DCU), which is more suitable for pixel-wise classification. The DCU adopts dense prediction instead of the center-prediction manner used in current convolution layers. The semantic label for every pixel is inferred from those overlapped center/off-center predictions from the perspective of probability. It helps to aggregate contexts and embeds connections between predictions, thus successfully generating accurate segmentation maps. DCU serves as the classification layer and is a better option than standard convolution in FCNs. This technique is applicable and beneficial to FCN-based state-of-the-art methods and works well in generating segmentation results. Ablation experiments on benchmark datasets validate the effectiveness and generalization ability of the proposed approach in semantic segmentation tasks.

INDEX TERMS Dense convolution unit, fully convolutional network, overlapped prediction, semantic segmentation.

I. INTRODUCTION

Semantic segmentation serves as an indispensable part in image and video content analysis. It aims to assign each pixel a predefined semantic tag and has long been a very challenging problem due to the high intra-class variability of data. Recently, many promising results have been reported regarding this dense prediction task, mainly benefiting from the advances in deep convolutional neural networks (DCNNs) [1]–[8]. Most state-of-the-art methods [9]–[13] consider semantic segmentation as a pixel-wise classification problem and adopt fully convolutional networks (FCNs) [14] to accomplish it. Remarkable performances have been achieved on various benchmark datasets [15]–[23].

Initialized with the parameters of DCNNs pre-trained on large-scale datasets such as ImageNet [24], FCN performs relatively well on semantic segmentation tasks with only an additional convolution layer serving as the classifier [14]. The superior ability of DCNNs in extracting high-level semantics determines that even such a straightforwardly assembled end-to-end network can be more effective than specially designed methods with hand-crafted features. Therefore, most following works inherit the fully convolution spirit and turn to discovering more suitable and effective structures for dense prediction networks. Among those studies, dilated convolution and deconvolution have become very popular techniques due to their excellent performances in generating high-resolution feature maps. More powerful baseline networks such as ResNet [2] further promotes the development of the semantic segmentation community. Currently, a typical FCN-based framework can be seen in the Deeplab family [10], [25].

Nevertheless, two inharmonious problems exist between FCNs and the semantic segmentation task. First, the spatial

The associate editor coordinating the review of this manuscript and approving it for publication was Michele Magno.

context relationships between pixels are not explicitly depicted in FCNs. Starting from deep classification networks, FCNs replace the last few fully connected layers with convolution layers and optimize the whole network with a pixel-wise loss function. Consequently, the label-prediction process for one pixel is largely independent from others. This unavoidably brings about many discontinuities and noises in the segmentation maps, as shown in Fig.1(a). To eliminate such discontinuities and noises, many approaches have been developed. Conditional random fields (CRFs) are widely adopted models in traditional methods. CRFs can, to some extent, refine the segmentation map, but it is inconvenient for them to be end-to-end trained with FCNs [26], [27]. Another method that is shown to be crucial and rather effective is incorporating more global context into the final classification layer [11].

The other problem is related to the rigid computing mode of standard 2D convolution. In current FCNs, convolution is performed over a pre-defined regular grid. Within the same layer, the receptive fields for all convolutions are actually the same. This mechanism deduces that activations are generated through an identical, default and unchangeable manner. Although the stack of convolutional layers exhibits a powerful representation ability, the increases in depth cannot fully offset this drawback. As pointed out in [28], standard convolution cannot adapt to the semantic boundaries. This limits the performances of convolutional neural networks (CNNs). In semantic segmentation networks, a fair number of predictions are performed over semantically vague areas, as illustrated in Fig. 1(b).

In summary, the key problem lies in the classifier of the FCN, which is currently performed by a single convolution layer. In the classification network, a fully connected layer or global pooling layer [2], [29] helps to enable adequate context embedding over whole feature maps, which is vital for accurate prediction. However, their counterpart in the FCN lacks such a global view. The fully convolution fashion determines that only part of the input map is involved in the label-prediction processes for pixels. We observed that the probability map generated by a typical FCN only exhibits a high confidence level on pixels near the centre of objects (''right'' areas). Apparently, many predictions are not made over their semantically corresponding areas. Decisions on pixels which are between objects are as random as a dice roll.

To solve the above problem, we propose the ''dense convolution'' scheme and develop the corresponding dense convolution unit (DCU) as a better classifier to current networks. As the name suggests, dense convolution produces a dense output for a single receptive field. It enables off-center prediction that is unavailable in the standard convolution. The final semantic label for every pixel is inferred from those overlapped center/off-center predictions. Similar ''dense'' ideas can be seen in [30]–[32]. With this mechanism, the ''right'' receptive field decides not only the semantic label of its center pixel but also the neighbors. Thus, pixels offset from the right positions can have the chance to further refine their labels. The contributions of this paper include the following:

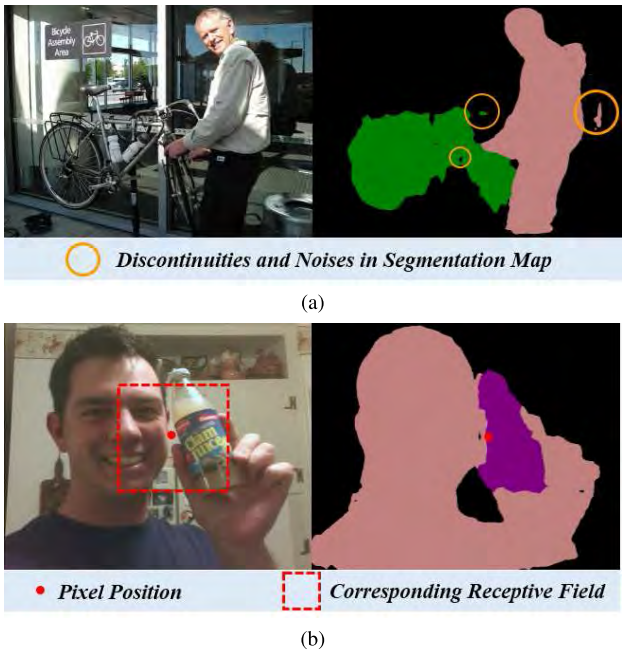

FIGURE 1. In FCN, the prediction for each pixel is relatively independent. Therefore, many discontinuities and "noises" exist in segmentation map. Moreover, due to the rigid receptive field of standard 2D convolution, lots of predictions are unreliable. (a) Discontinuities and noises. (b) Semantically vague area.

- We give a comprehensive explanation about the motivation and mentality of the dense convolutional network and present the network structures in detail.
- The complete formulation and thorough algorithm about training dense convolutional networks are developed. The theoretical analysis regarding the design of the weight map in DCU and corresponding visual illustration are also included.
- We conduct extensive experiments on benchmark datasets and report corresponding results. The ablation study and experiments demonstrate its superior performance over standard convolution and other competitors.

An early conference version of our work can be seen in [33]. In this paper, the relationships and differences to existing works are added to show the causes and effects. Besides the original ablation study on baseline model, we generalize the DCU to another well-known network as well as additional datasets and achieve significant improvements. In the following, we will first give a review to related work in Section II. Then, we present the technical details in Section III and show that dense convolution can be efficiently implemented and works well on semantic segmentation tasks. Experimental results and corresponding discussions are given in Section IV. The conclusion and future work are drawn in Section V.

II. RELATED WORK

Recent improvements on semantic segmentation can be largely attributed to the application of fully convolutional networks (FCNs) [14]. The typical convolutional neural networks (CNNs) used for classification tasks are transformed into dense prediction networks in a straightforward way. Such networks show amazing ability on semantic segmentation tasks, thus have becoming the paradigm in state-of-the-art methods. Meanwhile, many works have focused on the intrinsic drawbacks of the fully convolutional structure and developed many variants, which are summarized as follows.

A. RESOLUTION RECOVERY

Deep networks usually adopt consecutive pooling layers to embed more context, as well as to reduce the high computation requirements, which has been proven to be essential in image processing [1]–[3]. However, FCN-based methods suffer from such downsamplings a lot due to the loss in image resolution. To alleviate such loss, dilated convolution is proposed and has been widely adopted [34]. It can enlarge the receptive fields by inserting holes (zeros) to filters. By removing pooling layers and increasing the dilation rate of subsequent convolutions, feature resolution can be maintained without any change to the structure and parameter size of the model. Deconvolution [35]–[37] is another popular method to recover resolution. It can be considered as the inverse of convolution. Deconvolution realizes upsampling using learnable weights instead of nonparametric methods (e.g., bilinear interpolation) and shows superior performance. A similar idea can found in an encoder-decoder network [38].

B. MULTISCALE CONTEXT EMBEDDING

Unlike typical classification networks whose inputs are set to a fixed size, dense prediction networks receive images of arbitrary sizes. Unfortunately, mutative input limits the accuracy of segmentation because the neural networks cannot adapt to different scales effectively. Most existing works choose to embed multiscale context when dealing with this problem [39], [40]. Among those methods, multiscale testing is straightforward and easy to conduct. This method resizes input image to different scales then applies trained networks (the same or different) to them separately. Final result is obtained from the combination of segmentation maps in several scales. The feature pyramid structure [10] merges such multiscale embedding operation into networks in an end-to-end way. Spatial pyramid pooling (SPP) [11] shares a similar idea and obtains a significant margin compared to existing methods. The recent proposed Scale Normalization for Image Pyramids (SNIP) [41] was shown to better deal with objects with small or large scales, but it has not been applied in semantic segmentation tasks yet.

