Heterogeneous Grid Convolution for Adaptive, Efficient, and Controllable Computation

Ryuhei Hamaguchi[1], Yasutaka Furukawa[2], Masaki Onishi[1], and Ken Sakurada[1]
[1] National Institute of Advanced Industrial Science and Technology (AIST)
[2] Simon Fraser University
ryuhei.hamaguchi@aist.go.jp, furukawa@sfu.ca, onishi@ni.aist.go.jp, k.sakurada@aist.go.jp

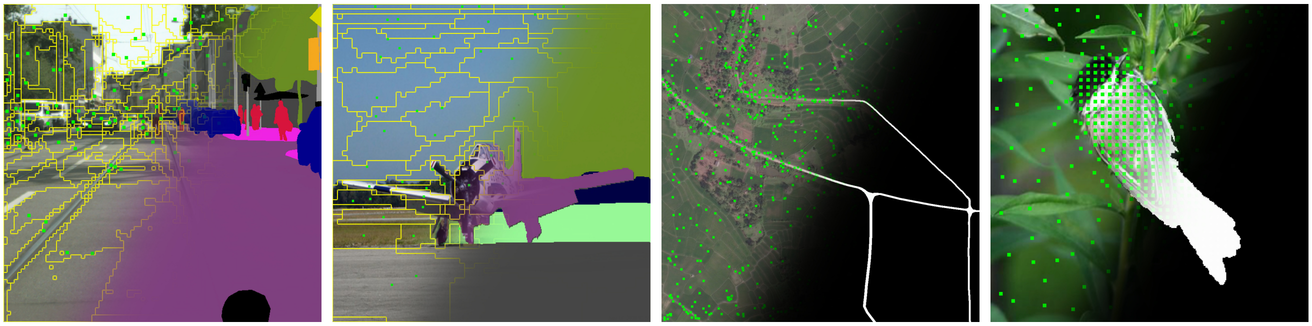


Figure 1. Heterogeneous grid convolution exploits the heterogeneity in the image to enable adaptive, efficient, and controllable computation for a range of image understanding tasks such as semantic segmentation, road extraction, and salient object detection from left to right.

Abstract

This paper proposes a novel heterogeneous grid convolution that builds a graph-based image representation by exploiting heterogeneity in the image content, enabling adaptive, efficient, and controllable computations in a convolutional architecture. More concretely, the approach builds a data-adaptive graph structure from a convolutional layer by a differentiable clustering method, pools features to the graph, performs a novel direction-aware graph convolution, and unpool features back to the convolutional layer. By using the developed module, the paper proposes heterogeneous grid convolutional networks, highly efficient yet strong extension of existing architectures. We have evaluated the proposed approach on four image understanding tasks, semantic segmentation, object localization, road extraction, and salient object detection. The proposed method is effective on three of the four tasks. Especially, the method outperforms a strong baseline with more than 90% reduction in floating-point operations for semantic segmentation, and achieves the state-of-the-art result for road extraction. We will share our code, model, and data.

1. Introduction

Our world is heterogeneous in nature. Looking at a scene from a car (See Fig. 1), the road occupies one third of the image with homogeneous textures. At the far end of the road are full of objects such as cars, pedestrians, or road signs. The density of semantic information varies per location. Our attention to the world is also heterogeneous in nature. With a specific task in mind, we focus our attention to a specific portion of an image, for example, tracing a road network in a satellite image.

While a regular grid feature representation has been successful, such a representation contains redundant information in low density regions, whereas the spatial resolution is insufficient in high density regions. Features should be stored adaptively based on the information density.

This paper studies a novel "heterogeneous grid convolution", which has the following three advantages. (*Adaptive*) The node features are adaptively allocated where necessary. (*Efficient*) The adaptive allocation reduces redundant computations. (*Controllable*) An agent can focus computation to a region of interest with an additional input.

There are two technical challenges in learning such a flexible feature representation: (a) how to adaptively allocate nodes while exploiting image heterogeneity, and (b) how to define a convolution operation on a heterogeneous grid structure. We propose a differentiable clustering-based graph pooling for the first challenge and a direction-aware extension of the graph convolution for the second challenge.

The combination of our graph pooling and direction-aware graph convolution forms a neural model, dubbed heterogeneous grid convolution, that can be inserted into any existing CNN architecture. By exploiting the proposed module, we also propose a heterogeneous grid convolutional neural networks (HG-CNNs) as a highly efficient yet strong extension of existing CNN architectures.

We have evaluated the proposed approach on four image understanding tasks, semantic segmentation, object localization, road extraction, and salient object detection. The proposed HG-CNNs are effective for three of the four tasks; the HG-CNN outperforms strong baselines with fewer floating-point operations (more than 90% reduction) on semantic segmentation; it achieves the state-of-the-art result on road extraction task; it yields compelling performance against state-of-the-arts on salient object detection. The current neural processors (i.e., NPU and GPU) are optimized for regular grid computations, and the proposed module is not necessarily computationally faster or more efficient in practice. However, the paper opens up a new avenue of research, potentially leading to an impact in vertical application domains, such as embedded devices with specialized hardware. We will share all our code and data to promote further research.

2. Related works

The literature of convolutional neural architecture is massive. The section focuses the description on the graph convolution, the graph pooling, and other closely related enhancement techniques in computer vision.

Graph convolution: Hammond et al. [12] and Defferrard et al. [9] formulated a convolution operation on graph-structured data based on spectral graph theory, approximating filters in the Fourier domain using Chebyshev polynomials. Kipf and Welling [17] further proposed a first-order approximation of the spectral graph convolution. Since the above works assume general graph data as inputs, they lack the capability of capturing spatial relationships between nodes for embedded graphs. To remedy this, various methods have been proposed [10, 20, 26, 32, 34, 38]. For instance, Spline-CNN [10] extends a regular kernel function to a continuous kernel function using B-spline bases, where convolution weights for the adjacent nodes can be modified according to their relative spatial position. In our experiments, we compare our direction-aware graph convolution to the above methods.

Ci et al. [7] extends widely used GCN for 3D pose estimation by using different weight parameters for every pair of nodes. However the application of the method is limited to the tasks where the graph structure is pre-defined, e.g., a skeleton body model in 3D pose estimation.

Graph pooling: Graph pooling is a key operation for learning hierarchical graph representations. DiffPool was proposed as a differentiable graph pooling method, in which soft-cluster assignments are directly estimated using graph convolution layers in an end-to-end manner [37]. Other methods defined graph pooling as a node selection problem. In such methods, the top-k representative nodes are selected using a trainable vector p [3] or self-attention mechanism [19]. Whereas the above methods globally select discriminative nodes, AttPool [15] also applies a local attention mechanism to prevent the pooling operation from being stuck within a narrow sub-graph.

