Pattern Recognition 90 (2019) 78–86

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Progressively diffused networks for semantic visual parsing

Ruimao Zhang[a], Wei Yang[b], Zhanglin Peng[c], Pengxu Wei[a,*], Xiaogang Wang[b], Liang Lin[a]

[a] Sun Yat-sen University, Guang Zhou, China
[b] The Chinese University of Hong Kong, Hong Kong, China
[c] Sensetime Research, China

ARTICLE INFO

Article history:
Received 28 February 2018
Revised 20 October 2018
Accepted 7 January 2019
Available online 15 January 2019

Keywords:
Visual understanding
Image segmentation
Recurrent neural networks
Representation learning

ABSTRACT

Recent deep models advance the task of semantic visual parsing by increasing the depth of networks and the resolution (size) of the predicted labelmaps. However, the contextual information within each layer and between layers is not fully explored. Long Short Term Memory Networks(LSTM) that learn to propagate information is well-suited to model pixels dependencies with respect to spacial locations within layers and depths across layers. Unlike previous LSTM-based methods that tend to enhance representation of each pixel only by involving the information from adjacent area. This work proposes Progressively Diffused Networks (PDNs) to deal with complex semantic parsing tasks. It can explore spatial dependencies in a larger field that represents the rich contextual information among pixels. The proposed model has three appealing properties. First, it enables information to be progressively broadcast across feature maps by stacking multiple diffusion layers. Second, in each layer, multiple convolutional LSTMs are adopted to generate a series of feature maps with different ranges of contexts. Third, in each LSTM unit, a special type of atrous filters are designed to capture the short range and long range dependencies from various neighbors. Extensive experiments demonstrate the effectiveness of PDNs to substantially improve the performances of existing LSTM-based models.

1. Introduction

Semantic parsing aims to predict the category label of each pixel in an image, and it plays a significant role in many complex vision problems, such as scene understanding and content-based vision search. With the rapid development of representation learning in recent years [1,2], deep Convolutional Neural Networks (CNNs) [3–7] have achieved remarkable progress in the task of semantic parsing due to its hierarchical architecture and end-to-end training strategy. The former transforms the input image into multiple levels of semantic representations, while the latter makes the learned features transferrable. (Fig. 1).

In order to improve the performance of semantic parsing, one way is to employ the deep Fully Convolutional Networks (FCNs). With the depth of networks growing, such as residual networks [7], each site (pixel) on the predicted label maps achieves large receptive field and can make more global reasoning in the dense prediction. For example, based on our experimental results, the mIoU score on ADE20K MIT benchmarks [8] is about 27% when applying 16-layer netwoks (i.e. VGG-16) [3], and this value is promoted by 8% when using 101-layer networks (i.e. Resnet-101) [7]. However, due to the limitation of computation resources, the growth of the depth is unsustainable.

Another branch of works [10–12] tried to explore rich contexts in images. These work reveal that incorporating graphical models such as CRF [13] or MRF [14,15] to smooth the predicted label maps was crucial. This post-processing improved the accuracy of dense prediction. For instance, Liang et al. [13] employed the fully connected pairwise CRF as a post-processing step to further refine the label maps. In [15], mean field algorithm (MF) algorithm was applied to solve MRF iteratively and passes the inference error backward into CNNs, achieving the joint optimization of MRF and CNNs. In [16], Zheng et al. further adopted a Recurrent Neural Networks (RNNs) to represent such inference procedure. In order to reduce the computational cost of the above methods, Liu et al. [14] proposed to approximate MF with convolution and pooling operations. Although these methods exploited the power of graphical models in semantic parsing task, the context modeling process required careful design of the pairwise constraints and did not explicitly enhance the pixel-wise representation, leading to suboptimal parsing results.

An alternative scheme focused on using Long Short Term Memory (LSTM) networks to automatically learn the spatial dependen-

* Corresponding author.
E-mail addresses: ruimao.zhang@ieee.org (R. Zhang), pengzhanglin@sensetime.com (Z. Peng), weipx3@mail.sysu.edu.cn (P. Wei), xgwang@ee.cuhk.edu.hk (X. Wang), linliang@ieee.org (L. Lin).

https://doi.org/10.1016/j.patcog.2019.01.011
0031-3203/

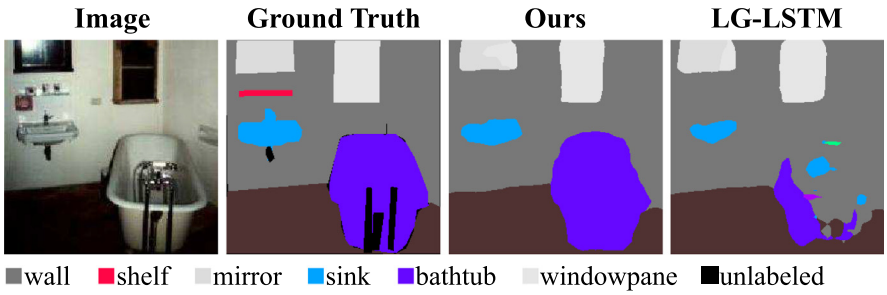

Fig. 1. An example of semantic parsing on ADE 20K dataset. From left to right are input image, ground-truth labeling, semantic parsing result by proposed PDNs and parsing result by LG-LSTM(Res101) [9]. Different from LG-LSTM that enhances representation of each pixel only by considering adjacent area, our model explores spatial dependencies in a larger field and captures the richer contexts among pixels. Therefore, the proposed PDNs can expand the receptive field effectively and achieve the more reasonable global reasoning.

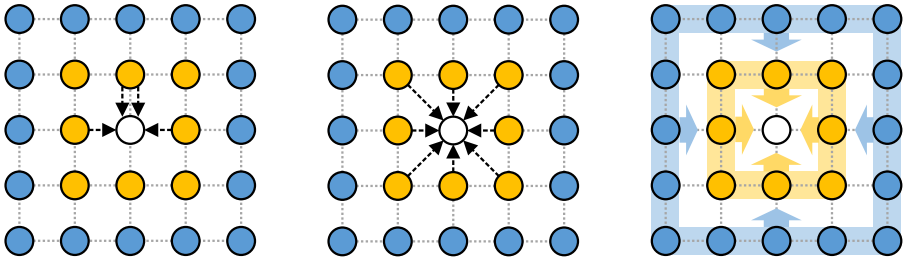

Fig. 2. The comparison of different types of LSTM unit. Sub-figure (a) and (b) show the previous pixel-wise LSTM unit (i.e. Diagonal BiLSTM [21] and LG-LSTM [9]) that update the states of each site by adopting fixed local factors (i.e. adjacent sites). In (c), our proposed LSTM unit can capture the short-range and long-range dependencies from the divers neighbors and can generate more informative data representation.

cies. These data-driven methods applied contextual information to enhance intermediate feature representations This branch of work achieved promising results on recent semantic parsing tasks [9,17–19], where the property of long-range dependencies were used to pass the information between neighbor pixels layer by layer. However, most of the existing approaches [9,19–21] was explored well-designed short distance. A wider range of information diffusion is achieved by stacking multiple LSTM layers. As illustrated in Fig. 2 (a) and (b), the feature enhancement of each position by the above methods is determined by the short distance neighbors (e.g. the closest 2 to 8 adjacent positions), limiting the breadth and the speed of information propagation. At the same time, all of the existing methods were based on fully-connected LSTM, whose computational cost was another limitation.

