Self-constructing graph neural networks to model long-range pixel dependencies for semantic segmentation of remote sensing images

Qinghui Liu[a,b] , Michael Kampffmeyer[b,a], Robert Jenssen[b,a] and Arnt-Børre Salberg[a]

[a] Norwegian Computing Center, Dept. SAMBA, P.O. Box 114 Blindern, NO-0314 Oslo, Norway
[b] Dept. Physics and Technology, UiT The Arctic University of Norway, NO-9037 Tromsø, Norway

ARTICLE HISTORY
Compiled March 9, 2022

ABSTRACT
Capturing global contextual representations in remote sensing images by exploiting long-range pixel-pixel dependencies has been shown to improve segmentation performance. However, how to do this efficiently is an open question as current approaches of utilising attention schemes, or very deep models to increase the field of view, increases complexity and memory consumption. Inspired by recent work on graph neural networks, we propose the Self-Constructing Graph (SCG) module that learns a long-range dependency graph directly from the image data and uses it to capture global contextual information efficiently to improve semantic segmentation. The SCG module provides a high degree of flexibility for constructing segmentation networks that seamlessly make use of the benefits of variants of graph neural networks (GNN) and convolutional neural networks (CNN). Our SCG-GCN model, a variant of SCG-Net built upon graph convolutional networks (GCN), performs semantic segmentation in an end-to-end manner with competitive performance on the publicly available ISPRS Potsdam and Vaihingen datasets, achieving a mean F1-scores of 92.0% and 89.8%, respectively. We conclude that the SCG-Net is an attractive architecture for semantic segmentation of remote sensing images since it achieves competitive performance with much fewer parameters and lower computational cost compared to related models based on convolutional neural networks.

KEYWORDS
Remote Sensing, Self-Constructing Graph (SCG), Graph Neural Networks (GNNs), Semantic Segmentation

1. Introduction

Semantic segmentation is one of the fundamental tasks in remote sensing and refers to classifying each pixel in remote sensing images to a semantic category, e.g., buildings, roads, rivers, etc. This is particularly challenging for very high resolution (VHR) aerial images, which are the focus of this work and often contain diverse objects, highly imbalanced classes, and intricate variations in aspect ratio and color textures (e.g. roads, roofs, shadows of buildings, low plants and branches of trees).

CONTACT Qinghui Liu. Email: liu@nr.no

All the authors are associated with the Centre for Research-based Innovation *Visual Intelligence*: `http://visual-intelligence.no`, funded by the Research Council of Norway and consortium partners. RJ, MK and QL are with the UiT Machine Learning Group: `http://machine-learning.uit.no`.

The emergence of deep learning and convolutional neural networks (CNNs) has led to significant improvements for remote sensing image semantic segmentation. Currently, most semantic segmentation models are inspired by the idea of fully convolutional networks (FCNs) (Long, Shelhamer, and Darrell 2015) and U-Net (Ronneberger, Fischer, and Brox 2015), which generally consist of an encoder-decoder architecture where all layers are based on CNNs. U-Net introduces skip connections between the encoder and decoder modules to make the spatial information to be gradually recovered by fusing skipped connections with upsampling layers. Since then, the FCN and encoder-decoder frameworks have been widely adapted and applied to remote sensing domain (Paisitkriangkrai et al. 2015; Sherrah 2016; Lin et al. 2016; Marmanis et al. 2016; Audebert, Le Saux, and Lefèvre 2016; Audebert, Le Saux, and Lefèvre 2017; Wang et al. 2017a; Mou and Zhu 2018; Kampffmeyer, Salberg, and Jenssen 2018; Liu et al. 2019a,b, 2020a). In general, these models differ from each other in how they capture global contextual information at multiple scales. Modeling the global contextual representations aims to obtain richer local and non-local information of complex-shaped and context-dependable objects by exploiting long-range context relations based on spatial and contextual coherence or similarities. For instance, the car is more likely found on the road than on the roof of a building. Capturing global contextual representations has shown to improve the segmentation performance on VHR remote sensing images (Liu et al. 2020a), and also benefit a wide range of computer vision problems (Hu et al. 2018; Zhang et al. 2018; Chen et al. 2019; Adelipour and Ghassemian 2019).

However, how to efficiently capture long-range dependencies is still an open question for semantic segmentation. CNNs are commonly limited by their efficiency and ability to obtain long-range contextual information due to their local valid receptive fields (Zhou et al. 2015). To address this issue, most existing pure deep CNNs based architectures (Ronneberger, Fischer, and Brox 2015; Badrinarayanan, Kendall, and Cipolla 2017; Zhao et al. 2016; Wang et al. 2017b; Peng et al. 2017; Chen et al. 2018; Liu et al. 2019a) normally rely on multi-scale and multi-stream CNN frameworks to obtain richer contextual information and higher performance. This typically requires more trainable parameters and computational resources, often resulting in inefficient and unnecessarily complex models. Several other works have recently been made to introduce self-attention mechanisms (Vaswani et al. 2017) into the convolutional structures to model long-range interdependencies, such as the non-local (NL) neural networks (Wang et al. 2018; Cao et al. 2019) and the Dual attention network (DANet) (Fu et al. 2019) that applied self-attention globally over a whole feature map to produce fully-connected pairwise relationships for non-local feature aggregation. However, these self-attention based methods can only be embedded into pure CNN-based models and have very large memory costs, which hinders their application to VHR remote sensing data.