Non-grid representations in computer vision: Graph-based representations have been proposed for modeling long-range dependencies in an image. Li et al. proposed a module that performs graph reasoning on a fully-connected graph acquired from a clustering-based graph projection [21]. Similar ideas are also proposed by [6] and [45]. To reduce the computational complexity of the fully-connected graph reasoning, recent work proposed a dynamic graph message passing that adaptively constructs a local graph for each location of a feature map [46]. These methods aim to refine a regular grid representation by adding an extra graph reasoning module on it, and thus still depend on regular convolution for spatial feature extraction. On the other hand, our aim is to replace the redundant regular convolutions by the proposed HG-Conv that gives a unified method for spatial feature extraction and long-range message passing on compact graph representations.

Marin et al. proposed a non-uniform downsampling method that learns deformations from uniform sampling points such that the points near semantic boundaries are sampled as many as possible [24]. More recently, Gao et al. proposed a method that constructs an adaptive triangular mesh on an image plane, and applied the method as learnable downsampling on semantic segmentation task [11]. The method predicts deformations from an initial triangular mesh such that each mesh has a small feature variance. These methods differ from our method in two points; 1) they applied conventional regular convolutions after non-uniform downsampling; 2) for this reason, the deformations are restricted so that the regularity of the output are kept. For this purpose, the methods introduced regularization terms. On the other hand, our graph convolution can operate directly on non-uniform inputs, and hence the proposed graph pooling can generate pooling regions of arbitrary shapes and sizes in a purely data adaptive manner.

PointRend [18] is proposed as a point-based feature refinement module. To generate high-resolution output, the module adaptively samples points from upsampled feature maps and apply MLP to refine the point features. The method is orthogonal to our method.

Ning et al. [28] proposed an efficient convolution method by reusing computation among similar features. While the method achieves an efficient approximation of a regular convolution, the method cannot be applied on non-

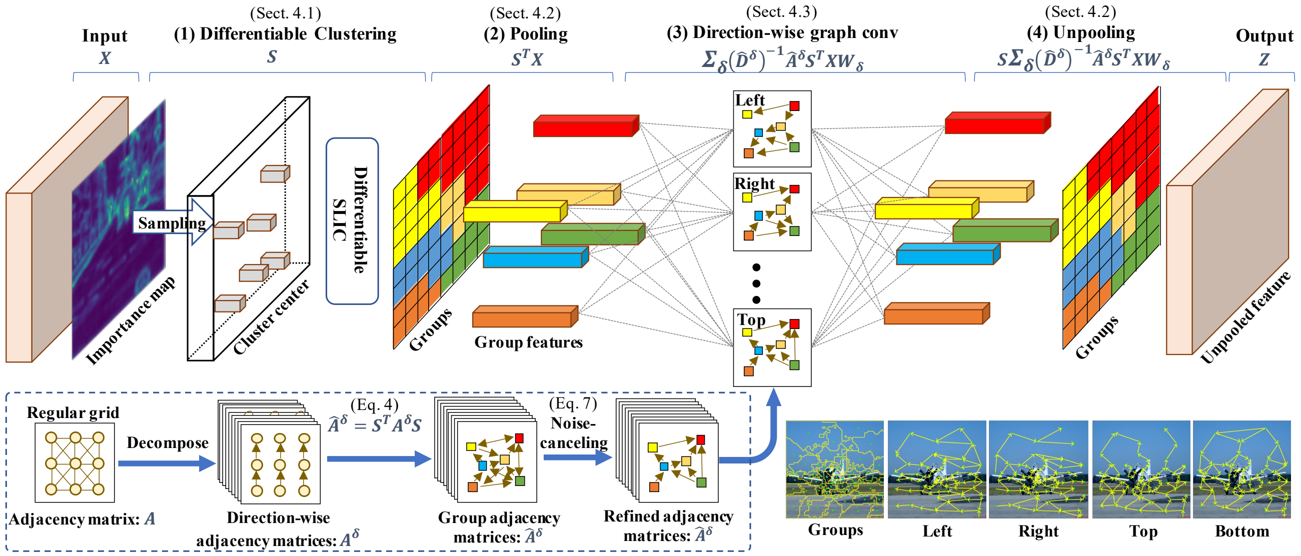

(Sect. 4.1)
(1) Differentiable Clustering
S
(Sect. 4.2)
(2) Pooling
$S^T X$
(Sect. 4.3)
(3) Direction-wise graph conv
$\Sigma_\delta (\hat{D}^\delta)^{-1} \hat{A}^\delta S^T X W_\delta$
(Sect. 4.2)
(4) Unpooling
$S \Sigma_\delta (\hat{D}^\delta)^{-1} \hat{A}^\delta S^T X W_\delta$
Input
X
Output
Z
Sampling
Importance map
Cluster center
Differentiable SLIC
Groups
Group features
Left
Right
Top
Groups
Unpooled feature
Regular grid
Adjacency matrix: A
Decompose
Direction-wise adjacency matrices: A^δ
(Eq. 4)
$\hat{A}^\delta = S^T A^\delta S$
Group adjacency matrices: $\hat{A}^\delta$
(Eq. 7)
Noise-canceling
Refined adjacency matrices: $\hat{A}^\delta$
Groups
Left
Right
Top
Bottom

Figure 2. Illustration of heterogeneous grid convolution. From input feature map, HG-Conv 1) finds groups of pixels shown in colored grids, 2) computes group feature vectors by taking average inside the groups, 3) performs convolution as direction-wise graph-convolutions over the groups, and 4) copies the group feature vector back to the pixels for each group.

grid inputs.

Other enhancement techniques: Dilated convolutions [39, 40] take a strategy of maintaining the feature resolution throughout the networks. Despite being a successful technique [41, 42, 43, 47], Dilated convolutions suffer from large memory consumption and computations due to the high-resolution feature maps. More recently, multi-resolution features [33] or a course-to-fine strategy [1, 4, 22, 23, 29, 31] have been proposed to alleviate the issue.

Multi-scale feature fusion has been studied for aggregating long-range contextual information [4, 13, 36, 39, 47]. The methods build multi-scale features by applying pyramid pooling [47] or dilated convolutions with different dilation rates [4]. Recent works [5, 41, 42, 43, 44] have proposed adaptive context aggregation methods that are based on the feature relation. For instance, OCNet [42] identifies the context for each pixel by adopting a self-attention mechanism. A^2-Net [5] applies a double attention mechanism, where the key features in a scene are aggregated during the first "gather" attention, and are distributed to each pixel during the second "distribute" attention.