In this paper, we propose a novel Progressively Diffused Networks (PDNs) that extend the traditional neural network structure by spreading the contextual information on image feature maps, and demonstrate its superiority on various complex semantic parsing tasks. PDNs introduce a stack of information diffusion layers for context modeling, each of which contains several multi-dimensional LSTMs.

The advantage of proposed PDNs can be summarized as two aspects: the diversity of propagated contextual information and the speed of propagation in each layer. First, in convolution based deep semantic segmentation models, the context information propagated from one location to its different neighbors is fixed in a certain layer. In contrast, by using various diffused LSTM units in our model, each location can propagate the different contextual information to its neighbors with different distances. Thus our model effectively expands the scope of communication and is more adequate to represent richer contextual information. Second, compared with LSTM methods, the contextual information can spread faster by using multiple LSTM units in a single layer.

Specifically, we propose two types of diffused LSTMs. One is called spatial LSTM and the other is depth LSTM. Each spatial LSTM in the diffusion layer generates a certain type of contextual feature maps. Intuitively, these contextual feature maps have different meanings comparing with the ones generated by traditional CNNs. Each entry in the convolutional maps represents the response of a local area under a certain pattern. In contrast, each site of the contextual feature maps involves the information that is propagated to its neighbors in the next state. Different with spatial LSTM, the depth LSTM [9] is adopted to realize the communication of each site from one layer to the next.

In each diffused LSTM unit, each type of contextual feature maps is corresponding to a special atrous filter [22], which is used to capture the diverse neighborhood information in a large range of local area. Finally, these filtering results will be integrated to calculate the information of each site passed to its neighbors or to itself in the next layer. Compared with the fully-connected LSTM in previous works [9,18–20], this convolution-based version is more intuitive, and can significantly improve the computational efficiency.

This paper has following three contributions. (1) We propose a Progressively Diffused Networks, which contain a module composed of multiple diffusion layers. Such module is fully differentiable and can be flexibly embedded into deep neural networks for explicitly capturing contextual dependency among image locations. (2) The special type of atrous filters are incorporated into proposed diffusion layers, each of which is corresponding to a special kind of contextual feature map. Through the convolutional operation, each site can receive information from distinct neighbors to further enhance its feature representation. (3) We obtain significant improvement on two challenging datasets (i.e. ADE20K MIT Dataset [8], PASCAL-Part Dataset [23]) compared with previous LSTM-based contextual modeling methods.

2. Related work

The performance of computer vision tasks is heavily dependent on the choice of visual representation. For that reason, many of previous efforts in deploying computer vision models focused on designing the pipelines to extract the effective visual representation [24–26]. Such feature engineering based methods are important but require a lot of domain knowledge, severely limiting the development of visual applications. In order to make the vision models less dependent on feature engineer, representation learning [1,2], which facilitates useful information extraction from raw data for building predictors, has attracted much attention in the past decade.

A typical representation learning model is Convolutional Neural Networks (CNN) [4,27–29], which is designed to process the data with multiple arrays such as images [4] or videos [30]. By stacking several convolution-pooling layers, this model transforms the visual representation from one level into a slightly more abstract level. Recently, many works tend to enhance representational power of CNN by increasing the depth of architectures [5,7,31,32], and achieve great success on image classification [4,7]. The dense prediction task, such as semantic parsing [33–35], has also benefited from such deep feature learning methods [13,28]. In [28], Long and Shelhamer firstly replaced fully-connected layers of CNN with convolutional layers, making it possible to accomplish pixel-wise prediction in the whole image by the deep model. Chen et al. [22] further proposed the atrous convolution to explicitly control the resolution of feature responses, and exhibited the atrous spatial pyramid pooling for dense predicting at multiple scales. In [36], Wang et al. proposed transition layers upon Deconvolutional Networks (DCNN) to make the predicted video segmentation results consistent in spacial and temporal domains.

Meanwhile, in order to explicitly discover the intricate structures in the visual data for dense labeling, the graphic models [37] were applied to explore the rich information (e.g. long-range dependencies or high-order potentials) in the image by defining the spatial constrains. In [13], the confidence maps generated by the Fully Convolutional Networks (FCN) [28] were fed into the Conditional Random Field (CRF) with simple pairwise potentials for post-processing, but this model treated the FCN and CRF as separated components, limiting the joint optimization of the model. In contrast, Schwing and Urtasun [15] jointly train the FCN and Markov Random Field (MRF) by passing the error generated by MRF back to the neural networks. However, the iterative inference algorithm (i.e. Mean Field inference) used in this method is time consuming. To improve computational efficiency, Liu et al. [14] solve MRF by the convolution operations, which devises the additional layers to approximate the mean field inference for pairwise terms. Although these methods significantly improve the performance of dense labelling, the contextual information is still not explicitly encoded into the pixel-wise representations.

In the literature, the Long Short Term Memory (LSTM) Network has been introduced to deal with the long-range dependencies in the representation modeling, and this advanced Recurrent Neural Network (RNN) has achieved great success in many intelligent tasks [38–41]. In recent years, it has been extended to multi-dimensional communication [18,20,42] and adapted to represent the rich contexts in image spatial [9,19]. In [9], a recent advance in LSTM-based context modeling was achieved by considering both short dependencies from local area and long-distance global information from the whole image. Liang et al. [19] further extended this work from multi-dimensional data to general graph-structured data, and constructed an adaptive graph topology to propagate contextual information between adjacent superpixels. Nevertheless, in these works, the feature representation of each position is affected by a limited local factors (i.e. the adjacent positions), which restricts the capacity of involving diverse visual correlations in a large range. Different from using limited local LSTM units, the proposed PDNs captures the short-range and long-range dependencies from various neighbors and can generate more informative representation for pixel-wise prediction.

3. Network overview

An overview of the proposed framework is illustrated in Fig. 3. We define two kinds of diffused LSTMs, i.e. depth LSTM and spatial LSTM, as our basic contextual information processing units. As shown in Fig. 3(a), the diffused LSTM layer, which includes one depth LSTM and several spatial LSTMs, are used to spread the context information among different locations and to generate multiple contextual feature maps for the next layer. The diffused module is obtained by stacking several diffused LSTM layers.