Recently, Graph Neural Networks (GNNs) have attracted a lot of attention and have emerged as powerful models to capture global dependencies by leveraging an interaction graph when such a graph is naturally available as a source of information about the problem, such as social networks (Chiang et al. 2019; Huang et al. 2018), bio-chemistry (Xu et al. 2019; Velickovic et al. 2019), and so on. Variants of GNNs have also been increasingly explored in various image analysis tasks that include image classification (Knyazev et al. 2019), few-shot and zero-shot learning (Garcia and Bruna 2017; Kampffmeyer et al. 2019; Marino, Salakhutdinov, and Gupta 2016), and have demonstrated very promising performance for various image-level reasoning tasks while significantly reducing the computational cost (Knyazev et al. 2019). However,

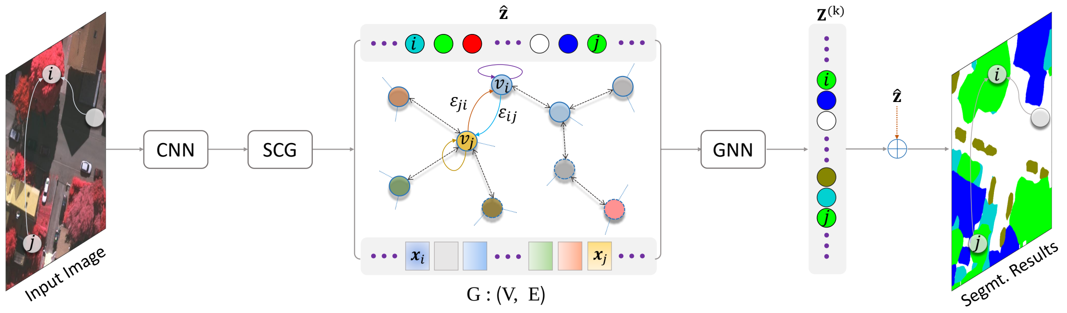

Figure 1. Our SCG-Net uses a conventional CNN backbone to learn a 2D feature map of an input image. The SCG module then learns to transform the 2D feature map into a latent graph structure $\mathrm{G} : (\mathrm{V}, \mathrm{E})$, construct the global context relations ($\varepsilon_{ij} \in \mathrm{E}$) and assign feature vectors ($\boldsymbol{x}_i$) to the vertices ($v_i \in \mathrm{V}$) of the graph. The k-layer GNNs are then exploited to first update the node embedding along the edges of graph with $(k-1)$ layers and finally predict the node labels, $\mathbf{Z}^{(k)}$, by the k-th GNN, the set of node labels are then projected back onto the original 2D plane to output the final segmentation results.

GNNs have not yet fully demonstrated their advantages and have been rarely deployed in dense prediction tasks such as semantic segmentation due to lack of prior knowledge graphs. Previous attempts (Liang et al. 2018; Wang et al. 2019; Landrieu and Simonovsky 2018; Qi et al. 2017) either manually produce prior knowledge graphs, or compute other forms of static graph structure based on raw input images, which are neither flexible nor easily generalized to other image datasets. Specifically, Wang et al. (2019) and Landrieu and Simonovsky (2018) encoded point clouds into k-nearest neighbor (KNN) graphs or super-point graphs to estimate the global context relations. In Qi et al. (2017), a directed graph is constructed based on the 2D position and the depth information of pixels. Pixels close to each other in depth are connected, but those that are close on a 2D grid but distant in depth are not connected with an edge. Note that the prior works mentioned above have not utilized GNNs to infer the final semantic predictions.

Most closely related to our work are the recent work of Li et al. (2020) and Ouyang and Li (2021) that directly apply GNNs to the semantic segmentation problem. Li et al. (2020) introduced a CNN-GCN framework combining CNN and graph convolutional networks (GCN) (Kipf and Welling 2016a) to address multi-label aerial image scene classification. The authors used the SLIC super-pixel algorithm (Achanta et al. 2012) to segment the input image and obtain non-overlapping regions as nodes of the graph, and then construct the spatial relationship (adjacency graph) according to the centroid pixel information of the region. Meanwhile, a series of feature maps are extracted by a pre-trained CNN and upsampled to the original input size. The authors then take the maximum value of each feature map slice as the corresponding vertex feature of the region according to the segmented region boundary, and finally use a GCN to aggregate representations and perform classification. Similarly, in Ouyang and Li (2021), a DSSN-GCN model combining a deep semantic segmentation network (DSNN) and a GCN was proposed for remote sensing image semantic segmentation. While the DSNN module is used to initialize the vertex features instead of a pre-trained CNN, the nodes and adjacency graphs are still derived from the super-pixel segmentation algorithm. Obviously, the above two works largely rely on the super-pixel algorithm that is applied to the raw images to construct the spatial relationships without considering high-level and long-range contextual interdependencies.