C. BOUNDARY REFINEMENT

Poor segmentation performance along object boundaries might be the common problem for all deep learning based methods. As the opposite of global context, local information is not well captured by DNNs. Though increasing network capacity (depth) leads to constantly better results [10], corresponding expenses in runtime and memory would soon be unbearable. Therefore, in many state-of-the-art works, CRF is adopted as post-processing due to its excellent performance along object boundaries. The end-to-end training of CRF with DNNs also has been studied in many works [26], [27], but specific modeling and inference methods can hardly be extended to other networks. More popular ways tend to add boundary refinement units into FCNs [12], [42]. Usually, a small kernel size is adopted for better characterization of local context. However, since the boundary accounts for a small proportion of the whole image, those methods bring limited improvements compared to techniques such as dilated convolution, SPP and so on.

D. STRUCTURE EVOLUTION

Recent studies rethink the DCNN structure for embedding high-level semantics. Deformable convnet [28] augments those fixed sampling locations in typical networks with extra parameters. As a data-driven method, it can adapt to the scale and shape of an object, which is unachievable in standard convolution. Dense upsampling convolution [32] divides the whole output map into several equal subparts and generates each part simultaneously using different channels. A similar idea can be found in FCIS [30] used in instance segmentation, where different channels encode position-sensitive features and then are assembled to obtain the final segmentation map. Methods of this kind exploit several specific structures for dense prediction networks motivated by various observations or assumptions. Their successes indicate that, though classification task transfers efficiently to semantic segmentation, task specific design is still indispensable for further breakthroughs.

We would mention that most above methods were tested on several benchmark datasets for fair comparison. However, fully annotated segmentations in existing datasets are relatively rare, so training DNNs from scratch is difficult. State-of-the-art performances are achieved through transfer learning based on classification networks. Therefore, methods adopting weakly supervised and semi-supervised learning [43]–[47] have a very promising future since they can utilize data with weak or even no annotations. In that case, specially designed loss functions and recursive training are widely adopted. More details can be found in references.

This paper is close to those works on structure evolution in terms of motivation. We argue that classifications should be made over semantically corresponding areas so high confidence level can be maintained. However, in contrast to deformable convnet where each activation attempts to find its semantic area, dense convolution lets semantic areas generate more decisions and spreads the results through the pixel-wise weight map. The extra channels in our network are not meant to encode more features for the same content, which is more like over-fitting, but established to encode different contents.

This paper tries to demonstrate that not all receptive fields are of the same importance when predicting the semantic labels and offset prediction of the "center area" is more credible than center prediction of the "offset area".

III. PROPOSED APPROACH

The convolutional neural network paves the way for the successful application of artificial neural networks to computer vision. Convolution (sometimes abbreviated as conv) takes advantage of the local properties inside images and is widely used in digital image processing. A standard convolution layer performs the following process (feature maps and vectors are shown in boldface):

$$\boldsymbol{y}_{h,w}^{(o)} = \sum_{i} \sum_{m} \sum_{n} \boldsymbol{\theta}_{m,n}^{(i,o)} \cdot \boldsymbol{x}_{h+m,w+n}^{(i)} + b^{(o)} \tag{1}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ denote the input and output feature maps, $\boldsymbol{\theta}$ denotes the weights of conv kernels, i, o represent different channels and h, w, m, n represent the index in corresponding feature maps and conv kernels. A simple end-to-end trainable network for semantic segmentation is then completed once the loss layer is applied after. The softmax loss function for the above network is defined as:

$$L_{softmax} = -\frac{1}{N} \sum_{h} \sum_{w} log(\exp(\boldsymbol{y}_{h,w}^{label}) / \sum_{o} \exp(\boldsymbol{y}_{h,w}^{(o)})) \tag{2}$$

where *label* denotes the ground truth label in position (h, w), N is the number of elements in segmentation map and equals to $h \cdot w$.

A. DENSE PREDICTION INSTEAD OF CENTRE PREDICTION

From the perspective of signal processing, the last convolution layer in FCN gives the probability of class o on position (h, w): $P_{h,w}^{(o)} \sim \boldsymbol{y}_{h,w}^{(o)}$. Analogous to forward prediction and backward prediction, which are common in signal processing, we can call the prediction manner standard convolution adopts as center prediction. Intuitively, the input-output correspondence for dense prediction networks should not be limited in such a fixed manner. In fact, the correspondence between one pixel and its "perfect" receptive field varies significantly when it lies in different positions of an object. To further exploit the ability, we extend the standard convolution and deduce the general form as the dense convolution.

For a clear presentation, in this section we use the similar notation to [28], which expresses the 2D location (h, w) in vector form and neglect dilation and stride for simplicity. Then the standard convolution on a single input map $\boldsymbol{x}$ can be reformulated as follows:

$$\boldsymbol{y}(\boldsymbol{p}_0) = \sum_{i=1}^{k^2} \boldsymbol{\theta}(\boldsymbol{p}_i) \cdot \boldsymbol{x}(\boldsymbol{p}_0 + \boldsymbol{p}_i) \tag{3}$$

where $\boldsymbol{p}_0$ denotes the location in $\boldsymbol{x}$ and $\boldsymbol{p}_i$ represents the ith item of the following:

$$\{(-\frac{k}{2}, -\frac{k}{2}), (-\frac{k}{2}, -\frac{k}{2}+1), \ldots, (\frac{k}{2}, \frac{k}{2}-1), (\frac{k}{2}, \frac{k}{2})\}$$

which enumerates all possible locations within the $k \times k$ convolution kernel. In common practice, k is set to an odd number and $k/2$ is rounded down to an integer. Usually padding will be adopted in case of dimension mismatching.

In every convolution process, k^2 inputs are sampled and convolved with shared learnable parameters. Every equally sampled $k \times k$ grid in $\boldsymbol{x}$ is responsible for one activation in $\boldsymbol{y}$ that located at the center of it. Apparently, the number of input and output in a standard convolution is k^2 versus 1. The proposed dense convolution introduces extra outputs into the above process. For a $k \times k$ grid $\boldsymbol{R}$ in $\boldsymbol{x}$, it computes k^2 activations (also a grid):

$$y_j = \sum_{i=1}^{k^2} \boldsymbol{\theta}_j(\boldsymbol{p}_i) \cdot \boldsymbol{R}(\boldsymbol{p}_i), \quad j = 1, 2, \ldots, k^2 \tag{4}$$

where $\boldsymbol{\theta}_j$ denotes the weights of the jth conv kernels. If conducting such convolutions on the whole input map $\boldsymbol{x}$ and assemble the jth activations, we get k^2 output maps:

$$\widetilde{\boldsymbol{y}}_j(\boldsymbol{p}_0) = \sum_{i=1}^{k^2} \boldsymbol{\theta}_j(\boldsymbol{p}_i) \cdot \boldsymbol{x}(\boldsymbol{p}_0 + \boldsymbol{p}_i + \boldsymbol{p}_j) \tag{5}$$

where $\boldsymbol{p}_j$ has the same meaning as $\boldsymbol{p}_i$ and encodes the spatial shift of $\widetilde{\boldsymbol{y}}_j$ to $\boldsymbol{y}$. The maps in $\widetilde{\boldsymbol{y}}_j$ are of the same size but not spatially aligned since they contain different elements of the output grid.

It is not necessary to increase the output channels of existing convolution layers k^2 times to accomplish such a dense prediction. Deep networks are always wide; therefore, all we need do is to divide the channels into groups and maintain k^2 channels within each group. In each group, the k^2 channels can be viewed as the dense prediction of one feature map. To enable this, we reorganize the relative position of those channels, as illustrated in the "Spatial Shift within Group" part of Fig. 2. In this way, we change the center prediction manner of the standard convolution while keeping similar parameters to existing models. Such division prevents the fast growth of parameter size as k increases.

Upon doing this, the dense convolution brings a noticeable problem called spatial overlap. This problem emerges when the output maps overlap with each other. In fact, such overlaps can be avoided by increasing the convolution stride to k. However, in those circumstances, the above process becomes block-to-block convolution which shows obvious artificial partition. Therefore, we keep the stride unchanged, which is set to 1 in a typical classification layer, and find a strategy to deal with the overlapped maps and generate the final outputs.