In practise, given an input image, we first extract its feature maps with a Deep Convolutional Neural Networks (DCNNs, e.g., ResNet-101 [7]). Then these feature maps are fed into a series of diffusion layers to progressively spread the context information on the image plane. After each diffused LSTM layer, the generated depth maps (i.e. white maps in Fig. 3) are convolved with 1×1 filters to calculate the score maps for dense prediction. For the model training, intermediate supervision is used for each diffusion layer. We use the cross-entropy loss over all pixels as the loss function for training. In testing phase, final prediction is obtained according to the output confidence maps of the entire networks.

The diffusion layer exploits the multidimensional convolutional LSTMs (i.e. denoted as depth LSTM, spatial LSTM-1, spatial LSTM-2 and so on) to receive and broadcast the information. For each convolutional LSTM, the input is a set of contextual feature maps in current state, while the output is a set of special type of contextual feature maps for next state. In other words, each LSTM unit can receive information generated by all the LSTMs from previous diffusion layer. Specifically, depth LSTMs are used to mix the information from previous layer and generate feature representation of each site in next layer, while spatial LSTMs are exploited to spread information spatially. In spatial LSTMs, each site propagates information to its n-nearest neighbors, where n varies for different spatial LSTMs. This allows the diffusion layer to obtain multi-context information. As illustrated in Fig. 3(b), spatial LSTM-1 outputs the yellow maps and each site on this map contains the information that it passes to its closest 8 neighbors in next state. Similarly, the blue map is generated by spatial LSTM-2, and each site contains the information spreading to its second closest 16 neighbors. It should be noted that the input contextual feature maps to the first diffusion layer are the same, and they are the different copies of the CNN output.

With the convolutional LSTMs, the prediction of site α is affected by different types of neighbors (e.g., 8 closest neighbors or 16 second closest neighbors). Fig. 3(c) gives the details of convolution operations in each LSTM unit. When contextual feature maps with different meanings have been fed into LSTM unit, some special type of atrous filters are applied to capture the short range and long range dependencies from various neighbors to a certain site and pass the accumulated information to the next state. For each kernel, the green regions are learnable and others are always set to zero. Thus depth feature maps specify the depth filter and only the center of the kernel has weight value. The e-th spatial filter is associated with e-th group contextual feature maps, and it introduces non-zero weights in the sites whose distance to the kernel center is e. Note that, if the site α has neighbor α' with distance e, we need to adopt the information of site α' in the e-th group contextual feature maps to enrich the representation of site α.

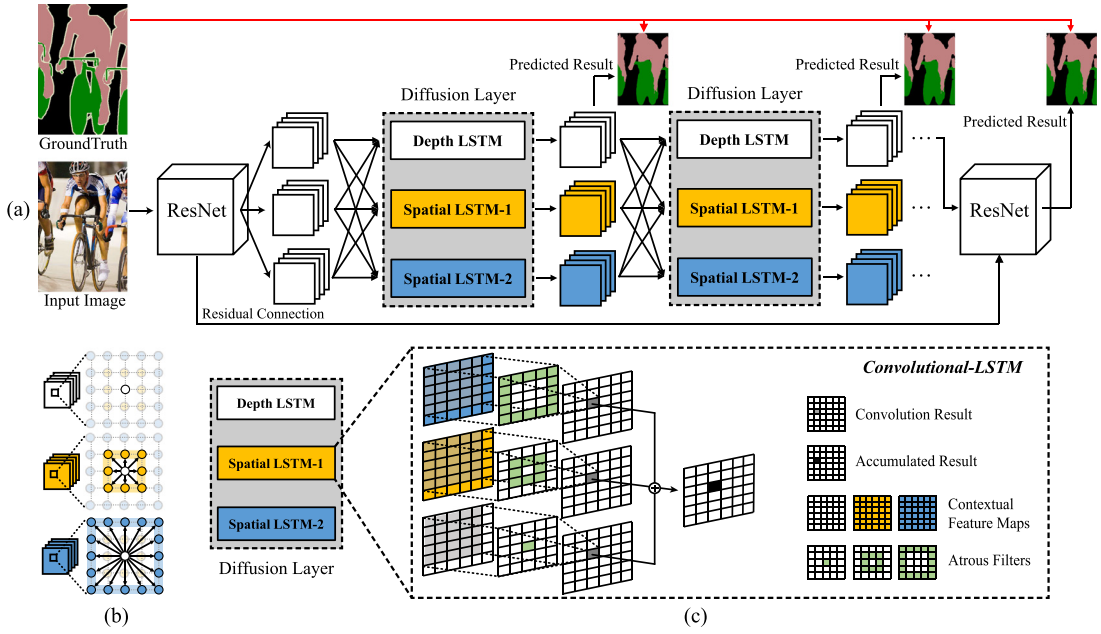

Fig. 3. The framework of proposed Progressively Diffused Networks. (a) Several diffusion layers are embedding into the deep Convolutional Neural Networks for context modeling. Each diffusion layer outputs several contextual feature maps (i.e. yellow and blue maps) for broadcasting neighborhood information on the image plane in the next layer. And depth feature maps (i.e. white maps) are also generated to communicate information of each site from one layer to the next. Note that each diffusion layer can have multiple spatial LSTM, and we only use two as an example in this figure. (b) The contextual feature maps with different meanings. The white maps are the output of Depth LSTM, and each site indicates the information passing from the corresponding site in previous state. The yellow maps are generated by spatial LSTM-1, and each site contains the information that the site will pass to its closest 8 neighbors in the current state. Similarly, each site in the blue maps denotes the information spreading to its second closest 16 neighbors. (c) Specific process in each LSTM unit. A special type of atrous filters are designed to capture the short range and long range dependencies from various neighbors to a certain site and pass the accumulated information to the next state. For each kernel, the green regions are learnable and others are always set to zero. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4. Progressive diffused networks

The proposed PDNs aim at broadcasting the contextual information on the image plane to increase the discrimination of feature representation for each pixel. This diffusion mechanism is partially inspired by the observation of biological research that the pheromone released by a single cell can affect not only adjacent cells, but also larger tissue areas [43]. Hence we propose the PDNs to incorporate multi-contextual information by spreading information from one site to a large field of neighbors. It includes two coherent aspects: (1) Using the information from different neighbors to enrich the feature representation of one certain site, as illustrated in Fig. 2. (2) Propagating different information from the certain site to its different neighbors to guide their further representations, as illustrated in Fig. 3.

The progressive diffused networks in this paper are similar to recent image processing work based on LSTM [9,18,20,21]. These work, however, use fixed factorization (e.g., 2 to 8 neighboring positions) to gather the contextual information of each position, as shown in Fig. 2(a, b). Different from these locally fixed LSTM units, the modified LSTM in our PDNs allows each location to receive messages from different numbers of neighbors, as illustrated in Fig. 2(c). For most of the previous approaches [9,20], the parameters of each LSTM are shared, thus the information that each site passes to all of its neighbors is equivalent. Therefore, these methods can be viewed as a special case of proposed diffusion networks.

