Spatial Pyramid Based Graph Reasoning for Semantic Segmentation

Xia Li[1,2,4,*] Yibo Yang[3,4,*] Qijie Zhao[5] Tiancheng Shen[3,4] Zhouchen Lin[4] Hong Liu[2,✉]

[1] Zhejiang Lab

[2] Key Laboratory of Machine Perception, Shenzhen Graduate School, Peking University

[3] Academy for Advanced Interdisciplinary Studies, Peking University

[4] Key Laboratory of Machine Perception (MOE), School of EECS, Peking University

[5] Wangxuan Institute of Computer Technology, Peking University

{ethanlee,ibo,zhaoqijie,tianchengshen,zlin,hongliu}@pku.edu.cn

Abstract

The convolution operation suffers from a limited receptive filed, while global modeling is fundamental to dense prediction tasks, such as semantic segmentation. In this paper, we apply graph convolution into the semantic segmentation task and propose an improved Laplacian. The graph reasoning is directly performed in the original feature space organized as a spatial pyramid. Different from existing methods, our Laplacian is data-dependent and we introduce an attention diagonal matrix to learn a better distance metric. It gets rid of projecting and re-projecting processes, which makes our proposed method a light-weight module that can be easily plugged into current computer vision architectures. More importantly, performing graph reasoning directly in the feature space retains spatial relationships and makes spatial pyramid possible to explore multiple long-range contextual patterns from different scales. Experiments on Cityscapes, COCO Stuff, PASCAL Context and PASCAL VOC demonstrate the effectiveness of our proposed methods on semantic segmentation. We achieve comparable performance with advantages in computational and memory overhead.

1. Introduction

Convolutional Neural Networks (CNNs) based architectures have revolutionized a wide range of computer vision tasks [20, 48, 5, 38]. Despite the huge success, convolutional operations suffer from a limited receptive filed, so they can only capture local information. Only with layers stacked as a deep model, can convolution networks have the ability to aggregate rich information of global context. However, it is an inefficient way since stacking local cues cannot always precisely handle long-range context relationships. Especially for pixel-level classification problems, such as semantic segmentation, performing long-range interactions is an important factor for reasoning in complex scenarios [5, 6]. For examples, it is prone to assign visually similar pixels in a local region into the same category. Meanwhile, pixels of the same object but distributed with a distance are difficult to construct dependencies.

Several approaches have been proposed to address the problem. Convolutional operations are reformulated with dilation [51] or learnable offsets [12] to augment the spatial sampling locations. Non-local network [46] and double attention network [9] try to introduce new interaction modules that sense the whole spatial-temporal space. They enlarge the receptive region and enable capturing long-range dependencies within deep neural networks. Recurrent neural networks (RNNs) can also be employed to perform long-range reasoning [16, 43]. However, these methods learn global relationships implicitly and rely on dense computation. Because graph-based propagation has the potential benefits of reasoning with explicit semantic meaning stored in graph structure, graph convolution [24] recently has been introduced into high-level computer vision tasks [28, 29, 10]. These methods first transform the grid-based CNN features into graph representation by projection, and then perform graph reasoning with graph convolution proposed in [24]. Finally, these node features are re-projected back into the original space. The projection and re-projection processes try to build connections between coordinate space and interaction space, but introduce much computation overhead and damage the spatial relationships.

As illustrated in Figure 1, in this paper, we propose an improved Laplacian formulation for graph reasoning that is directly performed in the original CNN feature space organized as a spatial pyramid. It gets rid of projection and re-projection processes, making our proposed method a light-weight module jointly optimized with the network

[*] : Equal contribution.

training. Performing graph reasoning directly in the original feature space retains the spatial relationships and makes spatial pyramid possible to sufficiently exploit long-range semantic context from different scales. We name our proposed method as **S**patial **Py**ramid Based **G**raph **R**easoning (SpyGR) layer.

Initially, graph convolution was introduced to extract representation in non-Euclidean space, which cannot be handled well by current CNN architectures [2]. It seems that graph propagation should be performed on graph-structured data, which motivates the construction of semantic interaction space in [28, 29, 10]. Actually we note that image features can be regarded as a special case of data defined on a simple low-dimensional graph [21]. When the graph structure of input is known, *i.e.*, the Laplacian matrix L is given, the graph convolution [24] essentially performs a special form of Laplacian smoothing on the input, making each new vertice feature as the average of itself and connected neighbors [26]. But for the case that graph structure is not given, as seen in CNN features, the graph structure can be estimated with the similarity matrix from the data [21], which achieves a similar goal with the projection process adopted in [28, 29, 10]. Different from their work where the Laplacian is a learnable data-independent matrix, in this study, we modify the Laplacian as a data-dependent similarity matrix, and introduce a diagonal matrix that performs channel-wise attention on the inner product distance. The Laplacian ensures that the long-range context pattern to learn is dependent on the input features and not restricted as a specific one. Our method spares the computation to construct an interaction space by projecting. More importantly, it retains the spatial relationships to facilitate exploiting long-range context from multi-scale features.

Spatial pyramid contains multi-scale contextual information that is important for dense prediction tasks [51, 56, 35]. For graph-structured data, multi-scale scheme is also the key to build hierarchical representation and enable the model to be invariant with scale changes [49, 32]. Global context owns multiple long-range contextual patterns that can be better captured from features of different sizes. The finer representation has more detailed long-range context, while the coarser representation could provide more global relationships. Because our method is able to perform graph reasoning directly in the original feature space, it is possible to build a spatial pyramid to further extend the long-range contextual patterns that our method can model.