B. A COMBINATION STRATEGY: FROM THE PERSPECTIVE OF PROBABILITY

As pointed out above, dense convolution is developed as the extension to standard convolution. Note that, in the original convolution layer, the activation indicates the probability for each class o on position (h, w): $P_{h,w}^{(o)} \sim \boldsymbol{y}_{h,w}^{(o)}$. Similarly, the multioutputs of dense convolution are also expected to

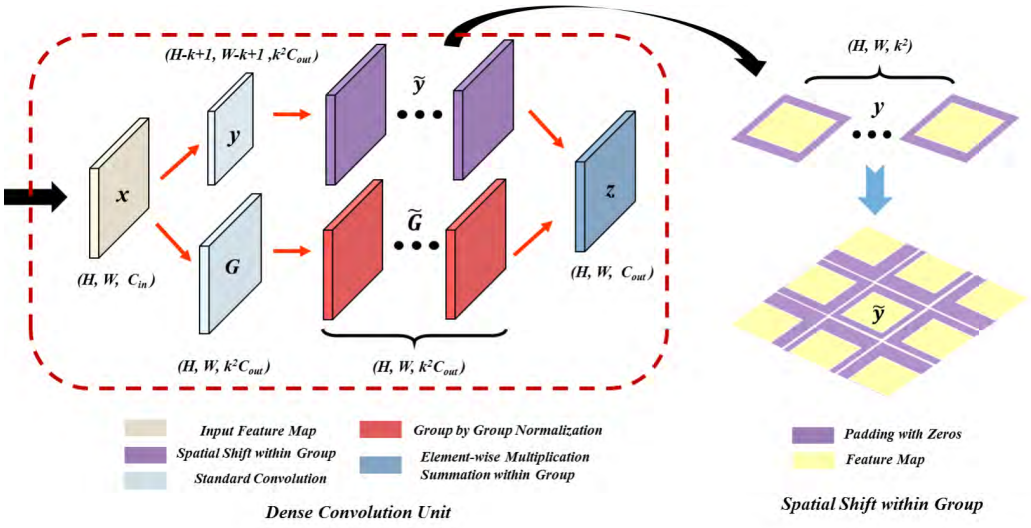

FIGURE 2. Structure of the dense convolution unit (DCU) when k = 3. All operations are performed group by group. Each group has k^2 maps and corresponds to one output channel. The spatial shift operation adds shifts and pads zeros to the feature maps. The triples denotes the height, width and channel of corresponding feature map. For better illustration, the feature maps of $\tilde{y}$ are spatially arranged to show the differences in paddings. Best viewed in color.

represent some kind of probability. Since those overlapped activations predict the same probability but are performed on different sampling grids (receptive fields), it is reasonable to view the overlapped activations as probabilities conditioned on different inputs. In this respect, the final result $P^{(o)}_{h,w}$ should be computed following the total probability formula:

$$P^{(o)}_{h,w} = \sum_j G^{(o)}_{h,w,j} \cdot P^{(o)}_{h,w|j}. \tag{6}$$

Here, the conditional probabilities $P^{(o)}_{h,w|j}$ are represented by those overlapped maps $\tilde{\boldsymbol{y}}_j(h, w)$ and the probabilities $G^{(o)}_{h,w,j}$ act as the normalized coefficients. Considering that different locations may encode different semantics and thus have different dependencies on those overlapped predictions $\tilde{\boldsymbol{y}}_j(h, w)$, a coefficient map can be used instead of sharing the coefficients across all positions. The coefficient maps $\boldsymbol{G}_j$ is generated through a convolution layer upon the input map $\boldsymbol{x}$.

Equation (6) determines that $G^{(o)}_{h,w,j}$ should always adds up to one for a single position (h, w). Therefore, we apply the softmax function on those maps (called weight maps in this paper) to satisfy this constraint. Softmax normalization is used because it has a good numerical stability and is easy to differentiate, i.e., $\boldsymbol{G}_j$ is normalized using the following:

$$\widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0) = \exp(\boldsymbol{G}_j(\boldsymbol{p}_0))/\sum_{m=1}^{k^2}\exp(\boldsymbol{G}_m(\boldsymbol{p}_0)), \quad j = 1, \ldots, k^2. \tag{7}$$

Therefore, in the proposed dense convolution scheme, weight summation is adopted as the combination strategy for overlapped feature maps. This is a simple yet efficient method. Finally, the output map $\boldsymbol{z}$ can be obtained via a weighted summation as follows:

$$\boldsymbol{z}(\boldsymbol{p}_0) = \sum_{j=1}^{k^2}\widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0)\cdot\sum_{i=1}^{k^2}\boldsymbol{\theta}_j(\boldsymbol{p}_i)\cdot\boldsymbol{x}(\boldsymbol{p}_0+\boldsymbol{p}_i+\boldsymbol{p}_j). \tag{8}$$

If the dilation rate is not 1, just replace the sample grid $\boldsymbol{R}$ in (4) with corresponding grid and the process can still be finished. When the stride $s > 1$, the convolution can be transformed into dilated convolution with stride 1, which maintains the feature resolution; Thus, this paper leaves such situations behind.

Under the most complete setting, the normalized coefficients (weight maps) are not shared by different positions nor by different output channels. In practice, certain compromises can be made in consideration of complexity. Fig. 3 illustrates how dense convolution works on one input image and visualizes the corresponding feature maps. After the previous feature extraction process, dense convolution first generate multiple middle predictions. Then, the spatial shift unit helps to correct their relative coordinates and relocates them properly. Meanwhile, normalized position-sensitive weight maps are generated through another standard conv layer and help to combine those multiple middle predictions. The "right" channel achieves the strongest activation and thus suppresses other channels after the softmax operation. Finally, the segmentation map is synthesized using argmax function.

C. TRAINING DENSE CONVOLUTIONAL NETWORK

To integrate dense convolution into current framework, the dense convolution unit (DCU) is developed. The overall structure is shown in Fig. 2. It includes two standard 2D convolutions, one group by group normalization, one spatial shift unit and one element-wise multiplication followed by summation.

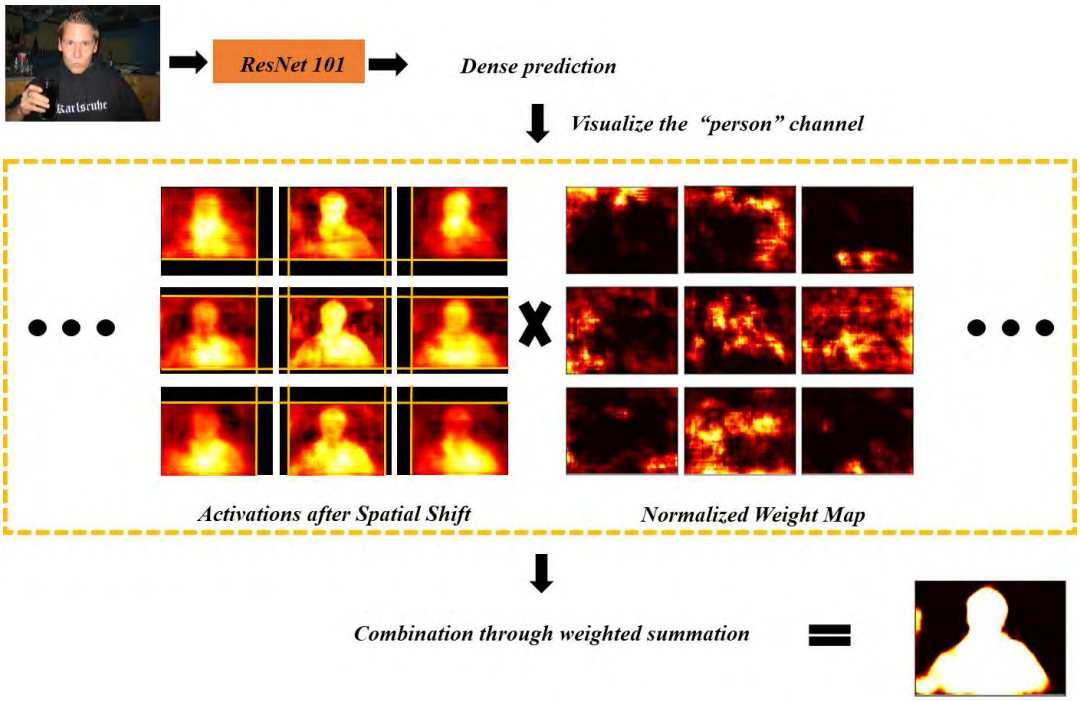

FIGURE 3. Visualization of the middle predictions and corresponding weight maps in DCU. The overlapped part is combined through element-wise multiplication and summation (kernel size is 3 × 3). Best viewed in color.