4.1. Receiving information in current layer

We define two kinds of convolutional LSTM in the proposed diffusion layer, named depth LSTM and spatial LSTM, by following the definition in [9]. Intuitively, the depth LSTM maintains the information from previous state at each site by applying the memory cells benefited from the LSTM mechanism. The spatial LSTM calculates the information that each position travels outward to their neighbors. Note that different spatial LSTMs adopt different n-nearest neighbors with different distances when propagating information. For example, the yellow maps in Fig. 3 are the outputs of spatial LSTM-1, and the value in each position denotes the information that the position propagates to its $3 \times 3 - 1 = 8$ nearest neighbors with distance 1. Analogously, the blue maps are the outputs of spatial LSTM-2, and the value in each position denotes the message passing to the $5 \times 5 - 3 \times 3 = 16$ second nearest neighbors with distance 2.

As illustrated in Fig. 3, the input image is corresponding to $E+1$ groups of hidden cell maps (denoted by contextual feature maps in Fig. 3), which are generated by one depth LSTM and E spatial LSTM. We set $E = 2$ in this article for illustration. Let $\mathcal{H}^s_{t,e} \in \mathbb{R}^{M\times N\times D}$, $e \in \{1, 2, \ldots, E\}$ denote the e-th group of hidden cell maps generated from e-th spatial LSTM, and the hidden cells in each position are used to propagate the information to its $(2e+1)^2 - (2e-1)^2 = 8e$ neighbors with distance e. Let $\mathcal{H}^d_t \in \mathbb{R}^{M\times N\times D}$ indicate the hidden cell maps calculated by the depth LSTM using the weights updated in the t-th layer. Thus the gate values of a certain LSTM unit (e.g., depth LSTM or spatial LSTM) in t-th layer can be calculated by,

$$\begin{aligned} g^i_t &= \sigma\left(\textstyle\sum_e \{W^s_{t,e}\}^i * \mathcal{H}^s_{t,e} + \{W^d_t\}^i * \mathcal{H}^d_t + b^i_t\right) \\ g^f_t &= \sigma\left(\textstyle\sum_e \{W^s_{t,e}\}^f * \mathcal{H}^s_{t,e} + \{W^d_t\}^f * \mathcal{H}^d_t + b^f_t\right) \\ g^c_t &= \sigma\left(\textstyle\sum_e \{W^s_{t,e}\}^c * \mathcal{H}^s_{t,e} + \{W^d_t\}^c * \mathcal{H}^d_t + b^c_t\right) \\ g^o_t &= \tanh\left(\textstyle\sum_e \{W^s_{t,e}\}^o * \mathcal{H}^s_{t,e} + \{W^d_t\}^o * \mathcal{H}^d_t + b^o_t\right) \end{aligned} \tag{1}$$

where $*$ denotes the convolution operator and the symbol σ indicates the sigmoid function. $W^s_{t,e}$ and W^d_t indicate the weights of kernels associated with e-th spatial hidden cell maps and depth

hidden cell maps in t-th layer. And the superscripts i, f, c and o correspond to distinct state gates.

When calculating the gate values, the convolutional operations in Eq. (1) allow each site to receive information from distinct neighbors. This is similar to feature enhancement in LG-LSTM [9]. The difference is that LG-LSTM needs to combine the feature representations from multi-neighbors and adopts the fully-connected operation to calculate these gate values.

4.2. Propagating information to next layer

Denote the e-th group of memory cells for the spatial dimension as $\mathcal{M}^s_{t,e} \in \mathbb{R}^{M\times N\times D}$ and the memory cells for depth dimension as $\mathcal{M}^d_t \in \mathbb{R}^{M\times N\times D}$. Same as convolutional LSTM, the novel hidden cell maps and memory cell maps in $t+1$-th layer are computed as,

$$\begin{aligned}
(\mathcal{H}^s_{t+1,1}, \mathcal{M}^s_{t+1,1}) &= \text{LSTM}(\mathcal{H}^\dagger_t, \mathcal{H}^d_t, \mathcal{M}^s_{t,1}, \{\mathbf{P}\}^s_{t,1}) \\
(\mathcal{H}^s_{t+1,2}, \mathcal{M}^s_{t+1,2}) &= \text{LSTM}(\mathcal{H}^\dagger_t, \mathcal{H}^d_t, \mathcal{M}^s_{t,2}, \{\mathbf{P}\}^s_{t,2}) \\
&\cdots \\
(\mathcal{H}^s_{t+1,E}, \mathcal{M}^s_{t+1,E}) &= \text{LSTM}(\mathcal{H}^\dagger_t, \mathcal{H}^d_t, \mathcal{M}^s_{t,E}, \{\mathbf{P}\}^s_{t,E}) \\
(\mathcal{H}^d_{t+1}, \mathcal{M}^d_{t+1}) &= \text{LSTM}(\mathcal{H}^\dagger_t, \mathcal{H}^d_t, \mathcal{M}^d_t, \{\mathbf{P}\}^d_t)
\end{aligned} \tag{2}$$

where $\mathcal{H}^\dagger_t = \{\mathcal{H}^s_{t,e}\}_{e=1}^E$ is the set of spatial hidden cell maps. $\mathbf{P} = \{\mathbf{W}, \mathbf{B}\}$ indicates the parameter set. Any hidden cell maps or memory cell maps for the next diffusion layer can be calculated by the following formula:

$$\begin{aligned}
\mathcal{M}_{t+1} &= g^f_t \odot \mathcal{M}_t + g^i_t \odot g^c_t \\
\mathcal{H}_{t+1} &= g^o_t \odot \tanh(\mathcal{M}_{t+1})
\end{aligned} \tag{3}$$

where $\odot$ denotes the Hadamard product.

In the above process, different numbers of spatial LSTMs allow the model to arbitrarily enlarge *field-of-view* in the context modeling. For a LSTM unit in the certain layer, there exist $E+1$ filters with distinct forms, and each one is associated with a group of hidden cell maps. In this way, each site in the input image can provide distinct guidance to its neighbors with different distances in the next diffusion layer, by employing specific spatial LSTMs, which takes the spatial layouts and interactions into account for feature learning. In order to ensure different neighbors receive various information, the weight matrices $\mathbf{W}^s_t$ and bias $\mathbf{B}^s_t$ of E spatial LSTMs are not shared in this article.

4.3. Comparison with existing methods

In the literature, the Long Short Term Memory (LSTM) Networks have been introduced to deal with the long-range dependencies in the representation modeling, and this advanced Recurrent Neural Networks (RNNs) have achieved great success in many intelligent tasks [38–41].