The SpyGR layer is light-weight and can be plugged into CNN architectures easily. It efficiently extracts long-range context without introducing much computational overhead. The contributions in this study are listed as follows:

- We propose an improved Laplacian formulation that is data-dependent, and introduce a diagonal matrix with position-agnostic attention on the inner product to enable a better distance metric.
- The Laplacian is able to perform graph reasoning in the original feature space, and makes spatial pyramid possible to capture multiple long-range contextual patterns. We develop a computing scheme that effectively reduces the computational overhead.
- Experiments on multiple datasets, including PASCAL Context, PASCAL VOC, Cityscapes and COCO Stuff, show the effectiveness of our proposed methods for the semantic segmentation task. We achieve top performance with advantages in computational and memory overhead.

2. Related Work

Semantic segmentation. Fully convolutional network (FCN) [38] has been the basis of semantic segmentation with CNNs. Because details are important for dense classification problems, different methods are proposed to generate desired spatial resolution and keep object details. In [40], deconvolution [52] is employed to learn finer representation from low-resolution feature maps, while SegNet [1] achieves this purpose using an encoder-decoder structure. U-Net [41] adds a skip connection between the down-sampling and up-sampling paths. RefineNet [34] introduce a multi-path refinement network that further exploits the finer information along the down-sampling path.

Another stream aims to enhance multi-scale contextual information aggregation. In [17], input images are constructed as a Laplacian pyramid and each scale is fed into a deep CNN model. ParseNet [36] introduces image-level features to augment global context. DeepLabv2 [5] proposes the atrous spatial pyramid pooling (ASPP) module that consists of parallel dilated convolutions with variant dilation rates. PSPNet [56] performs spatial pyramid pooling to collect contextual information of different scales. DeepLabv3 [6] employs ASPP module on image-level features to better aggregate global context.

Other methods that model global context include formulating advanced convolution operations [12, 46, 9], relying on attention mechanisms [7, 53, 57, 18], and introducing Conditional Random Field (CRF) [4, 58, 37] or RNN variants [30, 16, 43] to build long-range dependencies. Still, it needs further efforts to explore how to model global context more efficiently, and perform reasoning explicitly with the semantic meanings.

Graph convolution. Graph convolution was initially introduced as a graph analogue of the convolutional operation [2]. Later studies [13, 24] make approximations on the graph convolution formulation to reduce the computational cost and training parameters. It provides the basis of feature embedding on graph-structured data for semi-supervised learning [24, 26], node or graph classification [44, 49, 54], and molecule prediction [27]. Due to the

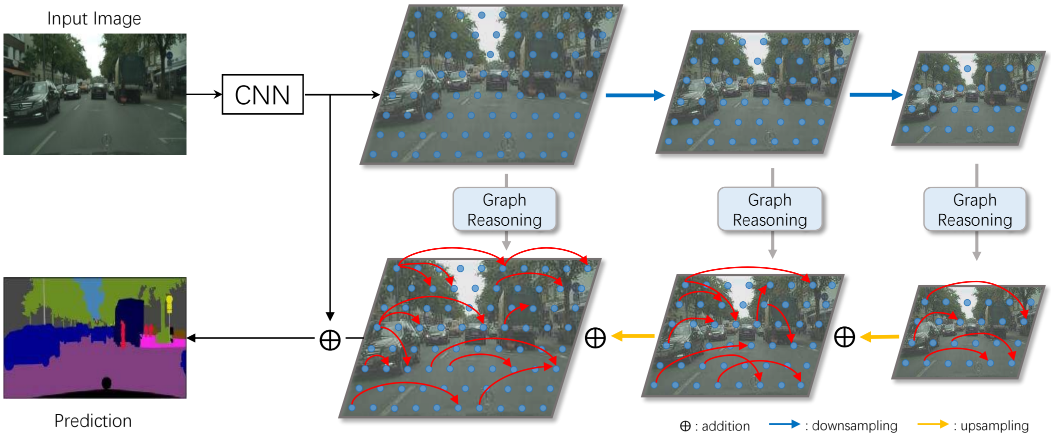


Figure 1: A diagram of our model with graph reasoning on spatial pyramid for the semantic segmentation task. The graph reasoning is directly performed in the original feature space. Multiple long-range contextual patterns are captured from different scales.

ability of capturing global information in graph propagation, the graph reasoning is introduced for visual recognition tasks [28, 29, 10]. These methods transform the grid-based feature maps into region-based node features via projection. Different from these studies, our method notes that the graph reasoning can be directly performed in original feature space, once the learnable Laplacian matrix is data dependent. It spares the computation of projection and re-projection, and retains the spatial relationships in the graph reasoning process.

Feature pyramid. Feature pyramid is an effective scheme to capture multi-scale context. It is widely adopted in dense prediction tasks such as semantic segmentation [5, 56] and object detection [35, 19]. Hierarchical representation is also shown to be useful for embedding on graph-structured data [49]. Different from the pyramid pooling module in [5], we build our spatial pyramid simply by down-sampling and up-sampling processes on the final predicting feature maps. We directly perform graph reasoning on each of the scale and aggregate them in order to capture sufficient long-range contextual relationships in the final prediction.