3. Convolution as a set of graph-convolutions

Convolution is a direction-wise set of graph-convolutions. We first show this not well-known fact, which will allow us to define heterogeneous grid convolution with the language of graph-convolutions towards a simple and efficient implementation in the next section.

Considering convolution as a message-passing architecture, (3×3) convolution passes messages along nine directions $\boldsymbol{\Delta} = \{\leftarrow, \rightarrow, \uparrow, \downarrow, \nwarrow, \nearrow, \swarrow, \searrow, \circlearrowleft\}$ (See Fig. 3):

$$\overrightarrow{\boldsymbol{z_p}} = \sum_{\boldsymbol{\delta} \in \boldsymbol{\Delta}} \overrightarrow{\boldsymbol{x_{p+\delta}}} \boldsymbol{W_\delta}. \tag{1}$$

$\overrightarrow{\boldsymbol{x_p}}$ is the $(1 \times \mathcal{N}_{in})$ input feature vector at pixel $\boldsymbol{p}$. $\overrightarrow{\boldsymbol{z_p}}$ is the $(1 \times \mathcal{N}_{out})$ output feature vector. With abuse of notation $\boldsymbol{\delta} (\in \boldsymbol{\Delta})$ is a positional displacement for a given direction. $\boldsymbol{W_\delta}$ is the $(\mathcal{N}_{in} \times \mathcal{N}_{out})$ kernel matrix for direction $\boldsymbol{\delta}$. [1]

Let $\boldsymbol{X}$ and $\boldsymbol{Z}$ denote the set of feature vectors for all the pixels as the $(\mathcal{N}_{pix} \times \mathcal{N}_{in})$ and $(\mathcal{N}_{pix} \times \mathcal{N}_{out})$ matrices, where $\mathcal{N}_{pix}$ is the number of pixels. The above message-passing equation can be written for all the pixels as

$$\boldsymbol{Z} = \sum_{\boldsymbol{\delta} \in \boldsymbol{\Delta}} \left(\boldsymbol{D^\delta}\right)^{-1} \boldsymbol{A^\delta X W_\delta}. \tag{2}$$

$\boldsymbol{A^\delta}$ is the $(\mathcal{N}_{pix} \times \mathcal{N}_{pix})$ asymmetric adjacency matrix for direction $\boldsymbol{\delta}$, that is, $\boldsymbol{A}^{\boldsymbol{\delta}}_{ij}$ is 1 if the i_{th} pixel is connected to the j_{th} pixel along direction $\boldsymbol{\delta}$. $\boldsymbol{D^\delta}$ is the $(\mathcal{N}_{pix} \times \mathcal{N}_{pix})$ degree matrix of $\boldsymbol{A^\delta}$: $\boldsymbol{D}^{\boldsymbol{\delta}}_{ii} = \max \left(\sum_j \boldsymbol{A}^{\boldsymbol{\delta}}_{ij}, \epsilon \right)$, where ($\epsilon =$1e-7) is used to avoid divide-by-zero in computing its inverse. The formula inside the summation is the widely used graph-convolution formulation by Kipf and Welling [17], which is summed over the message passing directions.

4. Heterogeneous Grid Convolution

Heterogeneous grid convolution (HG-Conv) is a natural extension of convolution in the heterogeneous grid domain.

[1] A kernel set is a 4D tensor, usually interpreted as a 2D matrix for a pair of input and output channels. $\boldsymbol{W_\delta}$ is a 2D slice of the 4D tensor per pixel, while masking out the contributions outside the given direction $\boldsymbol{\delta}$.

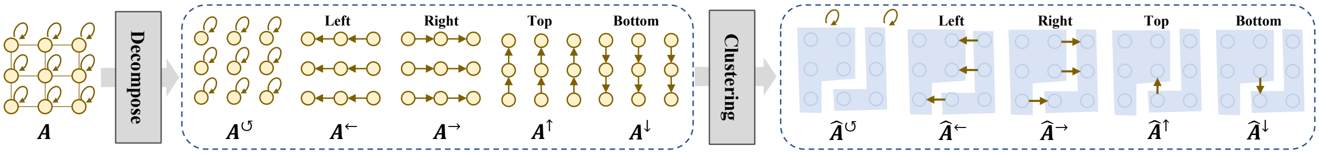

$\boldsymbol{A}$
Decompose
Left
Right
Top
Bottom
$\boldsymbol{A}^{\circlearrowleft}$
$\boldsymbol{A}^{\leftarrow}$
$\boldsymbol{A}^{\rightarrow}$
$\boldsymbol{A}^{\uparrow}$
$\boldsymbol{A}^{\downarrow}$
Clustering
Left
Right
Top
Bottom
$\hat{\boldsymbol{A}}^{\circlearrowleft}$
$\hat{\boldsymbol{A}}^{\leftarrow}$
$\hat{\boldsymbol{A}}^{\rightarrow}$
$\hat{\boldsymbol{A}}^{\uparrow}$
$\hat{\boldsymbol{A}}^{\downarrow}$

Figure 3. Illustration of direction-wise adjacency matrices. Adjacency matrices for group of pixels are defined by the summation of connections between groups. To avoid clutter, we exclude diagonal directions.

Understanding that the convolution is equivalent to the sum of direction-wise graph-convolutions (Eq. 2), HG-Conv is defined as a four-step process shown in Fig. 2: (1. Clustering) Find groups of pixels sharing similar features; (2. Pooling) Compute the group feature vector by taking the average over its pixels; (3. Graph-convolution) Perform convolution as direction-wise graph-convolutions over the groups; and (4. Unpooling) Copy the group feature vector back to the pixels for each group. The four steps are defined in the following formula:

$$\begin{aligned} \boldsymbol{Z} &= \boldsymbol{S} \sum_{\boldsymbol{\delta} \in \boldsymbol{\Delta}} \left(\hat{\boldsymbol{D}}^{\boldsymbol{\delta}}\right)^{-1} \hat{\boldsymbol{A}}^{\boldsymbol{\delta}} \boldsymbol{S}^T \boldsymbol{X} \boldsymbol{W}_{\boldsymbol{\delta}}, && (3) \\ \hat{\boldsymbol{A}}^{\boldsymbol{\delta}} &= \boldsymbol{S}^T \boldsymbol{A}^{\boldsymbol{\delta}} \boldsymbol{S}. && (4) \end{aligned}$$

$\boldsymbol{S}$ is a $\mathcal{N}_{pix} \times \mathcal{N}_{grp}$ group assignment matrix, where $\boldsymbol{S}_{pg}$ defines an assignment weight from pixel p to group g. $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$ is the $\mathcal{N}_{grp} \times \mathcal{N}_{grp}$ adjacency matrix for the groups. $\hat{\boldsymbol{D}}^{\boldsymbol{\delta}}$ is the $\mathcal{N}_{grp} \times \mathcal{N}_{grp}$ degree matrix of $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$. Starting from the stack of input feature vectors $\boldsymbol{X}$, (1. Clustering) is to compute $\boldsymbol{S}$; (2. Pooling) is the left-multiplication of $\boldsymbol{S}^T$; (3. Graph-convoluion) is the left-multiplication of $(\hat{\boldsymbol{D}}^{\boldsymbol{\delta}})^{-1}\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$ and right-multiplication of the learnable kernel $\boldsymbol{W}_{\boldsymbol{\delta}}$; and (4. Unpooling) is the multiplication of $\boldsymbol{S}$.