As a key solution to effectively exploit GNNs to model global representations and long-range context dependencies in remote sensing, we propose a novel self-constructing graph neural network (SCG-Net) model for pixel-level classification tasks as shown in Fig. 1. Our SCG-Net model can explicitly employ various kinds of GNNs to not only learn global context representations but also directly output the predictions. More specifically, we introduce a novel Self-Constructing Graph module (SCG), inspired by variational graph auto-encoders (VGAE) (Kipf and Welling 2016b) and variational autoencoders (VAE) (Kingma and Welling 2013), to *learn* how a 2D feature map can be transformed into a latent graph structure and how pixels can be assigned to the vertices of the graph from the available training data. In a nutshell, we model relations between pixels that are spatially similar in the CNN, while in the VAE-based SCG module, we incorporate context information between patches that are similar in feature space, but not necessarily spatially close.

Built upon the proposed SCG module, we further develop the end-to-end SCG-Net model for semantic segmentation in VHR arial images. In our SCG-Net framework, a standard CNN (e.g., the ResNet (He et al. 2016)) is utilized to extract high-level feature maps, which are then used to construct the underlying contextual graph using the SCG. A k-layer GNN is then exploited to not only learn the latent embedding, but also to infer the final node-wise labels based on the global contextual graph generated by the SCG module. The predicted node labels are finally projected back onto the original 2D plane. Fig. 1 presents an overview of our method.

Compared to most previous work on graph reasoning for scene recognition tasks (Liang et al. 2018; Wang et al. 2019; Landrieu and Simonovsky 2018; Qi et al. 2017), our SCG-Net effectively streamlines the semantic segmentation pipeline by transforming the pixel-wise classification problem in a Euclidean domain into a node-wise classification task in a structural domain, without relying on deep and multi-scale feature fusion architectures. The proposed SCG framework provides flexibility as it seamlessly combines CNNs with variants of GNNs (e.g. Gori, Monfardini, and Scarselli 2005; Hammond, Vandergheynst, and Gribonval 2011; Bronstein et al. 2017; Niepert, Ahmed, and Kutzkov 2016; Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2016a; Hamilton, Ying, and Leskovec 2017; Xu et al. 2019) together to solve semantic segmentation problems.

Our experiments demonstrate that the SCG-Net achieves very competitive accuracy with real-time performance on the representative ISPRS 2D semantic labeling datasets (Rottensteiner et al. 2012). In summary, our contributions are:

(1) We propose a novel self-constructing graph (SCG) framework that can effectively construct the global context relations (latent graph structure) without relying on prior knowledge graphs. Modeling such a latent long-range dependencies for semantic segmentation of VHR aerial images has not been fully explored yet to the best of our knowledge.
(2) Built upon SCG module, we design a new flexible segmentation network (SCG-Net) that can fully leverage the benefits of both CNNs and variants of GNNs with improved computational efficiency for the pixel-wise classification pipeline in the field of remote sensing.
(3) Our proposed SCG-GCN model achieves competitive performance on different representative remote sensing datasets with much fewer parameters, faster training and lower computational cost.
(4) We validate the effectiveness of our SCG module through extensive ablation studies, and also outline future research directions.

A preliminary version of this paper appeared in Liu et al. (2020c,b). Here, we extend our work by (i) extending our method with several new variants such as Autoencoder

(AE) based SCG, Directed SCG, and various combinations with different GNNs, to further boost the model's flexibility; (ii) expanding the experiment section by including more datasets with more variants of models, providing more training details and presenting additional result comparisons and analysis; (iii) providing sound ablation studies, qualitative analysis and in-depth discussions in terms of model's effectiveness, limitations and challenges for future work.

The paper is structured as follows. Section 2 introduces some preliminaries and background. In Section 3, we present the methodology in details. Section 4 introduces the datasets used in our work. Experimental procedure and evaluation of the proposed method is performed in Section 5. And, finally in Section 6, we draw conclusions and outline future research.

2. Preliminaries

Table 1. Commonly used notations in this paper.