3. Our Methods

In this section, we first briefly introduce the background of graph convolution, and then develop our method in detail. Finally, we analyze the complexity of our method.

3.1. Graph Reasoning on Graph Structures

Graph convolution was introduced as an analogue of convolutional operation on graph-structured data. Given graph $\mathcal{G} = (V, E)$ and its adjacency matrix A and degree matrix D, the normalized graph Laplacian matrix L is defined as: $L = I - D^{-1/2}AD^{-1/2}$. It is a symmetric positive semi-definite matrix and has a complete set of eigenvectors U formed by $\{u_s\}_{s=0}^{N-1}$, where N is the number of vertices. The Laplacian of graph $\mathcal{G}$ can be diagonalized as $L = U\Lambda U^T$. Then we have graph Fourier transform $\hat{x} = U^T x$, which transforms the graph signal x into spectral domain spanned by basis U.

Generalizing the convolution theorem into structured space on graph, convolution can be defined through decomposing a graph signal $\mathbf{s} \in \mathbb{R}^n$ on the spectral domain and then applying a spectral filter g_θ [2]. Naive implementation requires explicitly computing the Laplacian eigenvectors. To circumvent this problem, later study [13] approximated the spectral filter $g_\theta(\Lambda)$ with Chebyshev polynomials up to K^{th} order, *i.e.*, $g_\theta(\Lambda) \approx \sum_{k=0}^{K} \theta_k T_k(\Lambda)$, and then convolution of the graph signal can be formulated as:

$$g_\theta \star \mathbf{s} = \sum_{k=0}^{K} \theta_k T_k(L)\mathbf{s}, \tag{1}$$

where T_k is the Chebyshev polynomials and $\{\theta_k\}$ is a vector of Chebyshev coefficients. In [24], the formulation is further simplified by limiting $K = 1$, and approximating the largest eigenvalue of L by 2. In this way, the convolution becomes:

$$g_\theta \star \mathbf{s} = \theta \left(I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) \mathbf{s}, \tag{2}$$

with θ being the only Chebyshev coefficient left. They further introduce a normalization trick:

$$I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \to \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \tag{3}$$

where $\tilde{A} = A + I$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Generalizing the convolution to a graph signal with c channels, the layer-wise propagation rule in a multi-layer graph convolutional network (GCN) is given by [24]:

$$H^{(l+1)} = \sigma \left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} \Theta^{(l)} \right) \tag{4}$$

where $H^{(l)}$ is the vertices features of the l-th layer, $\Theta^{(l)}$ is the trainable weight matrix in layer l, and σ is the non-linear activation function.

The Eq (4) provides the basis of performing convolution on graph-structured data, as adopted in [54, 49]. For visual recognition tasks, in order to overcome the limited receptive field in current CNN architectures, some recent studies transform feature maps into region-based representation by projecting, and then perform graph reasoning with Eq (4) to capture global relationships [28, 29, 10].

3.2. Graph Reasoning on Spatial Features

Assuming that the propagation rule in Eq (4) is applied on CNN features, *i.e.*, $H^{(l)} = X^{(l)} \in \mathbb{R}^{H \times W \times C}$, the only difference between a GCN layer and a convolution layer is the graph Laplacian matrix $\tilde{L} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$ applied on the left of $X^{(l)}$. In our study, we note that the original grid-based feature space can be deemed as a special case of data defined on a simple low-dimensional graph [21]. Besides, the projecting process in current methods [28, 29, 10] actually achieves a similar purpose with the graph Laplacian matrix. They perform left multiplication on the input feature using a similarity matrix to have a global perception among all spatial locations. Therefore, we directly perform our graph reasoning in the original feature space. We save the projecting and re-projecting processes, and perform left matrix multiplication on the input feature only once.

The Laplacian matrices in most current studies are data-independent parameters to learn. In order to better capture intra spatial structure, we propose an improved Laplacian $\tilde{L}$ that ensures the long-range context pattern to learn is dependent on the input features and not restricted as a specific one. It is formulated with the symmetric normalized form:

$$\tilde{L} = I - \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tag{5}$$

where $\tilde{D} = diag(d_1, d_2, \ldots, d_n)$, $d_i = \sum_j \tilde{A}_{ij}$, and $\tilde{A} \in \mathbb{R}^{n \times n}$ is the data-dependent similarity matrix. We set $n = H \times W$, where $H \times W$ denotes the number of spatial locations of the input feature.

For similarity matrix $\tilde{A}$, Euclidean distance can be used to estimate the graph structure as suggested in [21]. We choose dot-product distance to calculate A, because dot product has a more friendly implementation in current deep learning platforms. The similarity between position i and j is expressed as:

$$\tilde{A}_{ij} = \phi(X)_i \tilde{\Lambda}(X) \phi(X)_j^T, \tag{6}$$

where $\phi(X) \in \mathbb{R}^{HW \times M}$ is a linear embedding followed by ReLU$(\cdot)$ non-linearity, M is the reduced dimension after transformation, and $\tilde{\Lambda}(X) \in \mathbb{R}^{M \times M}$ is a diagonal matrix that has position-agnostic attention on the inner product. It essentially learns a better distance metric for the similarity matrix $\tilde{A}$. Both $\phi(X)$ and $\tilde{\Lambda}(X)$ are data-dependent. Concretely, $\phi(X)$ is implemented as a 1×1 convolution, and $\tilde{\Lambda}(X)$ is implemented in a similar way as the channel-wise attention proposed in [22]. We calculate $\tilde{\Lambda}(X)$ as:

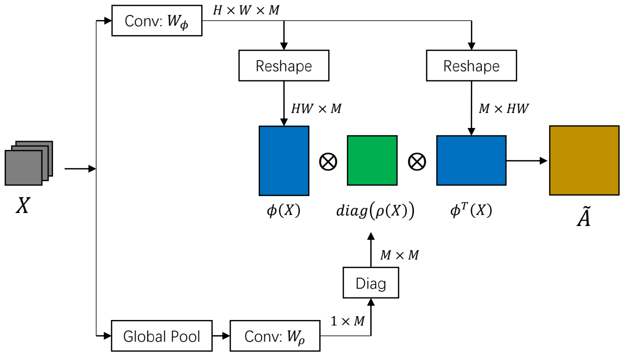


Figure 2: The computation procedures of the similarity matrix $\tilde{A}$ from the input feature X.

$$\tilde{\Lambda}(X) = diag\left(\rho\left(\bar{X}\right)\right), \tag{7}$$

where $\bar{X} \in \mathbb{R}^{1 \times 1 \times C}$ is the feature after global pooling, and $\rho(\cdot)$ is another linear embedding with 1×1 convolution that reduce the dimension from C to M. It is followed by the sigmoid function.

The computation procedures of $\tilde{A}$ is shown in Figure 2, and we have its formulation as follows:

$$\tilde{A} = \phi\left(X; W_\phi\right) diag\left(\rho\left(\bar{X}; W_\rho\right)\right) \phi\left(X; W_\phi\right)^T, \tag{8}$$

where W_ϕ and W_ρ are learnable parameters for the linear transformations. Because the degree matrix $\tilde{D}$ in Eq (5) has a function of normalization, we do not perform softmax on the similarity matrix $\tilde{A}$. Then we formulate the graph reasoning in our model as:

$$Y = \sigma\left(\tilde{L} X \Theta\right), \tag{9}$$

where X is the input feature, Θ is a trainable weight matrix, σ is the ReLU activation function, and Y is the output feature.

3.3. Graph Reasoning on Spatial Pyramid

Although graph reasoning is capable of capturing global context, we note that the same image contains multiple long-range contextual patterns. For examples, the finer representation may have more detailed long-range context, while the coarser representation provide more global dependencies. Since our graph reasoning module is directly performed in the original feature space, we organize the input feature as a spatial pyramid to extend the long-range contextual patterns that our method can capture.

As shown in Figure 1, graph reasonings are performed on each scale acquired by down-sampling, and then the output features are combined through up-sampling. It has a similar form with the feature pyramid network in [35]. But we implement our method on the final predicting feature, instead of the multi-scale features from the CNN backbone. Our graph reasoning on spatial pyramid can be expressed as follows:

$$\begin{aligned} Y^{(s+1)} &= \mathrm{GR}(X^{(s+1)}) + \Pi_{\mathrm{up}}(Y^{(s)}), \\ Y^{(0)} &= \mathrm{GR}(X^{(0)}), \\ X^{(s)} &= \Pi_{\mathrm{down}}(X^{(s+1)}), \end{aligned} \tag{10}$$

where GR denotes the graph reasoning with Eq (9), $s \geq 0$ denotes the level of scales, and $\Pi_{\mathrm{up}}, \Pi_{\mathrm{down}}$ represents the up-sampling and down-sampling operators, respectively. We implement Π_{down} using max-pooling with stride of 2, and Π_{up} simply by bilinear interpolation.

3.4. Complexity Analysis

In region-based graph reasoning studies [28, 29, 10], they transform the grid-based CNN features into region-based vertices by projecting, which reduces the computational overhead for graph reasoning because the number of vertices is usually less than that of spatial locations. It seems that our method consumes more computation since we implement graph reasoning directly in the original feature space. Actually, we adopt an efficient computing strategy that successfully reduces the computational complexity of our method. We note that large computation is caused by the similarity matrix $\tilde{A} \in \mathbb{R}^{HW \times HW}$, therefore we do not explicitly calculate $\tilde{A}$. Concretely, we calculate the degree matrix $\tilde{D}$ in Eq (5) as follow:

$$\tilde{D} = diag\left(\tilde{A} \cdot \vec{1}\right) = diag\left(\phi\left(\tilde{\Lambda}\left(\phi^T \cdot \vec{1}\right)\right)\right) \tag{11}$$

where $\vec{1}$ denotes an all-one vector in $\mathbb{R}^{HW}$. The brackets indicate the computation superiority. In this way, each step in Eq (11) is a multiplication with a vector, which effectively reduces the computational overhead. And then we calculate the left product of the Laplacian on the input feature as follows:

$$\begin{aligned} \tilde{L}X &= X - \tilde{D}^{-\frac{1}{2}}\phi\tilde{\Lambda}\phi^T\tilde{D}^{-\frac{1}{2}}X \\ &= X - P\left(\tilde{\Lambda}\left(P^T X\right)\right) \end{aligned} \tag{12}$$

where P is defined as $P = \tilde{D}^{-\frac{1}{2}}\phi$. Correspondingly, we calculate the terms in inner bracket first. In this way, we circumvent quadratic order of computation on the spatial locations $\mathcal{O}(H^2W^2)$.