In recent years, it has been extended to represent the rich contexts in image space [9,19,21]. Our work is close to these work but there exit distinct difference. As showed in Table 1, in order to illustrate the advantages of our Diffused LSTM, we compare our approach and related models in [21] and [9] from four aspects: (1) *Parameters of each LSTM unit*. Both our Diffused LSTM and Diagonal BiLSTM [21] apply the convolutional operation, thus the number of parameters is proportional to the feature dimension, the only factor that causes the difference is the number of involved neighbors in the LSTM unit. In contrast, the number of LG-LSTM [9] parameters is proportional to the square of the feature dimension, since it exploits the fully connected operation in each LSTM. All of the LSTM units listed in Table 1 contain 4 gate operations, thus the numbers of parameters are multiplied by 4. (2) *Total parameters of the LSTM module*. The parameters of the entire module will be related to the number of LSTM layers. In this article, we use 4 stacked LSTM layers and this number is less than the previous works. On the other hand, both our model and LG-SLTM [9] use multiple LSTM units in each layer, which also increases the number of related parameters. In practice, the feature dimension of LG-SLTM and our method are 64 and 256, thus for the best model of these two method, total parameters of LSTM module are about 2949K (LG-LSTM w/G) and 307K (Diffused LSTM-24NB), respectively. Therefore, even with the higher feature dimension, our method still has less parameters in total. (3) *The incremental receptive field size*. This item shows the change of receptive field [44] of each site before and after adding the LSTM module. The variable r in the Table 1 is the side length of receptive field. When our model uses 8 nieghborhood context information, the incremental receptive field is slightly smaller than LG-LSTM [9], mainly because our model use less LSTM layers. In contrast, if the number of neighbors increases to 24, the advantage will be highlighted. Since the receptive field in Diagonal BiLSTM [21] is not a square, the incremental receptive field is not listed. For fair comparison, we assume that the LSTM module is stacked on the top of Resnet101 [22]. In such case, the value of s is 8 and the increased of LG-LSTM and our Diffused LSTM are $(64+r)^2 - r^2$ and $(96+r)^2 - r^2$, where r is the respective field of each site on the output feature maps of ResNet101. Obviously, our method can enlarge the receptive field more effectively. (4) *GPU memory cost*. According to Table 1, the cost of GPU memory is positively correlated to the total number of parameters. The feature dimension of Diagonal BiLSTM and LG-LSTM are 512 and 64, which are the standard setting in their article. This value is set as 256 for our method. When we calculate this result, the size of feature maps is downsampled to 40×40 by convolutional operations.

5. Experiments

In this section, we demonstrate the effectiveness of proposed PDNs in semantic parsing tasks. In the following, we first give a brief overview of the datasets and evaluation metrics. Then we evaluate different architecture variants to verify the validity of important components in our model. The performance of PDNs on both scene parsing and human parsing tasks are investigated at the end of this section.

Datasets and evaluation metrics. We validate the effectiveness of proposed PDNs on two challenging semantic parsing datasets. **ADE20K MIT** [8] is a large-scale dataset for scene-centric semantic parsing task. It includes 150 semantic categories, and most categories have the similar appearance. In this dataset, 20,210 images are employed for model training and another 2000 images for validation. **PASCAL-Person-Part** dataset is a fine-grained human parsing benchmark collected by Chen et al. [23] from PASCAL VOC 2010 dataset. It contains the detailed part annotation for each person and these annotations are merged into six person parts (i.e. Head, Torso, Upper/Lower Arms and Upper/Lower Legs) and one background category [19,45,46]. Totally, 1716 images are used for model training and 1817 for test.

Implementation details. In our experiments, our architecture is implemented based on Caffe platform [47] and our models are trained according to two settings. The first one is based on single NVIDIA GeForce GTX TITAN X GPU with 12GB memory. In such case, the parameters in the BN layers are fixed and batch size is 1. The second one is based on the eight TITAN X GPUs with 12GB memory, thus the parameters in the BN layers are updated by the synchronized cross-GPU strategy, which is also applied in PSPNet [48]. In this case, the batch size is set as 16. The input image is randomly cropped to 321×321 for model training. Four diffusion layers are stacked as a module and inserted into different part of the convolutional neural networks (i.e. ResNet-

Table 1
The comparison of our model, Diagonal BiLSTM [21] and LG-LSTM [9] from four aspects: (1) parameters of each LSTM unit, (2) total parameters of the LSTM module, (3) the incremental receptive field, (4) GPU memory cost. For LG-LSTM, 'w/o G' and 'w/ G' indicate without and with considering the global information. The suffix 'NB' in our method denotes the number of neighbors adopted for context modeling. The variable d in the second and third column indicates the feature dimension of each site in the feature maps, r in the fourth column is the side length of receptive field of each site, and s denotes the offset of receptive fields of two adjacent sites. In practice, the variable d in LG-LSTM, Diagonal BiLSTM and our method are 64, 512 and 256. The value of s on the top of ResNet101 is 8 [22].

Method	Parameter of each LSTM	Total parameter of the module	Increasedreceptive field	GPU cost
Diag. BiLSTM [21]	$4 \times (4 \times d) = 16\ d$	$12 \times 4 \times (4 \times d) = 192\ d$	–	$\approx 0.1G$
LG-LSTM w/o G [9]	$4 \times (9 \times d^2) = 36\ d^2$	$5 \times 2 \times 4 \times (9 \times d^2) = 360\ d^2$	$[s \times (9-1)+r]^2 - r^2$	$\approx 4.0G$
LG-LSTM w/ G [9]	$4 \times (18 \times d^2) = 72\ d^2$	$5 \times 2 \times 4 \times (18 \times d^2) = 720\ d^2$	–	–
Diffused LSTM-8NB	$4 \times (9 \times d) = 36\ d$	$4 \times 2 \times 4 \times (9 \times d) = 288\ d$	$[s \times (7-1)+r]^2 - r^2$	$\approx 0.2G$
Diffused LSTM-24NB	$4 \times (25 \times d) = 100\ d$	$4 \times 3 \times 4 \times (25 \times d) = 1200\ d$	$[s \times (13-1)+r]^2 - r^2$	$\approx 0.9G$

Fig. 4. Visualization of *valid receptive field* (VRF) introduced by Zhao et al. [49]. The input image from PASCAL-Person-Part dataset is showed in (a). The VRF of the red dot calculated on the output feature maps of ResNet101 [22] is showed in (b). Obviously, our method adopting 8 neighbors (c) and 24 neighbors (d) can further enlarge the VRF of each site in the score maps. Best to enlarge three times. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

101[1]) with the residual connections. We fine-tune the model based on the pre-trained Convolutional Neural Networks. The learning rate of the newly added layers is initialized as 2.5×10^{-3} and that of other previously trained layers is initialized as 2.5×10^{-4}. All the parameters in the diffusion layers are randomly initialized from a Gaussian distribution with the mean 0 and the variance 0.01. We train all the models using stochastic gradient descent with the momentum of 0.9, and weight decay of 0.0005.