Notations	Descriptions
G	A graph
V	The set of nodes (vertices) in a graph
E	The set of edges (pairs/links of nodes) in a graph
C	The label set $\{1, 2, \ldots, c\}$
Y	The set of ground truth for all labeled nodes, composed by $\{y_i\}$
v_i	The i-th node $\in \mathrm{V}$
N_{v_i}	The set of nodes adjacent to v_i
ε_{ij}	The link $\in \mathrm{E}$ of the node pair (v_i, v_j) directed from v_i to v_j
$\boldsymbol{x}_i \in \mathbb{R}^d$	The d-dimensional feature vector associated to v_i
$\mathbf{A} \in \mathbb{R}^{n\times n}$	The adjacency matrix of a graph
$\mathbf{D} \in \mathbb{R}^{n\times n}$	The degree matrix of $\mathbf{A}$ with self-loop
$\hat{\mathbf{A}} \in \mathbb{R}^{n\times n}$	The normalized graph adjacency matrix
$\mathbf{X} \in \mathbb{R}^{n\times d}$	The feature matrix of a graph, composed by $[\boldsymbol{x}_i]$
$\hat{\mathbf{Z}} \in \mathbb{R}^{n\times c}$	The auxiliary predictions produced by SCG module
$\mathbf{Z}^{(k)} \in \mathbb{R}^{n\times c}$	The predictions produced by GNN module
$\mathbf{Z}^{\top} \in \mathbb{R}^{c\times n}$	Transpose of matrix $\mathbf{Z}$
$\mathbf{I}$	The identity matrix
$\boldsymbol{\theta}^{(l)}$	The learnable parameters of the l-th layer
$\mathbf{Z}^{(l)}$	The hidden feature of the (l)-th layer, composed by $[\boldsymbol{z}_i^{(l)}]$
$\omega^{(l)}$	A learnable parameter or a fixed scalar at layer l
$\boldsymbol{z}_i^{(l)}$	The hidden feature vector of node v_i at the l-th layer
n	The number of nodes
d	The dimension of a node feature vector
c	The number of classes
k	The last layer index
l	The hidden layer index
y_i	The true label of node v_i
$\delta(\cdot)$	The activation function, such as ReLU
$\lvert\cdot\rvert$	The length of a set

Here we begin by presenting some relevant background of the most common GNN models. The notations used in this paper are shown in Table 1. Unless particularly specified, we use upright letters to denote sets and subsets, bold capital characters for matrices, lowercase in italics for scalars and bold italics for vectors.

2.1. *Graph neural networks*

Consider a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ that consists of a set $\mathrm{V} = \{v_i = (i, \boldsymbol{x}_i) : i = 1, 2, \ldots, n\}$ of n vertices or nodes, where $\boldsymbol{x}_i \in \mathbb{R}^d$ denotes feature vectors for node v_i, and an associated set of edges $\mathrm{E} = \{\varepsilon_{ij} = (i, j, A_{ij}) : i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, n\}$, where A_{ij} represents the weight associated to the node pair (v_i, v_j) directed from v_i to v_j. Here, we assume a set of labeled nodes $\{(v_i, y_i \in \mathrm{Y}) : i = 1, 2, \ldots, m\}$, where Y contains the ground truth for all the labeled nodes from a label set $\mathrm{C} = \{1, 2, \ldots, c\}$, and $m = |\mathrm{Y}| \leq n$. The goal of the node classification problem is to learn a mapping $f : \mathrm{V} \rightarrow \mathrm{C}$ such that the labels of unlabeled nodes can be predicted.

The graph, G, can be also represented by $(\mathbf{A}, \mathbf{X})$, where the adjacency matrix[1] $\mathbf{A} \in \mathbb{R}^{n \times n}$ is composed of each link weight $A_{ij} \geq 0 \in \mathbb{R}$, and the feature matrix $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]$ contains each node embedding $\boldsymbol{x}_i$. Essentially, a GNN generalizes the convolution operator to irregular domains, that is typically expressed as a "neighborhood aggregation" or a "message passing" scheme (Fey and Lenssen 2019) in graph structures. GNN learns latent features, $\mathbf{Z}^{(l)}$, by recursively aggregating the information (features) from neighbouring nodes in the graph. The generalized Message-Passing (MP) architecture can be defined as

$$\mathbf{Z}^{(l)} = \mathrm{MP}(\hat{\mathbf{A}}, \mathbf{Z}^{(l-1)}; \boldsymbol{\theta}^{(l)}), l = 1, 2, \ldots, k\ , \tag{1}$$

where $\mathbf{Z}^{(l-1)}$ denotes the node features at the $(l-1)$-th layer and $\mathbf{Z}^{(0)} = \mathbf{X}$, $\boldsymbol{\theta}^{(l)}$ are the trainable parameters of the l-th layer, $\mathbf{Z}^{(l)}$ is the latent embedding space computed after (l) layers and $\mathrm{MP}(\cdot)$ denotes the message-passing function. Note that $\mathbf{A}$ is often re-normalized in a particular way to a normalized matrix $\hat{\mathbf{A}}$ based on the specific GNN variant (Kipf and Welling 2016a; Xu et al. 2019).

The most common GNNs follow the message-passing strategy that can be generalized as Eq. 1. There are many kinds of implementations of the propagation function. In this work, we mainly exploit two representative GNN variants, namely the spectral-based method - Graph Convolutional Network (GCN) (Kipf and Welling 2016a), and the spatial-based method - Graph Isomorphism Network (GIN) (Xu et al. 2019).