Based on existing deep learning frameworks, only a small modification is needed. The key point lies in the spatial shift operation. This part implements $\widetilde{\boldsymbol{y}}_j$ following (4) and (5). Suppose that input feature maps have C_{in} channels, output feature maps have C_{out} channels and the convolution kernel size is $k \times k$. To achieve dense convolution, the DCU first generates $k^2 C_{out}$ middle maps $\boldsymbol{y}_j$ using the standard convolution then add offsets to them through spatial shift layer and get $\widetilde{\boldsymbol{y}}_j$. Notice that in Section III-A, dense convolution is described on a single input layer. The multichannel formulation can be easily deduced as follows:

$$\boldsymbol{z}^{(C_o)}(\boldsymbol{p}_0) = \sum_{j=1}^{k^2} \widetilde{\boldsymbol{G}}_j^{(C_o)}(\boldsymbol{p}_0) \cdot \left(\sum_{C_{ip}} \sum_{i=1}^{k^2} \boldsymbol{\theta}_j^{(C_i, C_o)}(\boldsymbol{p}_i) \cdot \boldsymbol{x}^{(C_i)}(\boldsymbol{p}_0 + \boldsymbol{p}_i + \boldsymbol{p}_j) + b^{(C_o)}\right) \quad (9)$$

where C_o and C_i denotes different output and input channels. Now that the output has C_{out} channels, $\boldsymbol{y}$ and $\widetilde{\boldsymbol{y}}$ are divided into C_{out} groups with each group having k^2 maps, corresponding to different output channels. Then, the spatial shift operation is performed group by group. The convolution uses no padding since spatial shift will recover the reduced dimension. That means $\widetilde{\boldsymbol{y}}$ has the same dimension with input $\boldsymbol{x}$, yet $\boldsymbol{y}$ is smaller. Similarly, $k^2 C_{out}$ normalized weight maps $\widetilde{\boldsymbol{G}}$ can be obtained with another standard convolution followed by one group-by-group normalization using (7). In contrast, padding is used this time.

In the forward propagation process, output maps can be computed following (9). For backward propagation, the differentiation formulas should be established. To keep consistent with Section III-A, the following discussion is within a single group and the superscript C_{out} is omitted. This paper only presents the derivatives for the spatial shift operation. The rest can be completed with back propagation of the standard convolution and softmax layer.

Given $\partial Loss/\partial \boldsymbol{z}$ for output map $\boldsymbol{z}$, the gradient w.r.t. $\boldsymbol{y}_j$ and $\widetilde{\boldsymbol{G}}_j$ can be computed as

$$\frac{\partial Loss}{\partial \boldsymbol{y}_j(\boldsymbol{p}_0)} = \frac{\partial Loss}{\partial \boldsymbol{z}(\boldsymbol{p}_0 - \boldsymbol{p}_j)} \cdot \widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0 - \boldsymbol{p}_j) \quad (10)$$

and

$$\frac{\partial Loss}{\partial \widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0)} = \frac{\partial Loss}{\partial \boldsymbol{z}(\boldsymbol{p}_0)} \cdot \boldsymbol{y}_j(\boldsymbol{p}_0 + \boldsymbol{p}_j). \quad (11)$$

Here, the correspondence $\boldsymbol{y}_j(\boldsymbol{p}_0) = \widetilde{\boldsymbol{y}}_j(\boldsymbol{p}_0 - \boldsymbol{p}_j)$ is used. In the above scheme, each output channel has its own weight maps because they are expected to encode different semantics. Thus, the back propagations for $\boldsymbol{y}_j$ and $\widetilde{\boldsymbol{G}}_j$ in different groups are independent from each other. However, during the experiments, we found that if weight maps $\widetilde{\boldsymbol{G}}$ are shared across the output channels, the network still works well and only minor accuracy loss is observed. In that case, the gradient $\partial Loss/\partial \widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0)$ should sum up over all C_{out} channels and the formula becomes:

$$\frac{\partial Loss}{\partial \widetilde{\boldsymbol{G}}_j(\boldsymbol{p}_0)} = \sum_{C_o} \frac{\partial Loss}{\partial \boldsymbol{z}^{(C_o)}(\boldsymbol{p}_0)} \cdot \boldsymbol{y}_j^{(C_o)}(\boldsymbol{p}_0 + \boldsymbol{p}_j). \quad (12)$$

Algorithm 1 End-to-End Training of Dense Convolutional Network

Initialize: Set up convolution layers and set the kernel size of DCU to $k \times k$. Network parameters are denoted as Φ. Maximum number of iterations: max_iter.

while $iter < max\_iter$ **do**
  Forward:
  Generate $\{\boldsymbol{y}_j^{C_o}, \boldsymbol{G}_j^{C_o}\}_{j=1...k^2}^{C_o=1...C_{out}}$ with previous layers.
  for $C_o = 1$ to C_{out} **do**
    Add spatial shift to $\boldsymbol{y}_j$ using correspondence $\widetilde{\boldsymbol{y}}_j^{C_o}(\boldsymbol{p}_0) = \boldsymbol{y}_j^{C_o}(\boldsymbol{p}_0 + \boldsymbol{p}_j)$
    Normalize $\boldsymbol{G}_j^{C_o}$ using (7) $\Rightarrow \widetilde{\boldsymbol{G}}_j^{C_o}$.
    Output $\boldsymbol{z}^{C_o}$ using (9).
  end for
  Compute softmax loss through (2) then get $\partial Loss/\partial \boldsymbol{z}^{C_o}$.
  Backward:
  for $C_o = 1$ to C_{out} **do**
    Apply (10) $\Rightarrow \frac{\partial Loss}{\partial \widetilde{\boldsymbol{G}}_j^{C_o}}$.
    Apply (11) $\Rightarrow \frac{\partial Loss}{\partial \boldsymbol{y}_j^{C_o}}$.
  end for
  Back propagation to previous layers.
  Update Φ.
  $iter \Leftarrow iter + 1$
end while

Output: Trained network for inference.

The overall procedure for training a dense convolutional network is summarized in Algorithm 1.

IV. EXPERIMENTS

We conduct extensive experiments on the benchmark datasets PASCAL VOC 2012 [48] and Cityscapes [49] to demonstrate the effectiveness of the proposed dense convolution scheme. To begin with, a frequently used baseline model (with a single convolutional layer as classifier) is established following common practice. Then, several settings important for final accuracy are given and discussed. After that, the dense convolution is applied to the baseline model to exhibit its superior performance compared to the standard convolution. However, since the proposed method introduces more parameters (two middle convolutional layers) into the current network, it is hard to tell whether the improvement comes from the specially designed structure or the extra parameters. To further observe its capacity under similar or even fewer parameters than the standard convolution, the dense convolution is integrated into the well-known method PSPNet [11]. The parameters in the standard convolution and dense convolution are carefully controlled, and the corresponding results are represented. In addition, running time and complexity are also reported for comprehensive comparison. We make the code available at `https://github.com/hancy16/DCU`.

A. EXPERIMENTAL SETTINGS

Evaluation Metrics: To evaluate our approach, standard mean intersection over union (mIoU) is used. This metric prefers a class with large areas. However, in a dataset such as PASCAL VOC 2012, the number of pixels varies significantly for different classes. The area of background could be several dozen times larger than other class (as a result, background always has the highest accuracy). Thus, we also give the class-wise IoU score for better comparison. According to [14], the evaluation metrics are computed as:

- Pixel accuracy: ratio of correctly classified pixels to total number of pixels, defined as $\frac{\sum_i t_{ii}}{\sum_i T_i}$.
- Mean IoU: mean intersection over union percentage over all classes, defined as $\frac{1}{N}\sum_i \frac{t_{ii}}{T_i + \sum_j t_{ji} - t_{ii}}$.

where t_{ji} denotes number of pixels belonging to class j whereas predicted to be class i, T_i denotes the total number of pixels in class i, N is the number of classes.

Baseline Model: The Deeplab [10] framework is adopted as the baseline model. According to [14], we first modify ResNet-101 [2] into fully convolution fashion to enable dense prediction. This is accomplished by replacing the last few layers behind *conv*5_*x* with a convolution (classification) layer. In this way, the ResNet part serves as a feature extraction module and the pretrained model provides a proper initialization. The relative learning rate for previous layers is 1 and for the classification layer it is 10. Then, the last two pooling layers in original ResNet are discarded to maintain the feature resolution and dilated convolution is employed to keep the receptive fields unchanged. The dilation rate in *conv*4_*x* is set to 2 and in *conv*5_*x* it is set to 4, so the entire network has a downsample rate of 8. For our network, the conv layer behind *conv*5_*x* is replaced by the dense convolution unit. In our settings, the convolution layers in DCU use no batch normalization. During training and testing, the batch normalization layers in ResNet use the saved parameter, and the learning rate is zero. To fit the size of the output map, groundtruth maps are downsampled by 8 before being fed into the loss layer. In the inference stage, output maps are upsampled by 8 using bilinear interpolation. The loss layer uses softmax function and multinomial logistic loss function (see (2)) to compute the loss as well as pixel-wise gradients for back propagation and is applied after the classification layer.

B. RESULTS ON PASCAL VOC 2012

PASCAL VOC 2012 [48] is a widely used semantic segmentation dataset. Following common practice [10], [11], extra annotations provided by [50] are included in training. Finally, we have 13,487 well annotated images, of which 10,582 images are used for training, 1,449 for validation and 1,456 for testing.

The PASCAL VOC2012 dataset has 20 object classes and one background class. In addition, the dataset contains another annotation which means that area is difficult to seg-

ment, usually the object boundaries. Currently such areas are ignored in training and testing.