4.1. Differentiable clustering

The group assignment $\boldsymbol{S}$ is computed by sampling cluster centers from input pixels, and associating input features to the cluster centers using differentiable SLIC algorithm [16]. Note that $\boldsymbol{S}$ is a soft-assignment and trainable in an end-to-end manner. The cluster centers are sampled based on "importance" of each pixel. The importance is defined as L^2 distance between a pixel's feature and its adjacent features. As an extension, the importance map can be incorporated as an attention map for controlling node allocation as shown later.

4.2. Pooling

Given the group of pixels, group feature vectors are computed by the average pooling, which can be written as:

$$\hat{\boldsymbol{X}} = \bar{\boldsymbol{S}}^T \boldsymbol{X}. \tag{5}$$

$\bar{\boldsymbol{S}}$ is a column-wise normalized assignment matrix, i.e., $\bar{\boldsymbol{S}} = \boldsymbol{S}\bar{\boldsymbol{Z}}^{-1}$ and $\bar{Z}_{jj} = \sum_i S_{ij}$. The unpooling operation is defined via its transpose:

$$\boldsymbol{X} = \tilde{\boldsymbol{S}} \boldsymbol{X}' \tag{6}$$

where $\tilde{\boldsymbol{S}}$ is a row-wise normalized matrix, i.e., $\tilde{\boldsymbol{S}} = \tilde{\boldsymbol{Z}}^{-1}\boldsymbol{S}$ and $\tilde{Z}_{ii} = \sum_j S_{ij}$.

4.3. Graph-convolution

A convolution (Eq. 2) is defined as the left multiplication of the adjacency matrix $\boldsymbol{A}^{\boldsymbol{\delta}}$ (with the inverse of the degree matrix) and the right multiplication of the learnable kernel $\boldsymbol{W}_{\boldsymbol{\delta}}$. We define a convolution for groups by simply replacing the adjacency matrix of the pixels $\boldsymbol{A}^{\boldsymbol{\delta}}$ with the adjacency matrix of the groups $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$. $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}_{ij}$ should encode the amount of connections from the i_{th} group to the j_{th} group along direction $\boldsymbol{\delta}$, in other words, how many pixel-level connections there are along $\boldsymbol{\delta}$ from a pixel in the i_{th} group to a pixel in the j_{th} group. This can be calculated easily by the group assignment matrix S: $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}} = \boldsymbol{S}^T\boldsymbol{A}^{\boldsymbol{\delta}}\boldsymbol{S}$ [37]. The convolution for the groups is then given as the left multiplication of $(\hat{\boldsymbol{D}}^{\boldsymbol{\delta}})^{-1}\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$ and the right-multiplication of $\boldsymbol{W}_{\boldsymbol{\delta}}$.

We find that the clusters from differentiable SLIC tend to have complicated shapes and include many small disjoint regions, where Fig. 4 (a) illustrates the situation. Due to the disjoint cluster (depicted in blue), the green cluster is connected to the blue cluster in every direction, which results in a "noisy" group adjacency matrix (Fig. 4 (b)).

To this end, a "noise-canceling operation" is performed on the group adjacency matrix, which cancels out the connection weight by the weight of the opposite direction (Fig. 4 (c)). Let $\bar{\boldsymbol{\delta}}$ be the opposite direction of $\boldsymbol{\delta}$, the "noise-canceling" is performed as follows.

$$\hat{\boldsymbol{A}}^{\boldsymbol{\delta}} \leftarrow \max(\boldsymbol{0}, \hat{\boldsymbol{A}}^{\boldsymbol{\delta}} - \hat{\boldsymbol{A}}^{\bar{\boldsymbol{\delta}}}) \tag{7}$$

By abuse of notation, $\hat{\boldsymbol{A}}^{\boldsymbol{\delta}}$ is replaced with the refined matrices. In practice, the matrices are further simplified by only keeping the direction with the maximum connection (e.g., only the left-wise connection remains in case of Fig. 4 (c)). We empirically find that taking the strongest direction slightly improve the performance.

4.4. HG convolutional modules and networks

A standard convolutional architecture design is to repeat convolutional, batch normalization, and ReLU layers. This

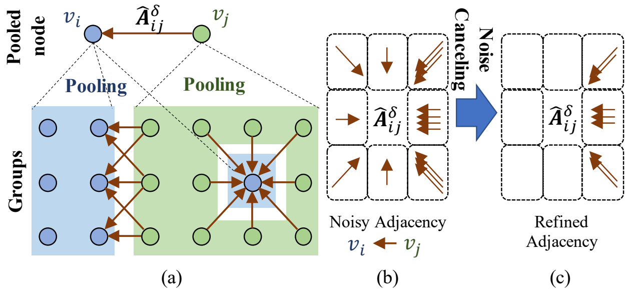


Figure 4. Illustration of noise-canceling for adjacency matrices. Pooling on disjoint cluster assignment (a) results in noisy adjacency matrices (b). Noise-canceling operation refines the noisy adjacency by canceling out connections of opposite direction (c).

design is immediately applicable to HG convolutions:

$$\boldsymbol{Z}_0 = \boldsymbol{S}^T \boldsymbol{X}, \tag{8}$$

$$\boldsymbol{Z}_l = F_{\text{BN-ReLU}} \left(\sum_{\delta \in \boldsymbol{\Delta}} \left(\hat{\boldsymbol{D}}^{\delta} \right)^{-1} \hat{\boldsymbol{A}}^{\delta} \boldsymbol{Z}_{l-1} \boldsymbol{W}_{\delta} \right), \tag{9}$$

$$\boldsymbol{Z} = \boldsymbol{S} \boldsymbol{Z}_L. \tag{10}$$

This HG-Conv module repeats HG-convolutions, batch normalization, and ReLU L times from $\boldsymbol{X}$ to $\boldsymbol{Z}$. $F_{\text{BN-ReLU}}$ is a batch normalization layer followed by ReLU activation. HG convolution is capable of incorporating other popular modules in the CNN literature such as residual blocks. Next, we will design a few representative heterogeneous convolutional neural networks (HG-CNNs) by using HG-convolution and other techniques, where the full specifications are referred to the supplementary.