5.1. Ablative study

In this subsection, we conduct ablation experiments on PASCAL-Person-Part dataset to validate effectiveness of different components of our model. For all succeeding experiments, the size of final output feature maps are 1/8 of the original image. If there is no special explanation, we use 24 neighbors to enrich the feature representation of each element on the feature maps in this subsection.

Valid receptive field. In literature, the receptive fields (RF) of the neural units can be computed according to the network architecture (i.e. the filter size, stride and number of layers) [44]. Large RF can be obtained by simply enlarging the filter size and the stride in each layer or constructing deeper networks. Recently, the concept of valid receptive field (VRF) is introduced in [49,50]. It is a sub-region in the receptive field, which provides the valid information to the central point for its prediction. For traditional deep convolutional neural networks such as ResNet101 [7], due to the network is very deep, the receptive field is usually as large as the input image. However, as showed in Fig. 4(b), the valid receptive field [49] still fails to cover every pixels on a certain person part in sometimes. In this experiment, we insert the LSTM module into the middle position between ResNet101 5b layer and 5c layer. With the number of neighbors growing, e.g., from 8 neighbors in Fig. 4(c) to 24 neighbors in Fig. 4(d), the VRF of each site in the score maps can be enlarged obviously. It demonstrates that the proposed LSTM module can involve rich contexts among data element and further promote the ability of global reasoning of each site.

Table 2
Experimental results of different numbers of diffusion layers on PASCAL-Person-Part dataset. The 'syn†' indicates BN synchronization when using multiple GPUs. Note that all of the above accuracies are not the final results of the model, but the prediction results of the LSTM module in the network.

Number of Layers	pixel acc. %	mIoU %	syn† pixel acc. %	syn† mIoU %
PDNs (1 layer)	93.0	61.6	95.4	65.2
PDNs (2 layers)	93.1	61.8	95.7	66.3
PDNs (3 layers)	93.3	62.4	95.9	67.1
PDNs (4 layers)	**93.3**	**62.5**	**96.2**	**67.7**

Performance ofdifferent diffusion layers. We evaluate the performance of different diffusion layers to validate that information diffusion can improve the discriminative ability of each site. Table 2 shows the breakdown IoU results, and the improvement can be observed by gradually adopting more diffusion layers. It demonstrates that stacking multiple LSTM layers can effectively diffuse the information on the feature maps and enhance the feature representation of each site for dense prediction. At the same time, we can find that the margin between the third layer and the fourth layer is small. Intuitively, we can assume that the information has been effectively propagated in the feature maps in the fourth layer. In practice, we insert the LSTM module into the middle position between ResNet101 5*b* layer and 5*c* layer, and add the softmax classifier on the top of each diffusion layer. Thus the prediction results in Table 2 are based on the output confidence maps of different diffusion layers, but not the final results of the model. The final segmentation accuracies of the full model are reported

[1] http://liangchiehchen.com/projects/DeepLab.html.

Table 3
The breakdown time consuming (millisecond per image) of each model. The results are based on the average time cost of all images in the test set. The marks '1M' and '2M' indicate the number of diffused module inserted into ResNet101 architecture. Note that LG-LSTM [9] in this table is our implementation by using Resnet101 [7] as the bottom neural networks.

Method	Res101	CRF	LSTM	Total
DeepLab-v2 [22]	381 ms	4773 ms	–	5154 ms
LG-LSTM [9]	381 ms	–	294 ms	675 ms
PDNs-1M	381 ms	–	112 ms	493 ms
PDNs-2M	381 ms	–	224 ms	605 ms

Table 4
Experimental results on ADE20K *val* set.

Method	BaseNet	BN Syn.	pixel acc. %	mIoU %
FCN [28]			69.05	24.86
DeepLab [13]			71.06	27.08
Deeplab-v2 [7]	Res101		75.09	35.07
Deeplab-v2+CRF [7]	Res101		77.22	36.79
PSPNet [48]	Res101	✓	80.64	**41.96**
Grid-LSTM [20]			69.37	25.11
LG-LSTM [9]			69.91	25.79
LG-LSTM [9]	Res101		77.17	36.41
PDNs-8NB-1Module	Res101		77.10	36.18
PDNs-24NB-1Module	Res101		77.51	37.02
PDNs-24NB-2Modules	Res101		77.59	37.71
PDNs-8NB-1Module	Res101	✓	79.69	40.87
PDNs-24NB-1Module	Res101	✓	80.30	41.56
PDNs-24NB-2Modules	Res101	✓	**80.81**	41.89

in Tables 4 and 5, which are calculated by up-sampling confidence maps extracted from ResNet 5c layer 8 times.

Time consuming. Since the CRF-based post processing [22] also considers the spatial dependence between different sites. In Table 3, we report the time consuming of DeepLab-v2 [22] and various LSTM-based structures in test phase. Our PDNs can realize fast computation in testing without requiring extra inference for solving CRF. At the same time, since the parameters of our model are less than LG-LSTM, the total time consuming for testing is less as well. Therefore, our approach is more suitable for large-scale semantic parsing task.

5.2. Experiment results and comparisons

ADE20K dataset [8]. We compare our model with six semantic parsing methods: FCN [28], DeepLab [13], Grid-LSTM [20], LG-LSTM [9], DeepLab-v2 [22] and PSPNet [48]. These methods can be grouped into two categories: CNN-based methods: [13,22,28,48] and LSTM-based methods: [9,20]. The first category employs convolution and pooling operations to directly extract the abstract feature representation. The second category uses LSTM to construct short-distance and long-distance spatial dependencies.

Since ADE20K dataset is a newly proposed large-scale scene parsing dataset, all the results in Table 4 are given based on our own implementation. All of the comparison methods have trained 200,000 iterations without any extra data. According to Table 4, the proposed PDNs outperform the baseline method DeepLab-v2(Res101) [22] in terms of pixel accuracy and mean IoU with 2.50% and 2.64% respectively. Compared with existing LSTM-based methods, such as LG-LSTM(Res101) [9], our best configuration is still able to achieve 1.3% promotion in terms of mean IoU. It well demonstrates that incorporating the contextual information from the larger range into representation learning can further enhance the discriminative ability of deep features. Fig. 5 gives the visualization results of our method, DeepLab-v2(Res101) and LG-LSTM(Res101). It is obvious that PDNs can effectively distinguish similar objects in complex scenes by local and global reasoning. For fair comparison with state-of-the-art methods, we enlarge the batchsize (i.e. 16 images) of proposed method by using BN synchronization in the training phase. According to Table 4, our method can achieve comparable segmentation accuracy compared with PSPNet [48]. It even obtains slightly higher pixel accuracy than PSPNet over all of the categories.