Training Protocol: When training the dense prediction networks, we randomly mirror and crop the training images for data augmentation. The crop size is set to 321 × 321. As suggested in [10], [11], the ''poly'' learning rate policy is adopted, the power is set to 0.9, base learning rate is set to 0.00025, momentum and weight decay are set to 0.9 and 0.0005, respectively. Our implementation is based on the Caffe [51] platform with modification from [10]. The batch size is set to 30. The baseline model has huge memory consumption, so a large batch size can be achieved by setting the batch size to 3 and iter_size to 10 in Caffe when training on a single GPU. The training runs for 30K iterations on the augmented PASCAL VOC2012 training dataset. The new added layers are randomly initialized using the ''msra'' manner [52].

1) ABLATION STUDY ON NETWORK ARCHITECTURE

First, we would like to report more details about the results presented in our conference paper and give some discussions.

There is an important factor in the baseline model that affects the accuracy significantly, i.e., the dilation rate r in the classification layer. Although the stacked layers in ResNet 101 bring large enough receptive fields for PASCAL VOC2012 images, a large dilation rate is still necessary in the classification layer. Similar observations can also be found in [10]. This phenomenon is partly explained in [53], i.e., the effective receptive field is not as large as expected. Therefore, we train several baseline models with different r following the protocol introduced above. When $r = 1$, the mean IoU is 69.2% on the validation set; as r becomes larger, the mean IoU improves. We finally choose $r = 12$ as the reference model, which achieves 70.1% mIoU. In the following experiments, we keep the receptive field of the classification layer unchanged to eliminate the effects brought by different receptive fields.

a: RESULTS OF DIFFERENT WEIGHT MAP STRATEGIES

In the baseline model, classification is completed by a standard 2D convolution layer with kernel size of $k = 3$ and dilation rate of $r = 12$. Our network replaces the standard convolution layer with the dense convolution unit with the same hyper parameters. To investigate which strategy best combines the middle activations, three methods are tried for generating the weight map $\widetilde{\boldsymbol{G}}$:

- M1: All activations (both the positions and channels) in the output map share the same weights, which means only 9 weights are needed. In this case, the weights are not generated by previous layers and therefore are not involved in further back propagation. The gradients are computed using a formula similar to (12) and sum up the overall output activations. Finally the network achieves 70.5% mIoU on the validation set.
- M2: Different activations within the same groups are given different weights; thus, weight maps are established. Different output channels share the weight maps. Under this setting, 9 weight maps are produced and expected to encode semantics in different positions. The corresponding mIoU increases to 71.4%.
- M3: Based on previous setting, weight maps are no longer shared across output channels. Compared to M2, less than 0.05% mIoU improvement is observed.

We finally choose the last method; in fact, the second is just as good. In those experiments, the dilation rate of the convolution layer that produces $\boldsymbol{G}$ and $\boldsymbol{y}$ is equal, denoted as $r_{\boldsymbol{G}}$ and $r_{\boldsymbol{y}}$. If $r_{\boldsymbol{G}}$ is changed to 1, a 0.01% loss on mIoU is observed. This indicates that a large receptive field is not necessary when generating weight maps. Therefore, in the following section, $r_{\boldsymbol{G}}$ is always equal to $r_{\boldsymbol{y}}$ for simplicity and will not be mentioned anymore. The comparison of those strategies is given in Table. 1.

TABLE 1. mIoU of different weight map strategies.

	Dilation $r_{\boldsymbol{G}}$	Dilation $r_{\boldsymbol{y}}$	Param_Size	mIoU
baseline(r=12)	—	12	—	70.1
DCU(k=3)_M1	—	12	9	70.5
DCU(k=3)_M2	12	12	166K	71.4
DCU(k=3)_M3	12	12	3.5M	71.4(+0.05)
DCU(k=3)_M3'	1	12	3.5M	71.4(-0.01)

b: COMPARISON OF DIFFERENT METHODS

In the DCU, the parameter size increases rapidly with k. Therefore, only one extra experiment regarding kernel size is carried on with k is set to 5. Since the receptive field is vital for final accuracy, the corresponding dilation rate is adjusted to 6. In this way, the receptive field remains constant. The class-wise IoU results of this setting together with previous experiments are listed in Table. 2. For $k = 5$, the mIoU achieves 71.7%. Under the same receptive field, a larger kernel size could produce better segmentation results. For a large kernel size, the kernel decomposition method proposed in [53] can be adopted to balance accuracy and complexity. The comparison with some other related methods are shown in the top of Table. 3. Overall, the DCU has advantages over its competitors which are still within the standard convolution scheme.

To show the effect of dense convolution clearly and avoid the influence of other factors, many common accuracy-improving tricks are not used here, such as randomly rescaling the training images, averaging the results across serval input scales, pretraining the network on a larger dataset then finetuning on the standard PASCAL VOC2012 training set, using CRFs as postprocess and so on. No batch normalization was adopted in the DCU for a fair comparison with the baseline model.

TABLE 2. Class-wise IoU of different settings.

	bg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mIoU
baseline(r=12)	92.3	85.5	38.6	82.7	64.1	77.5	89.3	82.0	87.2	33.9	74.5	42.7	79.4	71.9	76.6	80.8	53.0	78.0	35.1	81.0	66.6	70.1
DCU(k=3)_M1	92.5	86.4	38.7	82.6	62.5	77.7	89.3	82.6	87.2	33.8	71.4	44.3	80.0	73.0	77.0	82.3	54.5	78.5	38.7	78.6	68.4	70.5
DCU(k=3)_M2	92.8	88.0	40.0	85.0	65.1	78.2	90.3	83.5	88.2	35.7	70.9	42.2	80.8	72.0	78.9	82.9	56.4	79.0	40.1	79.9	70.0	71.4
DCU(k=3)_M3	92.8	86.8	39.7	85.1	66.1	79.1	90.3	82.9	87.9	35.5	72.1	42.8	81.0	72.1	78.8	82.6	56.3	78.8	39.2	80.3	69.7	71.4
DCU(k=5)	92.8	87.0	39.8	84.6	65.0	78.8	90.7	83.6	88.0	36.7	71.8	44.4	80.5	71.5	80.0	83.3	55.4	79.8	39.1	80.1	71.3	71.7

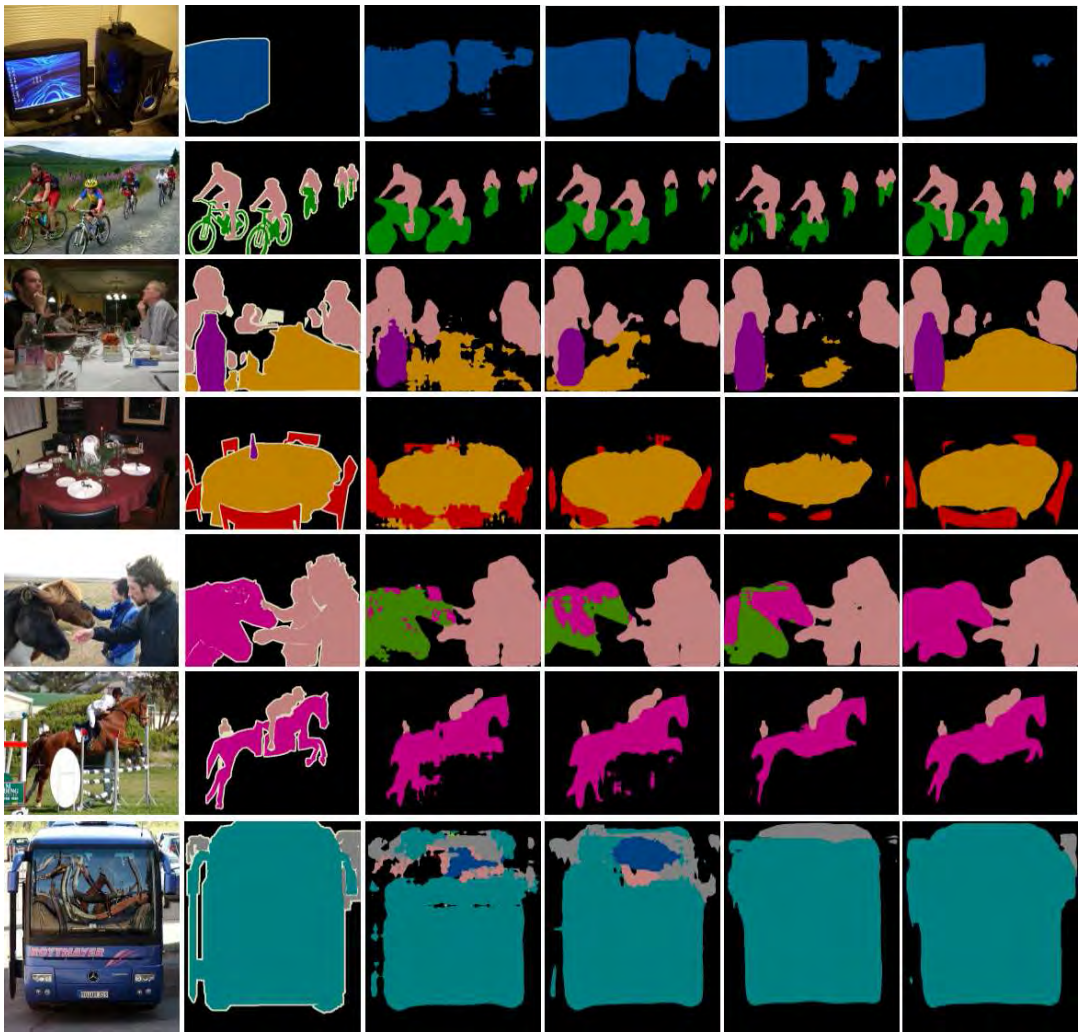

FIGURE 4. Visual comparison of different methods. From left to right: Original Image, Ground Truth, Res101-Conv, Res101-DCU, Res101-SPP-Conv, Res101-SPP-DCU. Conv denotes the standard convolution, DCU denotes the dense convolution unit. Best viewed in color.