HG-ResNet: A ResNet [14] is extended by replacing the 4th stage of the network with the HG-Conv module: The pooling (Eq. 8) is inserted at the beginning of stage 4; The subsequent regular convolutions are replaced by the HG-Conv; and The unpooling (Eq. 10) is inserted at the end of the stage. Finally, the module output is concatenated to the stage 3 output and further refined by a 1×1 convolution. Note that the parameter size is equal to the original ResNet except the final 1×1 convolution.

HG-HRNetV2: HRNetV2 [33] is a variant of HRNet that have recently shown outstanding performance on semantic segmentation. Similar to HG-ResNet, the HG-Conv module is applied to the last part of HRNetV2, in particular, all the 4 branches at the last block of stage 4.

HG-ResUNet: ResUNet is a variant of UNet [31], popular in the field of medical image analysis and remote sensing. We applied the HG-Conv module to the last and the first block of the encoder and decoder, respectively. In the same way as HG-ResNet, the output of each module is concatenated with the input, and refined by 1×1 convolution.

5. Experiments

We evaluate the proposed HG-CNNs on four image understanding tasks, semantic segmentation, object localization, road extraction, and salient object detection. On semantic segmentation, the HG-Conv outperforms strong baselines while representing an image with much fewer spatial nodes (less than 2%) (Sect. 5.1). However, HG-Conv does not perform effectively on object localization, which needs further exploration (Sect. 5.2). On the other two tasks, we demonstrate that the HG-Conv is able to control node allocations based on task-specific attention maps, an extension called "active focus" (Sects. 5.3 and 5.4).

5.1. Semantic Segmentation

Setup: We build three HG-CNNs based on HG-ResNet and HG-HRNetV2, and compare against their non-HG counterparts. First, we use HG-ResNet to build two HG-CNNs (HG-ResNet-Dilation and HG-ResNet-DCN) by using dilated convolutions and deformable convolutions at the 3rd residual stage. The non-HG counterparts (ResNet-Dilation and ResNet-DCN) are constructed by simply replacing the HG-Conv by dilated convolution and deformable convolution. The third HG-CNN is HG-HRNetV2, where the non-HG counterpart is HRNetV2, which is the start-of-the-art segmentation network. To further boost performance, we add auxiliary segmentation heads to the input of the HG-Conv modules for all HG-CNNs.

Unless otherwise noted, we determined the number of groups of HG-Conv as 1/64 of the number of input pixels (i.e., the downsampling rate is set as $1/64$). As the HG-Conv adaptively constructs graph representations, the number of floating-point operations varies per image. Although the fluctuation is negligible, we evaluate the FLOPs of the HG-Conv by the average over the validation images. We basically used multi-scale prediction.

Main results: Fig. 6 compares the performance and the computational complexity of the ResNet models and the HG-ResNet models with various depths (18/34/50/101). The HG-ResNet models outperform the corresponding baselines with much less floating-point operations. Especially on PASCAL-context, HG-ResNet34-DCN outperforms ResNet101-DCN (+0.7%) with only 10% floating-point operations. Furthermore, Table 1 shows that HG-HRNetV2 outperforms baseline HRNetV2 with less floating-point operations.

Comparison with other state-of-the-art non-grid convolutions Table 2 shows the comparison against other state-of-the-art non-grid convolutions, DCN [8] and DGMN [46]. In the table, HG-Conv outperforms DCN with less FLOPs. The combination of DCN and HG-conv (DCN at stage 3 and HG-Conv at stage 4) outperforms DGMN with less FLOPs. As for realistic runtime, HG-CNN is not faster than the con-

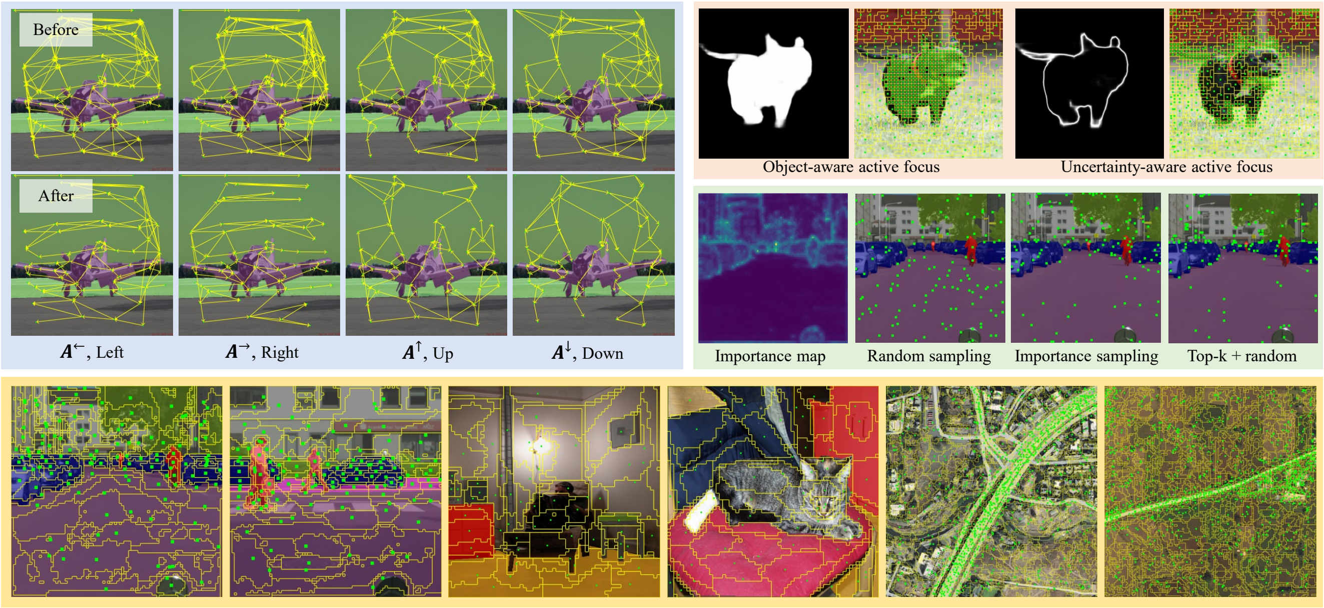


Figure 5. (Top left) Visualization of adjacency matrices before and after noise-canceling. Diagonal directions are excluded to avoid clutter. (Top right) Visualization of active focus for salient object detection. Object-aware attention map and resulting clustering are shown on the left two images, and uncertainty-aware attention map and resulting clustering is shown on the right two images. (Center right) Importance map and sampled cluster centers for each sampling method. (Bottom) Visualization of clustering for outdoor scenes, indoor scenes, and satellite imagery. For the satellite imagery, spatial nodes are focused on road lines (active focus).

ventional CNNs in practice. For instance, "DCN" in Table 2 can process 713×713 inputs in 14.9 FPS, while "DCN+HG-conv" processes the same inputs in 6.5 FPS. This is because today's processors (GPUs) are optimized for regular grid computations, not for graph processing. We believe that the runtime should be improved by more optimized implementation or specialized hardware.