2.1.1. Graph convolutional network

The GCN (Kipf and Welling 2016a) was presented as the first-order approximation of the spectral GNN (Hammond, Vandergheynst, and Gribonval 2011), that implements a message-passing function by a combination of linear transformations over one-hop neighbourhoods and non-linearities. It is defined as

$$\mathbf{Z}^{(l)} = \delta\left(\hat{\mathbf{A}}\mathbf{Z}^{(l-1)}\boldsymbol{\theta}^{(l)}\right)\ , \tag{2}$$

where δ denotes the non-linearity function (e.g. ReLU), $\hat{\mathbf{A}}$ is the normalized version of $\mathbf{A}$ with self-loops[2] given as

$$\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{\frac{1}{2}}\ , \tag{3}$$

[1]The adjacency matrix $\mathbf{A}$ is a square matrix used to represent a finite graph. The elements of the matrix indicate whether pairs of vertices are connected or not. Note that we assume G is a weighted graph instead of binary one in this paper.

[2]A self-loop denotes an edge that connects a node to itself. Note that only nodes that have self-loops will include their own features in the aggregate of the features of neighbor nodes.

where $\mathbf{D}$ is the degree matrix[3] that is defined as $D_{ii} = \sum_j (\mathbf{A} + \mathbf{I})_{ij}$, and $\mathbf{I}$ is the identity matrix. By re-normalizating the adjacency matrix (as Eq. 3) with the node degree matrix $\mathbf{D}$, both the vanishing or exploding gradient problem and the numerical instabilities caused by the sensitivity to the scale of each input feature when training such networks can be avoided (Kampffmeyer 2018).

2.1.2. Graph isomorphism network

Unlike the spectral-based GNN, GIN (Xu et al. 2019) was proposed as a spatial-based method that updates the node embedding based on the spatial relations of vertices. More specifically, the GIN's message-passing function can be defined as

$$\boldsymbol{z}_i^{(l)} = \mathrm{MLP}^{(l)} \left(\left(1 + \omega^{(l)}\right) \boldsymbol{z}_i^{(l-1)} + \sum_{j \in \mathrm{N}_{v_i}} \boldsymbol{z}_j^{(l-1)} \right) , \tag{4}$$

where $\boldsymbol{z}_i^{(l)} \in \mathbf{Z}^{(l)}$ is the feature vector of node v_i at the l-th layer, N_{v_i} represents a set of nodes adjacent to v_i, $\mathrm{MLP}^{(l)}$ denotes the multi-layer-perception (MLP) at layer l, and $\omega^{(l)}$ is a learnable parameter or a fixed scalar at layer l. Eq. 4 can be converted to a dense/matrix representation as

$$\mathbf{Z}^{(l)} = \delta \left(\left(\omega^{(l)} \mathbf{I} + (\mathbf{I} + \mathbf{A}) \right) \mathbf{Z}^{(l-1)} \boldsymbol{\theta}^{(l)} \right) . \tag{5}$$

Comparing Eq. 5 to Eq. 2, the major difference between the GIN and the GCN is that the normalized adjacency matrix $\hat{\mathbf{A}}$ is replaced by $\left(\omega^{(l)}\mathbf{I} + (\mathbf{I} + \mathbf{A})\right)$. We can therefore consider the GIN as a special version of the GCN that takes the raw adjacency matrix with a learnable or fixed-scaled diagonal matrix rather than using a Laplacian normalized one for message propagation.

In the following sections, we will use $\mathbf{Z}^{(k)} = \mathrm{GNN}(\mathbf{A}, \mathbf{X})$ to denote an arbitrary GNN module (e.g., GCN or GIN in this paper) implementing k steps of message passing based on some adjacency matrix $\mathbf{A}$ and input node features $\mathbf{X}$, where k is commonly in the range 1 to 6 due to GNN's over-smoothing limitation that would make node representations converge to indistinguishable vectors as the number of layers increases (Zhou et al. 2020).

3. The SCG-Net

We first describe the proposed Self-Constructing Graph (SCG) framework and its variants. We then present our design choices and provide detailed information about the end-to-end model SCG-Net for semantic segmentation of remote sensing images.

[3] The degree matrix is a diagonal matrix which contains the degree of each node—that is, the number of edges attached to each node (Chung, Lu, and Vu 2003).