2) FURTHER EXPLORATION ON DCU

In the above experiments, the dense convolution unit is compared with the baseline model and some related methods. However, those networks do not have comparable parameters. This makes the results less convincing because it is well known that for deep networks, improvements can be achieved through stacking convolution or deconvolution layers. Meanwhile, the compatibility and generalization of the DCU need to be tested since it is proposed as a substitution to standard convolution. Based on such considerations, we add the spatial pyramid pooling (SPP) module used in [11] to the baseline model and apply the DCU after it. This section tries to demonstrate that 1) the DCU exhibits superior performance compared to straightforward stacking layers and 2) it can be easily integrated into other frameworks and still be effective.

a: DETAILS ON NETWORK STRUCTURE

Note that in the original PSPNet [11], the classification is accomplished by two standard convolution layers. The first layer has 512 output channels with kernel size $k = 3$.

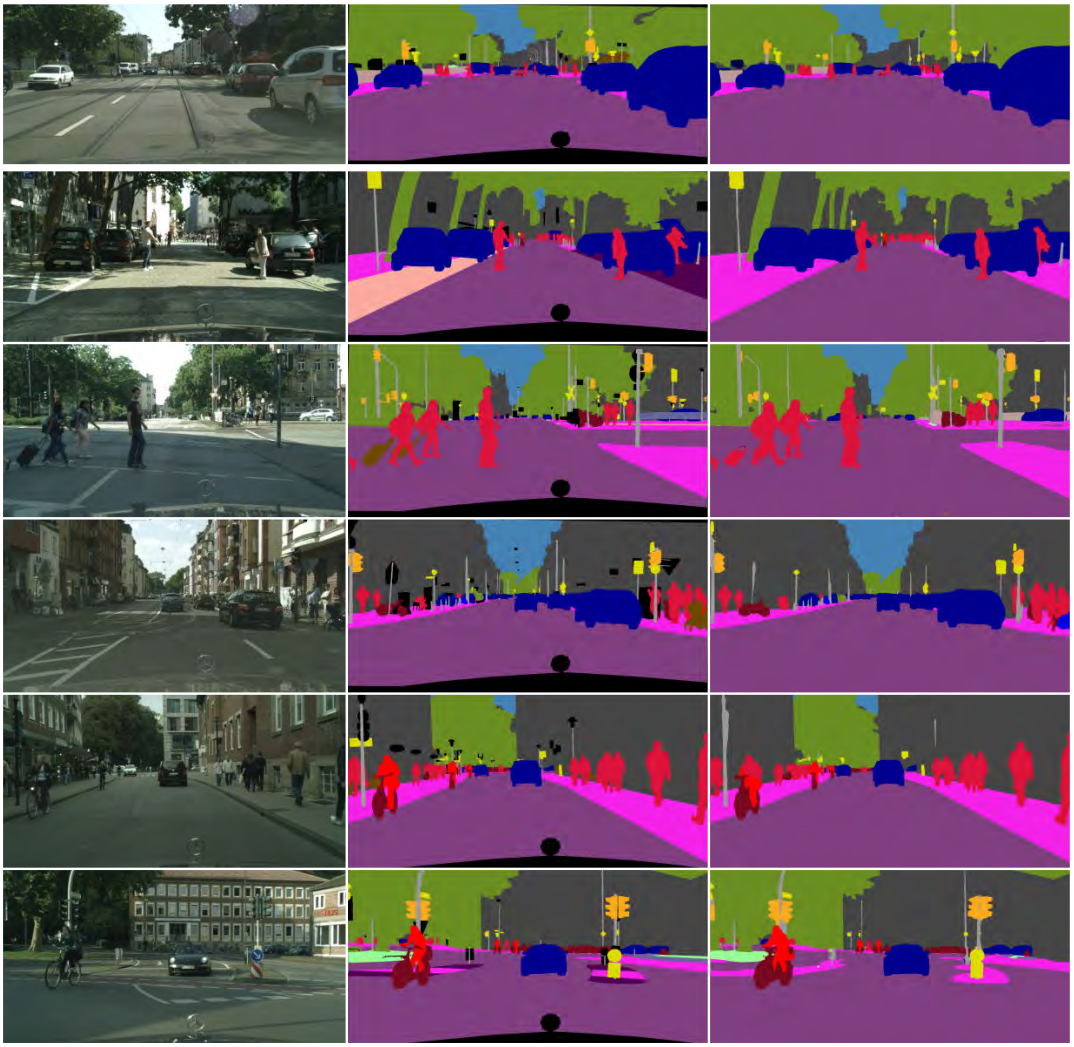

FIGURE 5. Visual examples of our results on Cityscapes validation dataset. From left to right: Original Image, Ground Truth, Res101-SPP-DCU. The model is trained on the training set. Best viewed in color.

TABLE 3. Mean IoU of different methods on PASCAL VOC2012 validation set.

	mIoU
baseline(r=1)	69.2
baseline(r=12): LargeFOV	70.1
DeepLab-v2 MSC [10]	71.2
GCN(k=3) [53]	70.1
GCN(k=5) [53]	71.1
DCU(k=3)	71.4
DCU(k=5)	71.7
Deformable Conv [28]	75.3
Res101-SPP-DCU	76.3

The second layer has 21 output channels with kernel size $k = 1$. Such a setting gives us the chance to conduct controlled experiments since they have comparable parameters with the proposed network. For the DCU, the convolution layers used to produce $\boldsymbol{G}$ and $\boldsymbol{y}$ have 378 channels in total, and the remaining part has no trainable parameters. To keep consistent with [11], one 1×1 convolution layer is placed after the DCU, the crop size is reset to 473 and the size of the output feature map is 60×60 after downsampling 8 times. The spatial pyramid pooling module consists of four pooling layers with kernel size and stride equaling 60, 30, 20, and 10. The output maps of the SPP are then concatenated with its input. All the convolution layers after the SPP module have a dilation rate of $r = 1$. In the experiments, layers behind the SPP module are trained from scratch with "msra" random initialization [52], and the previous part is initialized with pretrained weights in [11] because the model is slightly different from the original Res101. Under such a setting, only batch size 1 is available due to the limited physical memory of the single GPU. The models are trained on 4 TITAN XP GPUs. Limited by the number of GPUs, batch normalization parameters are frozen for the baseline model and not included in those newly added layers after the SPP.

b: CONSISTENT IMPROVEMENTS

The two networks are briefly denoted as Res101-SPP-Conv, Res101-SPP-DCU and trained on augmented training set for

TABLE 4. Comparison of parameter size, forward time and mIoU for different methods.

	Param_Size	Forward Time(sec)	mIoU(%)
Res101-Conv	0.4M	0.071	69.2
Res101-DCU	6.6M	0.082	71.4
Res101-SPP-Conv	13.0M	0.101	71.4
Res101-SPP-DCU	10.6M	0.110	76.3

30K iterations using the protocol described above. Similarly, the previous baseline model with dilation $r = 1$ and its counterpart dense convolution based network with kernel size $k = 3$ and dilation $r = 12$ are called Res101-Conv, Res101-DCU, correspondingly. The comparison of those four methods is presented in Table. 4 in terms of the parameter size(Res101 blocks not included), forward time of whole network and mean IoU. The main conclusions are as follows:

1) Starting from the baseline model, replacing the single convolution layer with the SPP module followed by two convolution layers brings 2.2% improvement of mIoU. The result is very close to that achieved by the DCU with a large FOV (dilation rate). However, the DCU has only half as many parameters, which demonstrates its superior performance over the spatial pyramid pooling module.
2) Applying the DCU upon the SPP increases the accuracy greatly, with a 4.9% mIoU improvement. This is not surprising since dense convolution is designed for enhancing connection and interaction between contexts. Thus, the DCU has the wind at its back when equipped with a powerful context embedding module, i.e., SPP.
3) In terms of dense prediction tasks, the DCU is more suitable than standard convolution. It achieves much higher accuracy even with fewer parameters.
4) The computation and depth brought by the DCU result in approximately 14% extra running time based on our un-optimized CPU-implementation. This is half time of that required by the SPP. In fact, the DCU does not contain many computations and can be optimized with a matrix multiplication manner and performed in the GPU, which will further reduce the time requirement.