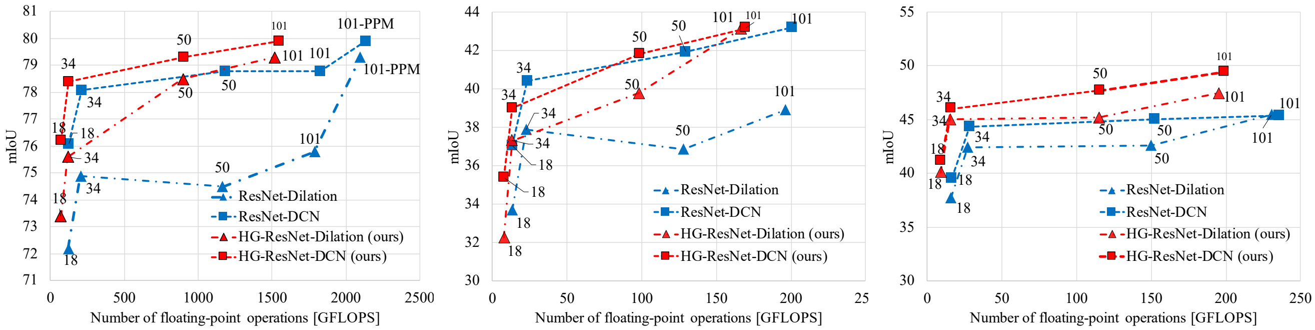
In Table 3, we also compared against other irregular convolutions studied in the field of geometric deep learning (e.g., GMMConv [26] and SplineConv [10]). Specifically, we replace the graph-convolution step of the HG-Conv module with the competing modules (See Table 3). The HG-Conv outperforms the other methods in most cases. Due to engineering challenges, fine-tuning from ImageNet pre-trained models was not possible for some methods. For a fair comparison, we also trained our model from scratch.

Ablation study: To validate the design choices of the HG-Conv, we conducted several ablation studies.
(*Sampling methods*) We compare three sampling methods for the cluster center sampling step of the HG-Conv: random sampling, importance sampling, and a combination of top-k and random sampling [18]. Fig. 5 visualizes the sampled cluster centers, and Table 4 reports the model performances for each sampling method. With random sampling, a large portion of the sampled locations lie on the homogeneous road region, and many objects at the far end are missed. In contrast, the other sampling methods properly place the cluster centers based on the importance map, which results in better segmentation performance.
(*Downsampling ratio*) Table 5 evaluates HG-ResNet with varying downsampling ratio. HG-ResNet outperforms the baseline ResNet with extremely small downsampling ratio.
(*Noise-canceling*) Table 7 demonstrates the effectiveness of noise-canceling operation on the adjacency matrices, which shows clear improvements on two of the three datasets. Max-direction heuristic achieves the well-balanced performance across all of the datasets. The effect of noise-canceling is qualitatively clear in Fig. 5.
(*HG-Conv for 3rd stage*) In Table 8, we further convert the 3rd stage of ResNet101 into the HG-Conv. Whereas the performance degrades from 79.9% to 78.1%, the computational cost reduction increases significantly (i.e., from 15.1 % to 54.7 %). However, as the result of HG-ResNet34 shows, reducing the depth of HG-ResNet is more effective than applying the HG-Conv at a shallow stage.

5.2. Object Localization

We also evaluated the proposed method on object localization tasks such as object detection and instance segmentation. For base models, we use Faster R-CNN and Mask R-CNN with FPN. Specifically, we compared two backbones, ResNet-DCN and HG-ResNet-DCN, on COCO dataset.

Table 9 shows the results of the comparison. While our method achieves 30% reduction in FLOPs, the accuracy of the model is degraded by HG-conv. We leave further explo-



(a) Cityscapes (b) ADE20K (c) PASCAL-context

Figure 6. Evaluation results of ResNet and HG-ResNet on (a) Cityscapes, (b) ADE20K, and (c) PASCAL-context. The number of floating-point operations are calculated for processing image size of $1,024 \times 2,048$, 473×473, and 520×520 respectively.

Table 1. Semantic segmentation results of HG-HRNetV2 on Cityscapes, ADE20K, and Pascal-context datasets (mIoU).

	GFLOPs	Cityscapes	ADE20K	PASCAL-context
HRNetV2 [33]	696	81.7	43.4	54.0
HG-HRNetV2	632	**82.4**	**44.2**	**54.9**

Table 2. Comparison with other non-grid convolution methods on Cityscapes. The models are evaluated at single scale.

	mIoU	#params	FLOPs
ResNet101-Dilation	75.3	52.07M	512.8 G
+ DCN [8]	78.1	+1.20M	+11.3G
+ DGMN [46]	79.2	+2.61M	+24.6G
+ HG-conv	78.8	+6.29M	-78.1G
+ DCN +HG-conv	**79.5**	+7.25M	-69.1G

Table 3. Comparison with other convolution methods for non-uniform inputs on semantic segmentation tasks (mIoU). The methods are evaluated using HG-ResNet101-dilation.

	Cityscapes	ADE20K	PASCAL-context
	w/o transfer		
Kipf and Welling [17]	75.8	37.3	41.7
DynamicEdgeConv [34]	77.2	39.7	**45.9**
HG-Conv	**77.7**	**40.6**	44.7
	w/ transfer		
GMMConv [26]	77.2	40.3	45.9
SplineConv [10]	78.4	41.2	45.5
HG-Conv	**78.8**	**42.0**	**47.5**

ration of HG-Conv on object localization for future work.

5.3. Road Extraction

Road extraction is a task for extracting road graphs from overhead images. On this task, we demonstrate the "active focus" capability that focuses cluster center allocation around predicted road lines. Active focus is particularly effective for road extraction where targets have thin structure. Finally, we apply the HG-Conv into the previous method, and achieve the state-of-the-art performance.