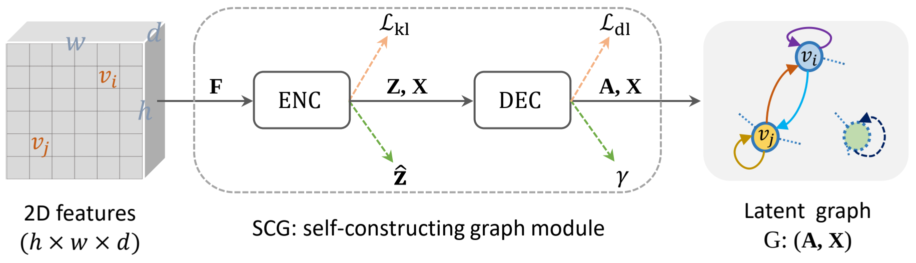

Figure 2. The illustration diagram of the SCG model. Overall, it is composed of 2 key modules, i.e., the ENC module that transforms the 2D features, $\mathbf{F} \in \mathbb{R}^{h \times w \times d}$, to a latent embedding space, $\mathbf{Z} \in \mathbb{R}^{n \times c}$, and, the DEC module that generate the graph representations $\mathrm{G} : (\mathbf{A}, \mathbf{X}) \Leftrightarrow (\mathrm{V}, \mathrm{E})$ by decoding the learned latent embeddings from ENC. Note that SCG model also introduce two regularization functions, i.e., $\mathcal{L}_{\mathrm{kl}}$ is the Kullback–Leibler divergence loss and $\mathcal{L}_{\mathrm{dl}}$ is the diagonal log loss, and two auxiliary terms, i.e., γ is the adaptive factor, $\hat{\mathbf{Z}}$ is the auxiliary predictions.

3.1. *The framework of SCG*

We propose the SCG model that aims to learn the latent graph representation for capturing the global context information across the scene image directly from 2D feature maps without relying on prior knowledge. Overall, the framework of SCG contains two key modules, i.e the encoding (ENC) module that transforms the input 2D features, $\mathbf{F} \in \mathbb{R}^{h \times w \times d}$, to a latent embedding space, $\mathbf{Z} \in \mathbb{R}^{n \times c}$, and the decoding (DEC) module that generates the graph representations $\mathrm{G} : (\mathrm{V}, \mathrm{E}) \Leftrightarrow (\mathbf{A}, \mathbf{X})$ by measuring the similarities between nodes from the learned latent embeddings from the ENC module (see Figure 2). Additionally, to further improve the learning process, the SCG module introduces two regularization terms, i.e the Kullback-Leibler divergence term ($\mathcal{L}_{\mathrm{kl}}$) and the diagonal log penalty ($\mathcal{L}_{\mathrm{dl}}$), and two adaptive enhancement methods, i.e the adaptive enhancement factor (γ) with the auxiliary embeddings ($\hat{\mathbf{Z}}$) to refine the final predictions. Hence, the generalized SCG function can be defined as

$$\left(\mathrm{G}, \mathcal{L}_{\mathrm{kl}}, \mathcal{L}_{\mathrm{dl}}, \hat{\mathbf{Z}}, \gamma\right) = \mathrm{SCG}\left(\mathbf{F}\right) \ . \tag{6}$$

3.1.1. *Encoding module*

In the encoding (ENC) module, we first apply an optional parameter-free pooling operation $\rho(\mathbf{F})$ (e.g. adaptive average pooling in our case) to reduce the spatial dimensions of $\mathbf{F}$ from $(h \times w)$ to $(h' \times w')$, and then reshape it to obtain $\mathbf{X} \in \mathbb{R}^{n \times d}$ as the vertex feature matrix containing n $(n = h'w')$ nodes. This optional pooling operation is used to reduce the time complexity and computational cost of matrix multiplication involved in the following modules when the size of the input image is very large and there are some strict computational constraints. Otherwise, it is not needed in general.

Inspired by the variational autoencoder framework (VAE) (Kingma and Welling 2013), the ENC module learns a mean matrix $\mathbf{M} \in \mathbb{R}^{n \times c}$ and a standard deviation matrix $\mathbf{\Sigma} \in \mathbb{R}^{n \times c}$ of a Gaussian using two single-layer convolutional networks

$$\mathbf{M} = \phi_{\mu}\left(\rho\left(\mathbf{F}\right); \boldsymbol{\theta}_{\mu}\right) \ , \tag{7}$$

$$\log(\mathbf{\Sigma}) = \phi_{s}\left(\rho\left(\mathbf{F}\right); \boldsymbol{\theta}_{s}\right) \ , \tag{8}$$

where c denotes the number of classes, ϕ_μ and ϕ_s represent the convolution layers with 3×3 filter $\boldsymbol{\theta}_\mu$ and 1×1 filter $\boldsymbol{\theta}_s$ respectively, and combine a reshape operator to reshape the outputs from $\mathbb{R}^{h' \times w' \times c} \to \mathbb{R}^{n \times c}$. Since CNNs implicitly learn to extract pixel-coordinate information and encode it in the feature maps (Islam, Jia, and Bruce 2020), the latent embeddings ($\mathbf{M}$ and $\boldsymbol{\Sigma}$) are able to encode both contextual and spatial information. Note that the output of the model for the standard deviation is $\log(\boldsymbol{\Sigma})$ to ensure stable training and positive values for $\boldsymbol{\Sigma}$.