In our experiments, we set C as 512, and M as 64. Assuming that height H and width W of the input features are 97, we calculate the computational and memory cost of our proposed layer, and compare with related methods in the same settings. As shown in Table 1, for our method on single-scale input, it has low computational cost. When we have spatial pyramid on 4 scales, the computational and memory overheads do not show drastic increment. Therefore, our SpyGR layer does not introduce unbearable overhead in spite of its directly performing graph reasoning in the original feature space.

Method	FLOPs (G)	Memory (M)
Nonlocal [46]	14.60	1072
A^2Net [9]	3.11	110
GloRe [10]	3.11	103
SGR [29]	6.24	118
DANet [18]	19.54	1114
SpyGR w/o pyramid	3.11	120
SpyGR	4.12	164

Table 1: Overhead of different modules with input feature in $[1 \times 512 \times 97 \times 97]$. We show the complexity of our model on single-scale feature, and on a spatial pyramid with 4 scales in the bottom two rows of the table.

4. Experiments

4.1. Datasets and Implementation Details

To evaluate our proposed SpyGR layer, we carry out comprehensive experiments on the Cityscapes dataset [11], the PASCAL Context dataset [39] and the COCO Stuff dataset [3]. We describe these datasets, together with implement details and loss function as follows.

Implement Details. We use ResNet [20] (pretrained on ImageNet [14]) as our backbone. We use a 3×3 convolution to reduce the channel number from 2048 to 512, and then stack SpyGR layer upon it. We set M as 64 in all our experiments. Following prior works [56, 5, 6], we employ a polynomial learning rate policy where the initial learning rate is multiplied by $(1 - iter/\ total\_iter)^{0.9}$ after each iteration. Momentum and weight decay coefficients are set to 0.9 and 0.0001 respectively, and the base learning rate is set to 0.009 for all datasets. For data augmentation, we apply the common scale, cropping and flipping strategies to augment the training data. Input size is set as 769×769 for Cityscapes, and 513×513 for others. The synchronized batch normalization is adopted in all experiments, together with the multi-grid [6] scheme. For evaluation, we use the Mean IoU metric as a common choice. We downsample for three times and have four levels in our pyramid.

Loss Function. We employ the standard cross entropy loss on both final output of our model, and the intermediate feature map output from res4b22. We set the weight over the

Method	mIoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
Deeplabv2 [5]	70.4	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
RefineNet [34]	73.6	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
DUC-HDC [45]	77.6	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
SAC [55]	78.1	**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
DepthSeg [25]	78.2	98.5	85.4	92.5	54.4	60.9	60.2	72.3	76.8	93.1	71.6	94.8	85.2	69.0	95.7	70.1	86.5	75.7	68.3	75.5
PSPNet [56]	78.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
AAF [23]	79.1	98.5	85.6	93.0	53.8	59.0	65.9	75.0	78.4	93.7	72.4	95.6	86.4	70.5	95.9	73.9	82.7	76.9	68.7	76.4
DFN [50]	79.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PSANet [57]	80.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DenseASPP [47]	80.6	**98.7**	**87.1**	93.4	**60.7**	62.7	65.6	74.6	78.5	93.6	72.5	95.4	86.2	71.9	96.0	**78.0**	**90.3**	80.7	69.7	76.8
GloRe [10]	80.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DANet [18]	81.5	98.6	86.1	93.5	56.1	**63.3**	69.7	77.3	81.3	**93.9**	**72.9**	**95.7**	87.3	72.9	**96.2**	76.8	89.4	**86.5**	**72.2**	78.2
SpyGR	**81.6**	**98.7**	86.9	**93.6**	57.6	62.8	**70.3**	**78.7**	**81.7**	93.8	72.4	95.6	**88.1**	**74.5**	**96.2**	73.6	88.8	86.3	72.1	**79.2**

Table 2: Per-class results on Cityscapes testing set. Best results are marked in bold and the second best results are underlined. It is shown that SpyGR achieves the highest performance and has superiority in most categories.

final loss as 1 and the auxiliary loss as 0.4, following the settings in PSPNet [56].

4.2. Results on Cityscapes

We first compare our method with existing methods on the Cityscapes test set. To fairly compare with others, we train our SpyGR upon ResNet-101 with output stride as 8. Note that we only train on fine annotated data. We adopt the OHEM scheme [42] for final loss, and train the model for 80K iterations, with mini-batch size set as 8. For testing, we adopt multi-scale (0.75, 1.0, 1.25, 1.5, 1.75, 2.0) inference and flipping, and then submit the predictions to official evaluation server. Results are shown in Table 2. We can see that SpyGR shows superiority in most categories. SpyGR outperforms GloRe [10], the latest graph convolutional networks (GCN) based model, by 0.7 in mIoU. Moreover, SpyGR even outperforms DANet, a recently proposed self-attention based model, whose computation overhead and memory requirements are much higher than our proposed methods, as shown in Table 1.