Setup: We make two HG-CNNs and their non-HG counterparts in this evaluation. The first pair is HG-ResUNet and ResUNet, and the second pair is HG-Orientation and Orientation [2] (a current state-of-the-art network for the task). We modified the above models slightly to keep high-resolution information; the stride of the first 7×7 convolution is decreased to 1, and the max-pooling layer is removed ($^+$). Furthermore, we make two modifications to employ active focus (-Attn): 1) A coarse segmentation head is added on the input feature of the HG-Conv module; and 2) The active focus is employed using the coarse prediction map as attention to focus the cluster center allocation on the road lines.

Table 4. Ablation study on sampling methods for the cluster center sampling step of HG-Conv.

	Cityscapes	ADE20K	PASCAL-context
Random sampling	79.1	41.0	45.8
Importance sampling	**79.4**	40.5	**46.0**
Top-k + random [18]	79.0	**41.3**	**46.0**

Table 5. HG-Conv with different downsampling rates. Results are reported for Cityscapes. The models use deformable convolution.

	ResNet101	HG-ResNet101			
Downsampling	1/1	1/16	1/32	1/64	1/128
mIoU	78.1	78.1	80.3	79.5	79.9
GFLOPs	1826	1574	1558	1550	1546
(reduction)	-	(13.8%)	(14.7%)	(15.1%)	(15.3%)

Results: Table 10 shows that our method (HG-Orientation$^+$-Attn) achieves the state-of-the-art result on both IoU and APLS metrics. The effectiveness of HG-ResUNet18$^+$ is also clear (+0.6% and +0.9% for IoU and APLS, compared to ResUNet18$^+$). The active focus is particularly effective on the task: By focusing spatial nodes on the road lines, the network can utilize high resolution information around road, while propagating contextual information from other regions (see Fig. 5 for visualization).

5.4. Salient Object Detection

Salient object detection is a task for identifying the object regions that are most attractive for human eyes. On the task, we compare two different types of active focus.

Table 6. Evaluation of HG-ResUNet and active focus for salient object detection.

	ECSSD		PASCAL-S		DUT-OMRON		HKU-IS		SOD		DUT-TE	
	MaxF↑	MAE↓	MaxF↑	MAE↓	MaxF↑	MAE↓	MaxF↑	MAE↓	MaxF↑	MAE↓	MaxF↑	MAE↓
ResUNet50	0.938	0.0413	0.860	0.0686	0.781	0.0625	0.928	0.0363	0.862	**0.1002**	0.872	0.0430
HG-ResUNet50	0.942	**0.0393**	0.868	0.0662	0.799	0.0563	**0.936**	0.0332	0.865	0.1036	0.881	0.0403
HG-ResUNet50-Attn (object)	**0.943**	0.0396	**0.871**	**0.0654**	**0.801**	**0.0560**	0.935	**0.0325**	0.862	0.1099	**0.884**	0.0387
HG-ResUNet50-Attn (uncertainty)	**0.943**	0.0395	0.867	0.0661	0.794	0.0561	0.934	0.0331	**0.866**	0.1098	**0.884**	**0.0382**

Table 7. Ablation study on noise-canceling of adjacency matrix.

Noise Canceling	Max Direction	Cityscapes	ADE20K	PASCAL-context
		78.4	40.4	45.0
✓		78.5	**41.6**	45.9
✓	✓	**79.0**	41.3	**46.0**

Table 8. Application of HG-Conv on shallower stages. The values in parentheses indicate the reduction efficiency of FLOPs compared to ResNet101-DCN.

	HG-Conv	GFLOPs	mIoU
ResNet101-DCN	none	1,826	78.8
HG-ResNet101-DCN	stage4	1,550 (15.1%)	79.9
	stage3,4	827 (54.7%)	78.1
HG-ResNet34-DCN	stage4	122 (93.3%)	78.4

Table 9. Results for object localization on COCO val-set. Computational complexity of backbone part is shown on "GFLOPs".

	Backbone	GFLOPs	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Faster R-CNN	ResNet50	27.6	**38.7**	**59.4**	**42.2**	—	—	—
	HG-ResNet50	19.1	37.0	58.3	40.4	—	—	—
Mask R-CNN	ResNet50	27.6	**40.0**	**60.4**	**43.9**	**36.1**	**57.3**	**38.6**
	HG-ResNet50	19.1	39.1	60.3	42.9	35.2	57.0	37.5

Table 10. HG-Conv and active focus on road extraction task.

	IoU	APLS
DeepRoadMapper [25]	62.6	65.6
Topology Loss [27]	64.9	66.0
LinkNet34 [48]	62.8	65.3
Orientation [2]	67.2	73.1
ResUNet18	65.2	69.4
ResUNet18$^+$	67.5	71.0
HG-ResUNet18$^+$	68.1	71.9
HG-ResUNet18$^+$-Attn	**68.3**	72.3
Orientation$^+$	67.8	76.0
HG-Orientation$^+$-Attn	**68.3**	**76.4**

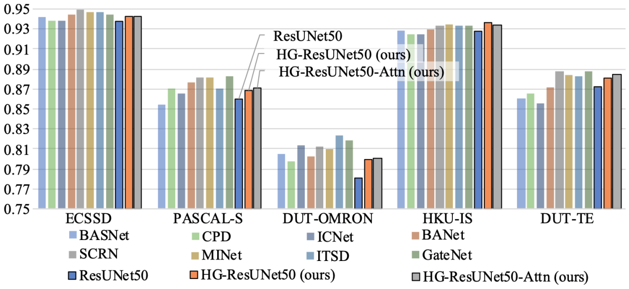


Figure 7. Evaluation results of ResUNet and HG-ResUNet on salient object detection.

Setup: ResUNet50 and HG-ResUNet50 are used for the evaluations. For active focus, the coarse segmentation head is attached on the HG-ResUNet50. We experiment two types of attention: "object-aware" and "uncertainty-aware". In object-aware attention, the predicted object mask is used as an attention, which places cluster centers on the object. In uncertainty-aware attention, entropy of the coarse prediction is used as an attention, which places cluster centers where the prediction is uncertain. For detailed training settings, please refer to the supplementary materials.

Results: Table 6 compares the HG-ResUNet models with baseline ResUNet50. For most datasets, HG-ResUNet50 outperforms the baseline. The object-aware active focus is particularly effective for the task. Whereas the uncertainty-aware active focus also performs well, the performance is worse compared to plain HG-ResUNet50 on three out of six datasets (See Fig. 5).