To strictly evaluate the effectiveness of using different numbers of spatial neighbors, we also report the performance of using 8 spatial neighbors and 24 spatial neighbors. We use the capital 'NB' to indicate the number of spatial neighboring connections for each site. According to Table 4, 'PDNs-24NB' with 24 spatial neighbors outperforms that with 8 neighbors by 0.84% over the metric of mIoU by using single GPU. By applying synchronized cross-GPU batch normalization, the gap has further enlarged to 1.12%. These results is in agreement with the experiment about valid receptive field. Additionally, we also find that embedding multiple LSTM modules at different depths of the network can further improve the performance of the model.

PASCAL-Person-Part dataset [23]. Table 5 shows the comparison results with ten approaches [9,13,19–22,45,46,48,51] on the metric of mean IoU. An obvious improvement, i.e. 2.5% increase by PDNs over the DeepLab-v2(Res101) [22], can be observed from the comparison on breakdown categories. In order to reflect the ad-

Table 5
Experimental results (**IoU**) on PASCAL-Person-Part set. † indicates BN synchronization.

Method	head	torso	u-arms	l-arms	u-legs	l-legs	bkg.	mIoU
DeepLab [13]	78.1	54.0	37.3	36.9	33.7	29.6	92.9	51.8
HAZN [45]	80.8	59.1	43.1	42.8	39.0	34.5	93.6	56.1
Attention [46]	–	–	–	–	–	–	–	56.4
ResNet [7]	85.1	68.4	50.9	51.2	46.7	39.4	95.6	62.4
ResNet + CRF [7]	84.2	68.8	51.2	52.2	46.8	39.4	95.3	62.6
Deeplab-v3†(Res101) [51]	87.7	72.7	57.1	58.1	53.4	**51.7**	97.6	68.3
PSPNet†(Res101) [48]	88.2	71.4	**57.9**	**58.9**	52.8	50.5	98.3	**68.6**
Grid-LSTM [20]	81.9	58.9	43.1	46.9	40.1	34.6	86.0	56.0
Diagonal BiLSTM [21]	82.7	60.6	45.0	47.6	42.0	37.3	88.1	57.6
LG-LSTM [9]	82.7	61.0	45.4	47.8	42.3	38.0	88.6	58.0
Graph-LSTM [19]	82.7	62.7	46.9	47.7	45.7	40.9	94.6	60.2
LG-LSTM (Res101) [9]	86.0	69.1	52.8	52.0	47.3	43.1	95.2	63.6
PDNs-8NB-1Module	85.7	68.4	52.6	51.9	46.6	42.9	95.3	63.3
PDNs-24NB-1Module	85.5	69.9	53.3	53.1	47.4	43.6	95.5	64.0
PDNs-24NB-2Modules	85.7	70.1	53.8	54.0	49.0	45.7	96.2	64.9
PDNs-24NB-1Module†	87.1	73.9	56.9	56.7	52.0	51.4	98.0	67.9
PDNs-24NB-2Modules†	**88.4**	**74.5**	56.1	57.1	**53.5**	50.1	**98.6**	68.3

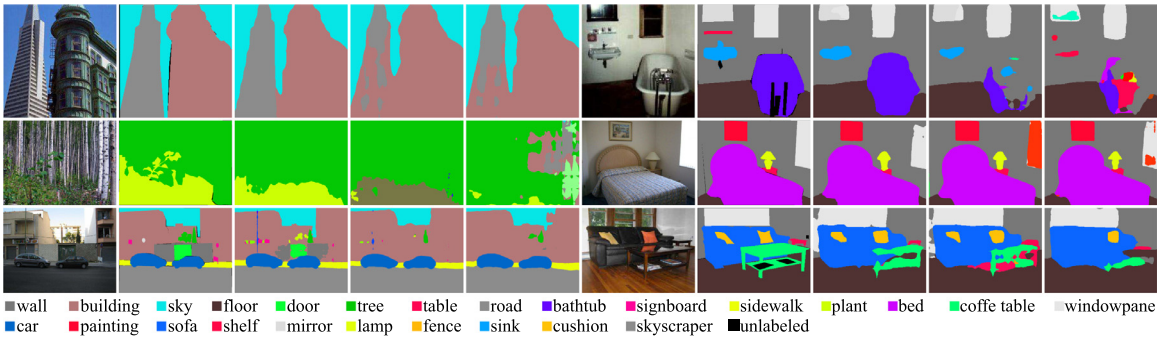

Fig. 5. Visualization of parsing results on ADE20K MIT dataset [8]. From left to right are input image, groundtruth labeling, parsing result by proposed PDNs, by LG-LSTM (Res101) [9] and by DeepLab-v2 (Res101) [22]. All of the models are trained on single GPU. Best viewed in color.

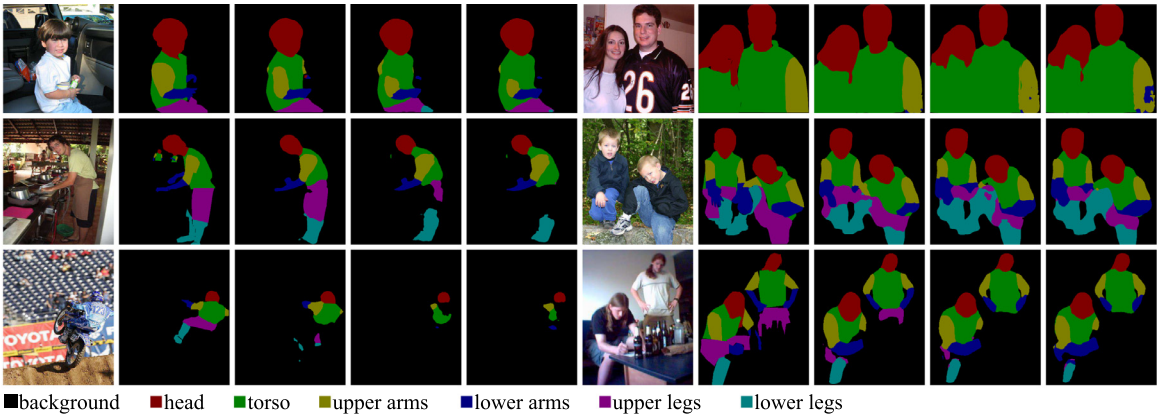

Fig. 6. Visualization of parsing results on PASCAL-Person-Part dataset [23]. From left to right are input image, groundtruth labeling, parsing result by proposed PDNs, by LG-LSTM (Res101) [9] and by DeepLab-v2 (Res101) [22]. All of the models are trained on single GPU. Best viewed in color.

vantages of proposed PDNs in context modeling, we implement the LG-LSTM [9] by adopting Resnet101 [7] as the bottom neural networks. Such model achieves mean IoU of 63.6%, which is still 1.3% less than our best model. When the model is trained by using BN synchronization, the mIoU of our method is 68.3%, which is comparable with Deeplab-v3 [51] and PSPNet [48]. The visualization of parsing results on this dataset are shown in Fig. 6.