Based on the generated mean and standard deviation of the conditional distribution $\mathbf{Z} \sim \mathcal{N}(\mathbf{M}, \boldsymbol{\Sigma})$, we draw latent variables, and optimize our parameters via gradient descent and backpropagation. We thus make use of reparameterization (Kingma and Welling 2013) to obtain the latent variables as $\mathbf{Z} = \mathbf{M} + \boldsymbol{\Sigma} \odot \boldsymbol{\Upsilon}$ where $\odot$ denotes element-wise product, $\boldsymbol{\Upsilon} \in \mathbb{R}^{n \times c}$ is an auxiliary noise variable and initialized from a standard normal distribution ($\boldsymbol{\Upsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$). This reparameterization is useful to ensure that our sampling process is deterministic and differentiable, and helps the encoder converge to the optimal gradients w.r.t parameters $\boldsymbol{\theta}_\mu$ and $\boldsymbol{\theta}_s$. Meanwhile, a centered isotropic multivariate Gaussian prior distribution is used to regularize the latent variables during training, by minimizing a Kullback-Leibler (KL) divergence loss (Kingma and Welling 2013) that is given as

$$\mathcal{L}_{\text{kl}} = -\operatorname{KL}\left[\mathcal{N}(\mathbf{M}, \boldsymbol{\Sigma}) \| \mathcal{N}(\mathbf{0}, \mathbf{I})\right] = -\frac{1}{2nc} \sum_{i=1}^{n} \sum_{j=1}^{c} \left(1 + \log\left(\mathit{\Sigma}_{ij}^2\right) - M_{ij}^2 - \mathit{\Sigma}_{ij}^2\right) , \quad (9)$$

where $\mathit{\Sigma}_{ij}$ and M_{ij} denote the element i, j of matrix $\boldsymbol{\Sigma}$ and $\mathbf{M}$ respectively. This regularization term penalizes the model if the encoder outputs representations $\mathbf{Z}$ (also called approximated variational posterior distributions) that are different than those from a standard normal distribution (our prior distribution). This encourages a robust latent variable space, where representations of similar semantic classes remain sufficiently close together (e.g., tree-images with even very different shapes, sizes, and textures could be encoded by very similar representations to each other, rather than that different trees have very different embeddings.).

Here, we also introduce a auxiliary embedding term $\hat{\mathbf{Z}}$ that is defined as

$$\hat{\mathbf{Z}} = \mathbf{M} \odot (1 - \log(\boldsymbol{\Sigma})) , \quad (10)$$

which can be used to refine the final prediction of the network after information has been propagated along the learned graph. Intuitively, the auxiliary term $\hat{\mathbf{Z}}$ can be viewed as a standard normal distribution space transformed from the latent space by $\mathbf{M} \odot \boldsymbol{\Sigma}^{-1}$. For computational simplicity and stabilization, we replace $\boldsymbol{\Sigma}^{-1}$ with $(1 - \log(\boldsymbol{\Sigma}))$, and $\log(\boldsymbol{\Sigma})$ is constrained to be smaller than 1 during the training.

3.1.2. Decoding module

In the decoding (DEC) block, the graph adjacency matrix $\mathbf{A}$ is generated by an inner product between latent embeddings as $\mathbf{A} = \operatorname{ReLU}\left(\mathbf{Z}\mathbf{Z}^\top\right)$. Note, $\mathbf{A}$ is not binary in our case but weighA weighted graph is a graph where each edge/link has a specific numerical value (a weight) associated with it. In our case, the weight $A_{ij} = \operatorname{ReLU}(\boldsymbol{z}_i^\top \boldsymbol{z}_j)$, is a positive real number that represents the weight associated to the edge of the node pair (v_i, v_j). and undirected. Thus, $A_{ij} = A_{ji} > 0$ indicates the presence of an edge associated with the learned weight (A_{ij}) between node i and j. Essentially, the DEC-block measures the similarity between patches (feature vectors of nodes) to build the graph that connects similar node representations together, such that the similar scene regions exchange information. And intuitively, we consider A_{ii} shall be > 0 and close to 1. We therefore introduce a diagonal log regularization term defined as

$$\mathcal{L}_{\mathrm{dl}} = -\frac{\gamma}{n^2} \sum_{i=1}^{n} \log \left(|A_{ii}|_{[0,1]} + \epsilon \right) \ , \tag{11}$$

where the index $_{[0,1]}$ denotes that A_{ii} is clamped to $[0,1]$, ϵ is a small positive infinitesimal scalar (e.g, $\epsilon = 10^{-7}$) and γ is the adaptive factor computed as

$$\gamma = \sqrt{1 + \frac{n}{\sum_{i=1}^{n} A_{ii} + \epsilon}} \ .$$

In order to preserve local information and stabilize training, we introduce an adaptive enhancement approach applied to both the adjacency matrix and the auxiliary embeddings. The enhanced $\mathbf{A}$ is designed as $\mathbf{A} = \mathbf{A} + \gamma \, \mathrm{diag}(\mathbf{A})$, and the enhanced auxiliary term $\hat{\mathbf{Z}}$ is updated as $\gamma \hat{\mathbf{Z}}$.