Comparison to the state-of-the-art methods: Fig 7 compares the baseline and the proposed methods against the previous state-of-the-art. Our method does not achieve the state-of-the-art over all, but performs comparably well for some datasets without using complicated architectures such as iterative refinement modules [30] or edge-aware loss functions [35].

6. Conclusions

This paper presents a novel heterogeneous grid convolution (HG-Conv) that builds an adaptive, efficient and controllable representation by exploiting heterogeneity inherent in natural scenes. Our experimental results demonstrate that HG-CNN is capable of reducing computational expenses significantly without much sacrifice to performance, even achieving state-of-the-art for some tasks. HG-CNN is further capable of controlling the focus of computations based on an application-specific attention maps. Our future work is to further explore the potentials of HG-CNN on more applications as well as the use of FPGA hardware, which is flexible and known to be effective for sparse computation.

Acknowledgement

This paper is partially based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO) and supported by JSPS KAKENHI Grant Number 20H04217.

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. 3

[2] Anil Batra, Suriya Singh, Guan Pang, Saikat Basu, and C V Jawahar Manohar. Improved Road Connectivity by Joint Learning of Orientation and Segmentation. *CVPR*, 2019. 7, 8

[3] Cătălina Cangea, Petar Veličković, Nikola Jovanović, Thomas Kipf, and Pietro Liò. Towards Sparse Hierarchical Graph Classifiers. *NeurIPS Workshop*, 2018. 2

[4] 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. 3

[5] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A^2-Nets: Double Attention Networks. *NeurIPS*, 2018. 3

[6] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, Jiashi Feng, and Yannis Kalantidis. Graph-Based Global Reasoning Networks. *CVPR*, 2019. 2

[7] Hai Ci, Chunyu Wang, Xiaoxuan Ma, and Yizhou Wang. Optimizing network structure for 3D human pose estimation. *ICCV*, 2019. 2

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. *ICCV*, 2017. 5, 7

[9] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *NeurIPS*, 2016. 2

[10] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. SplineCNN: Fast Geometric Deep Learning with Continuous B-Spline Kernels. *CVPR*, 2018. 2, 6, 7

[11] Jun Gao, Zian Wang, Jinchen Xuan, and Sanja Fidler. Beyond Fixed Grid: Learning Geometric Image Representation with a Deformable Grid. *ECCV*, 2020. 2

[12] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. 2

[13] Junjun He, Zhongying Deng, Lei Zhou, Yali Wang, and Yu Qiao. Adaptive pyramid context network for semantic segmentation. *CVPR*, 2019. 3

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CVPR*, 2016. 5

[15] Jingjia Huang, Zhangheng Li, Nannan Li, Shan Liu, and Ge Li. AttPool : Towards Hierarchical Feature Representation in Graph Convolutional Networks via Attention Mechanism. *ICCV*, 2019. 2

[16] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel Sampling Networks. *ECCV*, 2018. 4

[17] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *ICLR*, 2017. 2, 3, 7

[18] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. PointRend: Image Segmentation As Rendering. *CVPR*, 2020. 2, 6, 7

[19] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-Attention Graph Pooling. *ICML*, 2019. 2

[20] Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. DeepGCNs: Can GCNs Go as Deep as CNNs? *ICCV*, 2019. 2

[21] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. *NeurIPS*, 2018. 2

[22] Di Lin, Yuanfeng Ji, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Multi-scale Context Intertwining for Semantic Segmentation. *ECCV*, 2018. 3

[23] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. *CVPR*, 2017. 3

[24] Dmitrii Marin, Zijian He, Peter Vajda, Priyam Chatterjee, Sam Tsai, Fei Yang, and Yuri Boykov. Efficient segmentation: Learning downsampling near semantic boundaries. *ICCV*, 2019. 2

[25] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. DeepRoadMapper: Extracting Road Topology from Aerial Images. *ICCV*, 2017. 8

[26] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. *CVPR*, 2017. 2, 6, 7

[27] Agata Mosinska, Pablo Marquez-Neila, Mateusz Kozinski, and Pascal Fua. Beyond the Pixel-Wise Loss for Topology-Aware Delineation. *CVPR*, 2018. 8

[28] Lin Ning, Hui Guan, and Xipeng Shen. Adaptive deep reuse: Accelerating CNN training on the fly. *ICDE*, 2019. 2

[29] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. *ICCV*, 2015. 3

[30] Youwei Pang, Xiaoqi Zhao, Lihe Zhang, and Huchuan Lu. Multi-scale interactive network for salient object detection. *CVPR*, 2020. 8

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI*, 2015. 3, 5

[32] Przemysław Spurek, Tomasz Danel, Jacek Tabor, Marek Śmieja, Łukasz Struski, Agnieszka Słowik, and Łukasz Maziarka. Geometric Graph Convolutional Neural Networks. *ICLR*, 2020. 2

[33] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep High-Resolution Representation Learning for Visual Recognition. *TPAMI*, 2020. 3, 5, 7

[34] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics*, 38(5):1–12, oct 2019. 2, 7

[35] Zhe Wu, Li Su, and Qingming Huang. Stacked cross refinement network for edge-aware salient object detection. *ICCV*, 2019. 8

[36] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. DenseASPP for Semantic Segmentation in Street Scenes. *CVPR*, 2018. 3

[37] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical Graph Representation Learning with Differentiable Pooling. *NeurIPS*, 2018. 2, 4

[38] Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware Graph Neural Networks. *ICML*, 2019. 2

[39] Fisher Yu and Vladlen Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In *ICLR*, 2016. 3

[40] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *CVPR*, 2017. 3

[41] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-Contextual Representations for Semantic Segmentation. *ECCV*, 2020. 3

[42] Yuhui Yuan and Jingdong Wang. OCNet: Object Context Network for Scene Parsing. *CoRR*, 2018. 3

[43] Fan Zhang, Yanqin Chen, Zhihang Li, Zhibin Hong, Jingtuo Liu, Feifei Ma, Junyu Han, and Errui Ding. ACFNet: Attentional Class Feature Network for Semantic Segmentation. *ICCV*, 2019. 3

[44] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. *CVPR*, 2019. 3

[45] Li Zhang, Xiangtai Li, Anurag Arnab, Kuiyuan Yang, Yunhai Tong, and Philip H. S. Torr. Dual Graph Convolutional Network for Semantic Segmentation. *BMVC*, 2019. 2

[46] Li Zhang, Dan Xu, Anurag Arnab, and Philip H.S. Torr. Dynamic graph message passing networks. *CVPR*, 2020. 2, 5, 7

[47] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 3

[48] Lichen Zhou, Chuang Zhang, and Ming Wu. D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. *CVPRW*, 2018. 8