4.3. Comparisons with DeepLabV3

DeepLabV3 [6] and DeepLabV3+ [8] report their results on Cityscapes by training on the *fine+coarse* set. In order to show the effectiveness of our proposed methods over them, we conduct detailed comparisons on both Cityscapes and PASCAL VOC. As shown in Table 3, SpyGR consistently has at least 1 mIoU gains over DeepLabV3. The advantages of SpyGR over DeepLabV3+ are more significant on PASCAL VOC than Cityscapes.

4.4. Results on COCO Stuff

For the COCO Stuff dataset, we train SpyGR with output stride of 8, and mini-batch size of 12. We train for 30K iterations on the COCO Stuff training set, around 40 epochs, which is much shorter than DANet's 240 epochs. Multiscale input and flipping are used for testing. The comparison on the COCO Stuff dataset is shown in Table 4. Similar to the other two datasets, our SpyGR also outperforms other methods performance on the COCO Stuff dataset. It has a comparable result with DANet, but shows a significant superiority over SGR.

4.5. Results on PASCAL Context

We carry out experiments on the PASCAL Context dataset to further evaluate the validity of our proposed SpyGR. We train our model with mini-batch size of 16 and output stride of 16, and inference with output stride of 8. To make SpyGR operated with the same stride during both training and inference phase, we upsample C5 from ResNet-101, and concatenate it with C3, which has an output stride of 8. A 3×3 convolution is appended over the concatenation of C3 and C5, and then we add our SpyGR layer. We optimize the whole network on training set of PASCAL Context for 15K iterations, around 48 epochs. As a comparison, DANet trains for 240 epoch, around 5 times of us. For evaluation on test set, we adopt the multiscale and flipping augmentations. We show the experimental results of PASCAL Context in Table 5. It is shown that even SpyGR with ResNet-50 as backbone achieves comparable performance with SGR on ResNet-101, and outperforms MSCI [33] on ResNet-152. Furthermore, SpyGR on ResNet-101 gains higher performance than SGR+, even though SGR+ is pre-trained on the COCO Stuff dataset.

Methods	Cityscapes			PASCAL VOC		
	Val		Test	Val		Test
	SS	MS	+Coarse	SS	MS	Finetune
DeepLabV3	78.3	79.3	81.3	78.5	79.8	-
DeepLabV3+	79.6	80.2	82.1	79.4	80.4	83.3
SpyGR	**79.9**	**80.5**	**82.3**	**80.2**	**81.2**	**84.2**

Table 3: Comparisons with DeepLabV3. **SS** means single scale, **MS** denotes multi-scale. **+Coarse** means training on *fine+coarse* set. **Finetune** means finetuning on the *trainval* set. To be fair, all results of compared methods are tested on their newest implementations.

Method	Backbone	mIoU (%)
RefineNet [34]	ResNet-101	33.6
CCL [15]	ResNet-101	35.7
DANet [18]	ResNet-50	37.2
DSSPN [31]	ResNet-101	37.3
SpyGR	ResNet-50	37.5
SGR [29]	ResNet-101	39.1
DANet [18]	ResNet-101	39.7
SpyGR	ResNet-101	**39.9**

Table 4: The comparison on the COCO Stuff test set.

And once again, SpyGR outperforms DANet by a small margin, but with much less computational overhead and memory cost, and a significantly shorter training scheduler.

4.6. Ablation Studies

We conduct ablation studies to explore how does each part of SpyGR contribute to the performance gain. We carry out all ablation experiments on Cityscapes over ResNet-50. For inference, we only use single-scale input image. The comparisons are listed in Table 6. We analyze each part of SpyGR as follow.

Simplest GCN. We consider the case without the attentional diagonal matrix. The similarity matrix $\tilde{A}$ reduces to:

$$\tilde{A} = \phi\left(X, W_\phi\right) \phi\left(X, W_\phi\right)^T . \tag{13}$$

Removing the identity in Laplacian, the propagation rule of graph reasoning in Eq (9) now becomes as follow:

$$Y = \sigma\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta\right) . \tag{14}$$

The simplest GCN brings an increase of 1.64 in mIoU.

With data-independent $\tilde{\Lambda}$. Corresponding to Eq (6), we now introduce a diagonal matrix into the inner product of ϕ and ϕ^T to have a better distance metric. However, we make the diagonal matrix $\tilde{\Lambda}$ feature independent, which means that it is a vector of parameters to learn. It outperforms the simplest GCN by 0.60. We can see that the diagonal matrix indeed makes a better distance metric with only a few trainable parameters, and leads to a higher performance.

Method	Backbone	mIoU (%)
PSPNet [56]	ResNet-101	47.8
DANet [18]	ResNet-50	50.1
MSCI [33]	ResNet-152	50.3
SpyGR	ResNet-50	50.3
SGR [29]	ResNet-101	50.8
CCL [15]	ResNet-101	51.6
EncNet [53]	ResNet-101	51.7
SGR+ [29]	ResNet-101	52.5
DANet [18]	ResNet-101	52.6
SpyGR	ResNet-101	**52.8**

Table 5: The comparison on the test set of PASCAL Context. '+' means pretrained on COCO Stuff.

FCN	GCN	$\tilde{\Lambda}$	$\tilde{\Lambda}(X)$	Identity	Pyramid	mIoU
✓	-	-	-	-	-	76.34
✓	✓	-	-	-	-	77.98
✓	✓	✓	-	-	-	78.58
✓	✓	✓	✓	-	-	79.05
✓	✓	✓	✓	✓	-	79.42
✓	✓	✓	✓	✓	✓	79.93

Table 6: Ablation experiments on the Cityscapes dataset.