As for multiscale training and testing, the images are resized using factors 0.5, 0.75, 1.25, 1.5, and 1.75. During inference, the probability maps are averaged across different scales. Flipped images are also included. Under this condition, the model achieves a 76.8% mIoU. Such improvement remains consistent with other works. Unexpectedly, setting the dilation rate of the DCU in Res101-SPP-DCU to $r = 12$ results in decreased accuracy. The result is listed in Table. 5. The decline in mIoU suggests that the spatial pyramid pooling module provides large enough receptive fields for following classification; thus, a large FoV in the DCU brings limited global context which could not offset the loss in details.

TABLE 5. Experimental results with MS_Mirror testing and larger dilation rate.

MS_Mirror	Dilation Rate	mIoU(%)	Pixel Accuracy(%)
	1	76.3	94.26
√	1	76.8	94.48
√	12	75.9	93.44

TABLE 6. Experimental results on cityscapes test set. only fine data is used during training. ‡ means that the model is trained on both the training and validation set.

Method	IoU Cla(%)	iIoU Cla(%)	IoU (%)	iIoU Cat(%)
CRF-RNN [26]	62.5	34.4	82.7	66.0
FCN [14]	65.3	41.7	85.7	70.1
SiCNN+CRF [54]	66.3	44.9	85.0	71.2
DPN [55]	66.8	39.1	86.0	69.1
Dilation10 [34]	67.1	42.0	86.5	71.1
LRR [56]	69.7	48.0	88.2	74.7
DeepLab [10]	70.4	42.6	86.4	67.7
Piecewise [57]	71.6	51.7	87.3	74.1
RefineNet [12]	73.6	47.2	87.9	70.6
FoveaNet [58]	74.1	52.4	89.3	77.6
PEARL [59]	75.4	51.6	89.2	75.1
TuSimple [32]	77.6	53.6	90.1	75.2
SAC-multiple [60]	78.1	55.2	90.6	78.3
PSPNet [11]	78.4	56.7	90.6	78.6
ResNet-38 [61]	78.4	59.1	90.9	81.1
Res101-SPP-Conv	77.0	54.4	89.9	77.1
Res101-SPP-DCU	77.8	56.1	90.3	78.3
Res101-SPP-DCU‡	78.9	57.8	90.5	78.0

c: VISUAL COMPARISON

In Fig. 4, some visual segmentation results are presented for the four methods. Based on the visualization, the dense convolution unit produces much fewer discontinuities and smoother boundaries in the segmentation map. In the standard convolution, it is inevitable for some positions to encounter confusing contexts within their corresponding receptive fields. Then noise will definitely appear, especially in complicated scenes. However, such discontinuities can be corrected through the overlapped predictions in the DCU, because the right predictions can likely be found from their neighbors.

C. RESULTS ON CITYSCAPES

We also conduct experiments on the Cityscapes [49] dataset with the Res101-SPP-DCU model. Cityscapes is a street scene dataset that contains 2975, 500, and 1525 finely annotated images for training, validation and testing. All the images are $2048 \times 1024\ px$. We use the same settings as above except that the training process takes 90K iterations on this dataset. The main difference in the model structure is that the DCU contains 504 channels ($\boldsymbol{y}$ and $\boldsymbol{G}$ in total), which achieves similar parameters with its baseline counterpart.

The results on the Cityscapes test set are reported in Table. 6 and Table. 7. Multiscale fusing and mirroring are used for testing. Some visual examples of the Cityscapes validation set are presented in Fig. 5. Due to the lack of the batch normalization layer and auxiliary loss as well as the

TABLE 7. Class-wise IoU on cityscapes test set. ‡ means that the model is trained on both the training and validation set.

Method	road	swalk	build.	wall	fence	pole	tlight	sign	veg.	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
CRF-RNN [26]	96.3	73.9	88.2	47.6	41.3	35.2	49.5	59.7	90.6	66.1	93.5	70.4	34.7	90.1	39.2	57.5	55.4	43.9	54.6	62.5
FCN [14]	97.4	78.4	89.2	34.9	44.2	47.4	60.1	65.0	91.4	69.3	93.9	77.1	51.4	92.6	35.3	48.6	46.5	51.6	66.8	65.3
SiCNN+CRF [54]	96.3	76.8	88.8	40.0	45.4	50.1	63.3	69.6	90.6	67.1	92.2	77.6	55.9	90.1	39.2	51.3	44.4	54.4	66.1	66.3
DPN [55]	97.5	78.5	89.5	40.4	45.9	51.1	56.8	65.3	91.5	69.4	94.5	77.5	54.2	92.5	44.5	53.4	49.9	52.1	64.8	66.8
Dilation10 [34]	97.6	79.2	89.9	37.3	47.6	53.2	58.6	65.2	91.8	69.4	93.7	78.9	55.0	93.3	45.5	53.4	47.7	52.2	66.0	67.1
LRR [56]	97.7	79.9	90.7	44.4	48.6	58.6	68.2	72.0	92.5	69.3	94.7	81.6	60.0	94.0	43.6	56.8	47.2	54.8	69.7	69.7
DeepLab [10]	97.9	81.3	90.3	48.8	47.4	49.6	57.9	67.3	91.9	69.4	94.2	79.8	59.8	93.7	56.5	67.5	57.5	57.7	68.8	70.4
Piecewise [57]	98.0	82.6	90.6	44.0	50.7	51.1	65.0	71.7	92.0	72.0	94.1	81.5	61.1	94.3	61.1	65.1	53.8	61.6	70.6	71.6
RefineNet [12]	98.2	83.3	91.3	47.8	50.4	56.1	66.9	71.3	92.3	70.3	94.8	80.9	63.3	94.5	64.6	76.1	64.3	62.2	70.0	73.6
FoveaNet [58]	98.2	83.2	91.5	44.4	51.2	63.2	70.8	75.5	92.7	70.1	94.5	83.3	64.2	94.6	60.8	70.7	63.3	63.0	73.2	74.1
PEARL [59]	98.4	84.5	92.1	54.1	56.6	60.4	69.0	74.0	92.9	70.9	95.2	83.5	65.7	95.0	61.8	72.2	69.6	64.8	72.8	75.4
TuSimple [32]	98.5	85.5	92.8	58.6	55.5	65.0	73.5	77.9	93.3	72.0	95.2	84.8	68.5	95.4	70.9	78.8	68.7	65.9	73.8	77.6
SAC-multiple [60]	98.7	86.5	93.1	56.3	59.5	65.1	73.0	78.2	93.5	72.6	95.6	85.9	70.8	95.9	71.2	78.6	66.2	67.7	76.0	78.1
PSPNet [11]	98.6	86.2	92.9	50.8	58.8	64.0	75.6	79.0	93.4	72.3	95.4	86.5	71.3	95.9	68.2	79.5	73.8	69.5	77.2	78.4
ResNet-38 [61]	98.5	85.7	93.1	55.5	59.1	67.1	74.8	78.7	93.7	72.6	95.5	86.6	69.2	95.7	64.5	78.8	74.1	69.0	76.7	78.4
Res101-SPP-Conv	98.5	85.5	92.5	53.0	60.1	62.7	73.4	77.2	93.1	71.4	94.7	84.4	66.0	95.5	63.3	79.5	68.8	68.0	75.2	77.0
Res101-SPP-DCU	98.6	86.2	92.7	53.7	61.2	64.1	74.4	78.4	93.3	72.0	94.7	85.4	67.8	95.7	64.7	80.4	70.3	69.3	76.2	77.8
Res101-SPP-DCU‡	98.6	86.6	92.8	51.8	62.6	64.4	74.7	78.8	93.4	72.4	94.9	85.9	69.6	95.8	71.7	83.7	74.9	69.9	76.9	78.9

small crop size (473 × 473 here, and 713 × 713 in [11]), our baseline result is a slightly lower than the original PSPNet. In fact, recent improvements on the Cityscapes dataset have been slow. Most advances introduce heavy computations and more parameters. However, our technique still obtains benefits under similar parameters and is quite concise to reproduce.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose the dense convolution unit (DCU) for semantic segmentation. DCU produces multioutputs and introduces spatial overlaps into current convolutions. To obtain the final segmentation map, weight maps are adopted to enable an effective combination of those overlapped activations with learnable parameters. Ablation studies on benchmark datasets demonstrate the effectiveness of the proposed approach and its superior ability over standard convolution. Overall, the dense convolution unit is a promising component for semantic segmentation and can be widely adopted in dense prediction networks.

In the future, we will explore more unified and elegant strategies to enable the dense convolution throughout the whole networks. This indicates the cascade of dense convolution units and requires more effective structures for both computations and combinations. Such specially designed structures for semantic segmentation networks help to reduce the heavy computations in state-of-the-art methods and are essential for their implementations in practical applications.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.

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

[3] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2261–2269.

[4] L. Zhang *et al.*, "Improving semantic image segmentation with a probabilistic superpixel-based dense conditional random field," *IEEE Access*, vol. 6, pp. 15297–15310, 2018.