6. Conclusion

This work presented Progressively Diffused Network (PDN) to model the information propagation on the image plane. By adopting convolutional LSTM with special atrous filters, the stacked diffusion layers make the context information from a certain site spreading to a large range of the image in a few steps, which effectively enlarge the valid receptive field of each site. This work has demonstrated diffused layer is a simple yet effective module for context propagation, where PDN outperforms previous LSTM based methods without bells and whistles.

Future work will explore PDN in many other dense predication tasks, such as instance object segmentation, crowd flow estimation and so on. Moreover, how to accelerate PDN to deal with real-time application is still an open issue. At last, combining proposed method with other techniques, such as knowledge graph, is also an exciting research direction.

Acknowldgements

This work was supported in part by the National Key Research and Development Program of China under grant no. 2018YFC0830103, in part by National High Level Talents Special Support Plan (Ten Thousand Talents Program), in part by National Natural Science Foundation of China (NSFC) under grant no. 61622214, and 61836012, in part by the Ministry of Public Security Science and Technology Police Foundation project no. 2016GABJC48.

References

[1] Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.
[2] L. Yann, B. Yoshua, H. Geoffrey, Deep learning, Nature 521 (2015) 436–444.
[3] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, ICLR, 2015.
[4] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, NIPS, 2012.
[5] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, ICLR (2015).
[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, CVPR, 2015.
[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, CVPR (2016).
[8] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba, Semantic understanding of scenes through the ADE20k dataset, Arxiv Preprint (2016).
[9] X. Liang, X. Shen, D. Xiang, J. Feng, L. Lin, S. Yan, Semantic object parsing with local-global long short-term memory, CVPR (2016).
[10] V. Koltun, Efficient inference in fully connected CRFS with Gaussian edge potentials, 2011.
[11] V. Vineet, J. Warrell, P.H. Torr, Filter-based mean-field inference for random fields with higher-order terms and product label-spaces, Int. J. Comput. Vis. 110 (3) (2014) 290–307.
[12] J. Yang, B. Price, S. Cohen, M.-H. Yang, Context driven scene parsing with attention to rare classes, CVPR, 2014.
[13] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, Semantic image segmentation with deep convolutional nets and fully connected crfs, ICLR, 2014.
[14] Z. Liu, X. Li, P. Luo, C.C. Loy, X. Tang, Semantic image segmentation via deep parsing network, ICCV, 2015.
[15] A.G. Schwing, R. Urtasun, Fully connected deep structured networks, ArXiv Preprint (2015).

[16] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, P.H.S. Torr, Conditional random fields as recurrent neural networks, ICCV, 2015.
[17] W. Byeon, T.M. Breuel, F. Raue, M. Liwicki, Scene labeling with lstm recurrent neural networks, CVPR, 2015.
[18] L. Theis, M. Bethge, Generative image modeling using spatial lstms, NIPS, 2015.
[19] X. Liang, X. Shen, J. Feng, L. Lin, S. Yan, Semantic object parsing with graph LSTM, ECCV, 2016.
[20] N. Kalchbrenner, I. Danihelka, A. Graves, Grid long short-term memory, Arxiv Preprint (2015).
[21] A. van den Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks, ICML, 2016.
[22] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, ArXiv Preprint (2016).
[23] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, A. Yuille, Detect what you can: detecting and representing objects using holistic models and body parts, CVPR, 2014.
[24] J. Yang, K. Yu, Y. Gong, T. Huang, Linear spatial pyramid matching using sparse coding for image classification, CVPR, 2009.
[25] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1627–1645.
[26] P. Schnitzspan, M. Fritz, S. Roth, B. Schiele, Discriminative structure learning of hierarchical representations for object detection, CVPR, 2009.
[27] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput 1 (4) (1989) 541–551.
[28] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, CVPR, 2015.
[29] S. Ding, L. Lin, G. Wang, H. Chao, Deep feature learning with relative distance comparison for person re-identification, Pattern Recognit. 48 (10) (2015) 2993–3003.
[30] S. Ji, W. Xu, M. Yang, K. Yu, 3D convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 221–231.
[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, CVPR, 2015.
[32] R.K. Srivastava, K. Greff, J. Schmidhuber, Highway networks, ArXiv Preprint (2015).
[33] L. Dong, N. Feng, Q. Zhang, Lsi: latent semantic inference for natural image segmentation, Pattern Recognit. 59 (2016) 282–291.
[34] D. Ravì, M. Bober, G.M. Farinella, M. Guarnera, S. Battiato, Semantic segmentation of images exploiting DCT based features and random forest, Pattern Recognit. 52 (2016) 260–273.
[35] R. Zhang, L. Lin, G. Wang, M. Wang, W. Zuo, Hierarchical scene parsing by weakly supervised learning with image descriptions, IEEE Trans. Pattern Anal. Mach. Intell. (2018).
[36] Y. Wang, J. Liu, Y. Li, J. Fu, M. Xu, H. Lu, Hierarchically supervised deconvolutional network for semantic video segmentation, Pattern Recognit. 64 (2017) 437–445.
[37] Z. Zhang, F. Xing, H. Wang, Y. Yan, Y. Huang, X. Shi, L. Yang, Revisiting graph construction for fast image segmentation, Pattern Recognit. 78 (2018) 344–357.
[38] A. Graves, J. Schmidhuber, Offline handwriting recognition with multidimensional recurrent neural networks, NIPS, 2008.
[39] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, NIPS, 2014.
[40] K. Xu, J. Ba, R. Kiros, K. Cho, A.C. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, ICML, 2015.
[41] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, W. Woo, Convolutional LSTM network: a machine learning approach for precipitation nowcasting, NIPS, 2015.
[42] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, Y. Chen, Learning contextual dependence with convolutional hierarchical recurrent neural networks, IEEE Trans. Image Process. 25 (7) (2016) 2983–2996.
[43] G.Y. Chen, G. Núñez, Inflammasomes in intestinal inflammation and cancer, Gastroenterology 141 (6) (2011) 1986–1999.
[44] J.L. Long, N. Zhang, T. Darrell, Do convnets learn correspondence? NIPS, 2014.
[45] F. Xia, P. Wang, L.-C. Chen, A.L. Yuille, Zoom better to see clearer: huamn part segmentation with auto zoom net, ArXiv Preprint (2015).
[46] L.-C. Chen, Y. Yang, J. Wang, W. Xu, A.L. Yuille, Attention to scale: scale-aware semantic image segmentation, ArXiv Preprint (2015).
[47] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R.B. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, ACM Multimedia, 2014.
[48] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, CVPR, 2017.
[49] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Object detectors emerge in deep scene CNNS, ICLR, 2015.
[50] W. Luo, Y. Li, R. Urtasun, R. Zemel, Understanding the effective receptive field in deep convolutional neural networks, NIPS, 2016.
[51] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam, Rethinking atrous convolution for semantic image segmentation, arXiv preprint (2017).