With data-dependent $\tilde{\Lambda}(X)$. In this case, we calculate $\tilde{A}$ using Eq (6), and the attention diagonal matrix becomes data-dependent by Eq (7). This mechanism works in a way similar to soft-attention. As a result, It further has a performance gain of 0.47 on mIoU over the data-independent case. It is demonstrated that the attention diagonal matrix $\tilde{\Lambda}(X)$ is more representative, and provides a better distance metric conditioned on the distribution of input features.

Identity. Now we recover the identity term in the Laplacian formulation, and calculate $\tilde{L}$ exactly following Eq (5). The identity term also plays a role of shortcut connection to facilitate optimization of graph reasoning. We see that the performance has a further increment.

Spatial Pyramid. Finally, we organize the input feature as a spatial pyramid following Eq (10), which enables capturing multiple long-range contextual patterns from different scales. It further brings a performance gain of 0.51 in mIoU.

4.7. Analysis

In order to have a better sense of the effects of our proposed spatial pyramid based graph reasoning, we visualize the similarity matrix in different scales on the Cityscapes

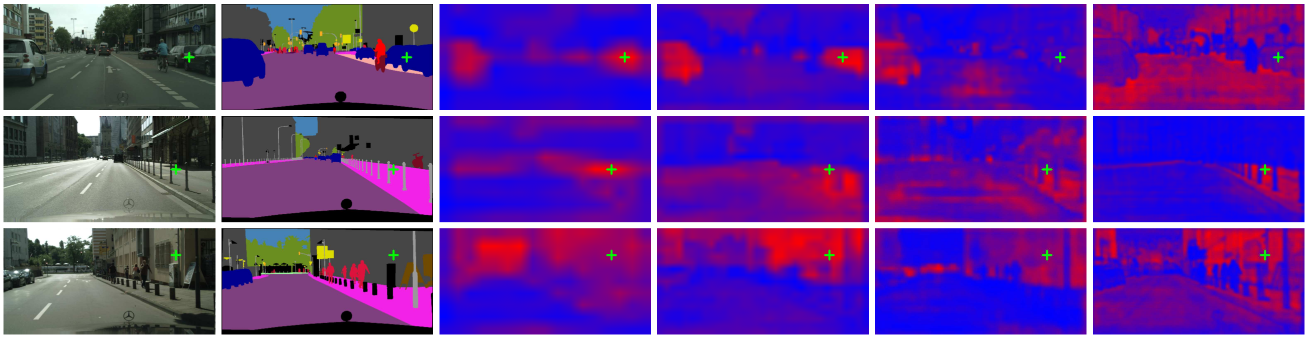
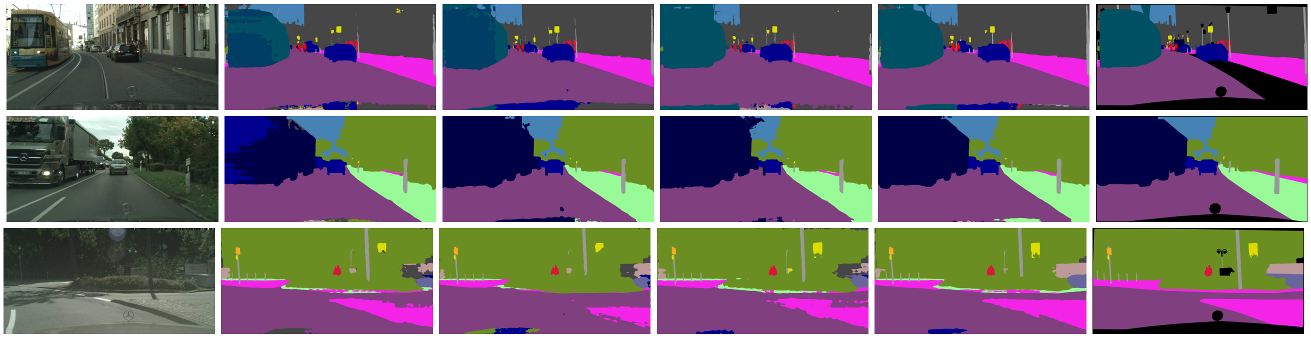


Figure 3: Visualization of the similarity matrix $\tilde{A}_i$ of a randomly sampled location i marked in green cross. The left two columns are input images and ground truths respectively. The similarity matrix of different scales in the pyramid are re-scaled in the same size and shown in the right four columns from the coarsest to the finest (from left to right). Multiple long-range contextual patterns are captured in different scales, and aggregated in the finest level. Zoom in to have a better view.



(a) Image (b) FCN (c) ASPP (d) PSP (e) SpyGR (f) Label

Figure 4: Visualisation comparison with other methods.

dataset. Concretely, as shown in Figure 3, we randomly generate a sampling point i and mark it by the green cross. And then we visualize the i-th row of the similarity matrix, *i.e.*, $\tilde{A}_i \in \mathbb{R}^{H\times W}$, as a heatmap. The right four columns show the similarity matrix from the coarsest level to the finest level. We can observe that, different long-range contextual patterns are captured in the spatial pyramid. For the sampling points located on the car, the strongest activations of the four scales are distributed on different cars. These different long-range relationships are finally aggregated into the finest level for prediction. This also happens to other categories such as sidewalk, bus and vegetation. For the sampling points located on the boundary line of two semantic categories, the interactions in different scales help to better assign the pixel into the right category. The aforementioned analysis shows that our proposed spatial pyramid is able to aggregate rich semantic information and capture multiple long-range contextual patterns. We also show the visualisation comparison with other methods in Figure 4.