[5] L. Fan, W. Wang, F. Zha, and J. Yan, "Exploring new backbone and attention module for semantic segmentation in street scenes," *IEEE Access*, vol. 6, pp. 71566–71580, 2018.

[6] T. D. Nguyen, A. Shinya, T. Harada, and R. Thawonmas, "Segmentation mask refinement using image transformations," *IEEE Access*, vol. 5, pp. 26409–26418, 2017.

[7] Z. Jiang, Q. Wang, and Y. Yuan, "Modeling with prejudice: Small-sample learning via adversary for semantic segmentation," *IEEE Access*, vol. 6, pp. 77965–77974, 2018.

[8] M. Naseer, S. Khan, and F. Porikli, "Indoor scene understanding in 2.5/3D for autonomous agents: A survey," *IEEE Access*, vol. 7, pp. 1859–1887, 2019.

[9] J. Fu, J. Liu, Y. Wang, and H. Lu, "Densely connected deconvolutional network for semantic segmentation," in *Proc. Int. Conf. Image Process.*, Sep. 2017, pp. 3085–3089.

[10] 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. 2017.

[11] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 6230–6239.

[12] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5168–5177.

[13] A. W. Harley, K. G. Derpanis, and I. Kokkinos, "Segmentation-aware convolutional networks using local attention masks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 5048–5057.

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

[15] X. Chen, G. Wang, C. Zhang, T.-K. Kim, and X. Ji, "SHPR-NET: Deep semantic hand pose regression from point clouds," *IEEE Access*, vol. 6, pp. 43425–43439, 2018.

[16] R. Wang, Y. Meng, W. Zhang, Z. Li, F. Hu, and L. Meng, "Remote sensing semantic segregation for water information extraction: Optimization of samples via training error performance," *IEEE Access*, vol. 7, pp. 13383–13395, 2019.

[17] Z. Yang, H. Yu, W. Sun, Z. Mao, and M. Sun, "Locally shared features: An efficient alternative to conditional random field for semantic segmentation," *IEEE Access*, vol. 7, pp. 2263–2272, 2019.

[18] M. Martin-Abadal, E. Guerrero-Font, F. Bonin-Font, and Y. Gonzalez-Cid, "Deep semantic segmentation in an AUV for Online Posidonia Oceanica meadows identification," *IEEE Access*, vol. 6, pp. 60956–60967, 2018.

[19] X. Jiang, Y. Gao, Z. Fang, P. Wang, and B. Huang, "An end-to-end human segmentation by region proposed fully convolutional network," *IEEE Access*, vol. 7, pp. 16395–16405, 2019.

[20] L. Fan, H. Kong, W.-C. Wang, and J. Yan, "Semantic segmentation with global encoding and dilated decoder in street scenes," *IEEE Access*, vol. 6, pp. 50333–50343, 2018.

[21] Y. Duan *et al.*, "Hierarchical multinomial latent model with G^0distribution for synthetic aperture radar image semantic segmentation," *IEEE Access*, vol. 6, pp. 31783–31797, 2018.

[22] S. Saito, R. Arai, and Y. Aoki, "Seamline determination based on semantic segmentation for aerial image mosaicking," *IEEE Access*, vol. 3, pp. 2847–2856, 2015.

[23] D. Sakkos, E. S. L. Ho, and H. P. H. Shum, "Illumination-aware multi-task GANs for foreground segmentation," *IEEE Access*, vol. 7, pp. 10976–10986, 2019.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F. F. Li, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[25] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. (2017). "Rethinking atrous convolution for semantic image segmentation." [Online]. Available: https://arxiv.org/abs/1706.05587

[26] S. Zheng *et al.*, "Conditional random fields as recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1529–1537.

[27] S. Chandra and I. Kokkinos, "Fast, exact and multi-scale inference for semantic image segmentation with deep Gaussian CRFs," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 402–418.

[28] J. Dai *et al.*, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 764–773.

[29] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–10.

[30] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4438–4446.

[31] G. Huang, D. Chen, T. Li, F. Wu, L. van der Maaten, and K. Q. Weinberger. (2017). "Multi-scale dense convolutional networks for efficient prediction." [Online]. Available: https://pdfs.semanticscholar.org/6b86/7004cda7d2682159bc745d5ea1ef1bff48fc.pdf

[32] P. Wang *et al.* (2017). "Understanding convolution for semantic segmentation." [Online]. Available: https://arxiv.org/abs/1702.08502

[33] C. Han, X. Tao, Y. Duan, and J. Lu, "Dense convolution for semantic segmentation," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2018, pp. 2222–2226.

[34] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–13.

[35] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1520–1528.

[36] J. Fu, J. Liu, Y. Wang, and H. Lu. (2017). "Stacked deconvolutional network for semantic segmentation." [Online]. Available: https://arxiv.org/abs/1708.04943

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2015, pp. 234–241.

[38] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for scene segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[39] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3640–3649.

[40] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, "Feedforward semantic segmentation with zoom-out features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3376–3385.

[41] B. Singh and L. S. Davis. (2017). "An analysis of scale invariance in object detection–SNIP." [Online]. Available: https://arxiv.org/abs/1711.08189

[42] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-resolution residual networks for semantic segmentation in street scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3309–3318.

[43] S. Hong, H. Noh, and B. Han, "Decoupled deep neural network for semi-supervised semantic segmentation," in *Proc. Neural Inf. Process. Syst.*, 2015, pp. 1495–1503.

[44] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1742–1750.

[45] A. Bearman, O. Russakovsky, V. Ferrari, and F. F. Li, "What's the point: Semantic segmentation with point supervision," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 549–565.

[46] P. Luo, G. Wang, L. Lin, and X. Wang, "Deep dual learning for semantic image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 2718–2726.

[47] I. Ahn and C. Kim, "Face and hair region labeling using semi-supervised spectral clustering-based multiple segmentations," *IEEE Trans. Multimedia*, vol. 18, no. 7, pp. 1414–1421, Jul. 2016.

[48] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[49] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3213–3223.

[50] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 991–998.

[51] J. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Dec. 2015, pp. 1026–1034.

[53] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters—Improve semantic segmentation by global convolutional network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1743–1751.

[54] I. Krešo, D. Čaušević, J. Krapac, and S. Šegvić, "Convolutional scale invariance for semantic segmentation," in *Proc. German Conf. Pattern Recognit.* Cham, Switzerland: Springer, 2016, pp. 64–75.

[55] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1377–1385.

[56] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 519–534.

[57] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3194–3203.

[58] X. Li *et al.* (2017). "FoveaNet: Perspective-aware urban scene parsing." [Online]. Available: https://arxiv.org/abs/1708.02421

[59] X. Jin *et al.*, "Video scene parsing with predictive feature learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 5581–5589.

[60] R. Zhang, S. Tang, Y. Zhang, J. Li, and S. Yan, "Scale-adaptive convolutions for scene parsing," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2050–2058.

[61] Z. Wu, C. Shen, and A. van den Hengel, "Wider or deeper: Revisiting the ResNet model for visual recognition," *Pattern Recognit.*, vol. 90, pp. 119–133, Jun. 2019.

CHAOYI HAN received the B.S. degree in electrical engineering from Tsinghua University, in 2016, where he is currently pursuing the Ph.D. degree. His research interests include image processing and machine learning.

YIPING DUAN received the B.S. degree from the School of Computer Science and Technology, Henan Normal University, Xinxiang, China, in 2010, and the Ph.D. degree from the School of Computer Science and Technology, Xidian University, Xi'an, China, in 2016. She is currently a Research Assistant with the Department of Electronic Engineering, Tsinghua University. Her current research interests include semantic mining, machine learning, and image processing.

XIAOMING TAO received the B.E. degree from the School of Telecommunications Engineering, Xidian University, in 2003, and the Ph.D. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2008, where she is currently a Professor with the Department of Electronic Engineering. Her research interests include wireless communications and networking, and multimedia signal processing.

JIANHUA LU received the B.E. and M.S. degrees from Tsinghua University, Beijing, China, in 1986 and 1989, respectively, and the Ph.D. degree in electrical and electronic engineering from The Hong Kong University of Science and Technology, Hong Kong. Since 1989, he has been with the Department of Electronic Engineering, Tsinghua University, where he currently serves as a Professor. He has authored more than 180 technical papers in international journals and conference proceedings. His research interests include broadband wireless communication, multimedia signal processing, and wireless networking. He is also a Fellow of the IEEE Communication Society and the IEEE Signal Processing Society. He has served as a member of Technical Program Committees in numerous IEEE conferences and served as the Lead Chair of the General Symposium of the IEEE ICC 2008, as well as the Program Committee Co-Chair of the Ninth IEEE International Conference on Cognitive Informatics, in 2010. He has been an Active Member of professional societies. He was a recipient of best paper awards at the IEEE International Conference on Communications, Circuits and Systems 2002, ChinaCom 2006, and IEEE Embedded-Com 2012, and the National Distinguished Young Scholar Fund by the NSF Committee of China, in 2005. He is now a Chief Scientist of the National Basic Research Program (973), China.