5. Conclusion

In this paper, we aim to model long-range context using graph convolution for the semantic segmentation task. Different from current methods, we perform our graph reasoning directly in the original feature space organized as a spatial pyramid. We propose an improved Laplacian that is data-dependent, and introduce an attention diagonal matrix on the inner product to make a better distance metric. Our method gets rid of projecting and re-projecting processes, and retains the spatial relationships that enables spatial pyramid. We adopt a computing scheme to reduce the computational overhead significantly. Our experiments show that each part of our design contributes to the performance gain, and we outperform other methods without introducing more computational or memory consumption.

6. Acknowledgement

Zhouchen Lin is supported by National Natural Science Foundation (NSF) of China (grant no.s 61625301 and 61731018), Major Scientific Research Project of Zhejiang Lab (grant no.s 2019KB0AC01 and 2019KB0AB02), Beijing Academy of Artificial Intelligence, and Qualcomm. Hong Liu is supported by NSF China (grant no. U1613209) and NSF Shenzhen (grant no. JCYJ20190808182209321).

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, 39(12):2481–2495, 2017.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *CVPR*, pages 1209–1218, 2018.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4):834–848, 2018.

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[7] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, pages 3640–3649, 2016.

[8] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pages 801–818, 2018.

[9] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. In *NIPS*, pages 350–359, 2018.

[10] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. *arXiv preprint arXiv:1811.12814*, 2018.

[11] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016.

[12] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.

[13] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pages 3844–3852, 2016.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[15] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multiscale aggregation for scene segmentation. In *CVPR*, pages 2393–2402, 2018.

[16] Heng Fan, Peng Chu, Longin Jan Latecki, and Haibin Ling. Scene parsing via dense recurrent neural networks with attentional selection. *arXiv preprint arXiv:1811.04778*, 2018.

[17] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *TPAMI*, 35(8):1915–1929, 2013.

[18] Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. *arXiv preprint arXiv:1809.02983*, 2018.

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[21] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[22] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, pages 7132–7141, 2018.

[23] Tsung-Wei Ke, Jyh-Jing Hwang, Ziwei Liu, and Stella X Yu. Adaptive affinity fields for semantic segmentation. In *ECCV*, pages 587–602, 2018.

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[25] Shu Kong and Charless C Fowlkes. Recurrent scene parsing with perspective understanding in the loop. In *CVPR*, pages 956–965, 2018.

[26] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*, 2018.

[27] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *AAAI*, 2018.

[28] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. In *NIPS*, pages 9245–9255, 2018.

[29] Xiaodan Liang, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P Xing. Symbolic graph reasoning meets convolutions. In *NIPS*, pages 1858–1868, 2018.

[30] Xiaodan Liang, Xiaohui Shen, Donglai Xiang, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with local-global long short-term memory. In *CVPR*, pages 3185–3193, 2016.

[31] Xiaodan Liang, Hongfei Zhou, and Eric Xing. Dynamic-structured semantic propagation network. In *CVPR*, pages 752–761, 2018.

[32] Renjie Liao, Zhizhen Zhao, Raquel Urtasun, and Richard S Zemel. Lanczosnet: Multi-scale deep graph convolutional networks. In *ICLR*, 2019.

[33] Di Lin, Yuanfeng Ji, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Multi-scale context intertwining for semantic segmentation. In *ECCV*, pages 603–619, 2018.

[34] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, pages 1925–1934, 2017.

[35] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017.

[36] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015.

[37] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *ICCV*, pages 1377–1385, 2015.

[38] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[39] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, pages 891–898, 2014.

[40] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *ICCV*, pages 1520–1528, 2015.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.

[42] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, pages 761–769, 2016.

[43] Bing Shuai, Zhen Zuo, Bing Wang, and Gang Wang. Scene segmentation with dag-recurrent neural networks. *TPAMI*, 40(6):1480–1493, 2018.

[44] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[45] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *WACV*, pages 1451–1460. IEEE, 2018.

[46] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018.

[47] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *CVPR*, pages 3684–3692, 2018.

[48] Yibo Yang, Zhisheng Zhong, Tiancheng Shen, and Zhouchen Lin. Convolutional neural networks with alternately updated clique. In *CVPR*, June 2018.

[49] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *NIPS*, pages 4805–4815, 2018.

[50] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. In *CVPR*, pages 1857–1866, 2018.

[51] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[52] Matthew D Zeiler, Graham W Taylor, Rob Fergus, et al. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, volume 1, page 6, 2011.

[53] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, pages 7151–7160, 2018.

[54] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *AAAI*, 2018.

[55] Rui Zhang, Sheng Tang, Yongdong Zhang, Jintao Li, and Shuicheng Yan. Scale-adaptive convolutions for scene parsing. In *ICCV*, pages 2031–2039, 2017.

[56] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017.

[57] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, pages 267–283, 2018.

[58] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *ICCV*, pages 1529–1537, 2015.