3.2. *Variants of SCG*

3.2.1. *Auto-encoder based SCG*

We also introduce an auto-encoder based SCG module ($\mathrm{SCG}_{\mathrm{ae}}$) as an alternative to a variational auto-encoder based structure to learn the latent embedding $\mathbf{Z}$. The ENC module can thus learn $\mathbf{Z}$ as follows

$$\mathbf{Z} = \phi_{\mu} \left(\rho \left(\mathbf{F} \right); \boldsymbol{\theta}_{\mu} \right) \ . \tag{12}$$

Then the overall formula of $\mathrm{SCG}_{\mathrm{ae}}$ simplifies to

$$(\mathrm{G}, \mathcal{L}_{\mathrm{kl}}, \mathbf{Z}, \gamma) = \mathrm{SCG}_{\mathrm{ae}}(\mathbf{F}) \ . \tag{13}$$

Note that there is no need to compute the mean matrix ($\mathbf{M}$), the deviation matrix ($\boldsymbol{\Sigma}$), the auxiliary term ($\hat{\mathbf{Z}}$), and the Kullback-Leibler divergence loss ($\mathcal{L}_{\mathrm{kl}}$), as for the VAE approach. However, the DEC part of $\mathrm{SCG}_{\mathrm{ae}}$ is kept unchanged. Meanwhile, in order to be consistent with the VAE method, we just make use of $\mathbf{Z}$ as the auxiliary term to refine the final prediction of our model.

3.2.2. *Directed graph-based SCG*

The standard SCG framework can only generate un-directed graphs (symmetric adjacency matrix) by the inner product operation. Here we propose a SCG variant $\mathrm{SCG}_{\mathrm{dir}}$ that is able to produce directed[4] graph structures. To do so, we first normalize the

[4]In this work, A_{ij} and A_{ji} represent the weights associated the edges (ε_{ij} and ε_{ji}) between the node pair (v_i, v_j) directed from v_i to v_j and from v_j to v_i respectively. For the un-directed graph, A_{ij} is always equal to A_{ji}, while for the directed graph, A_{ij} is in general not equal to A_{ji}.

latent embeddings by the standard softmax operation as

$$\bar{Z}_{ij} = \frac{e^{Z_{ij}}}{\sum_{k=1}^{c} e^{Z_{ik}}} \quad i = 1, 2, \ldots, n \quad j = 1, 2, \ldots, c\,. \tag{14}$$

Then the directed adjacency matrix can be defined as $\mathbf{A} = \mathrm{ReLU}\left(\bar{\mathbf{Z}}\mathbf{Z}^\top\right)$. Note that $A_{ij} \neq A_{ji} > 0$ indicates the presence of directed edges associated with two different weights between node i and j. Intuitively, we assume that the directed graph can represent richer interactions between objects.

3.3. *SCG-Net architecture*

Built upon SCG and GNNs with the incorporation of backbone CNNs, we propose a new end-to-end method (SCG-Net) that streamlines semantic segmentation as a node classification problem. Fig. 1 shows the illustration of the proposed SCG-Net model for semantic segmentation of remote sensing images. The model architecture details are shown in Table 2.

Table 2. The general end-to-end SCG-Net Model Details with one sample of input image size of $h_0 \times w_0 \times 3$.

Layers	Outputs	Sizes
Backbone-CNN	$\mathbf{F}$	$h \times w \times 1024$
SCG	$(\mathbf{A}, \mathbf{X}, \hat{\mathbf{Z}}, \gamma)$	$(n \times n), (n \times 1024), (n \times c), \mathbb{R}_{\geq 1}$
GNN^1	$(\hat{\mathbf{A}}, \mathbf{Z}^{(1)})$	$n \times d$
GNN^2	$(\hat{\mathbf{A}}, \mathbf{Z}^{(2)})$	$n \times c$
Sum(opt)	$(\gamma\hat{\mathbf{Z}} + \mathbf{Z}^{(2)})$	$n \times c$
Projection	$\tilde{\mathbf{Y}}$	$h_0 \times w_0 \times c$

3.3.1. CNN backbone

A conventional CNN backbone, e.g, the pretrained ResNet50 (He et al. 2016) in this work, is employed to extract the high-level representations. Following our previous work (Liu et al. 2020a), we only utilize the first three bottleneck layers of pretrained ResNet50 and remove the last bottleneck layer and the fully connected layers to reduce the number of parameters to train. We assume that the size of the input image is $h_0 \times w_0 \times 3$ (with 3 color channels), the output size of our CNN backbone is thus $h \times w \times 1024$ $(h, w = \frac{h_0}{16}, \frac{w_0}{16})$.

3.3.2. SCG and GNN decoder

We combine our proposed SCG module (e.g, the standard SCG, or other variants including SCG$_{\mathrm{ae}}$ and SCG$_{\mathrm{dir}}$) with a 1-layer GNN (either GCN or GIN in this work) as the decoder. We utilize ReLU activation and batch normalization for the GNN layer. Guided by our empirical observation, we set $n = h \times w$ and $d = 128$ in this work, and $c = 6$ is equal to the number of classes in the datasets.

3.3.3. Prediction and projection

The final prediction $(\mathbf{Z}^{(2)})$ is produced by the second layer GNN (either GCN or GIN) without activation and normalization. There is also an optional element-wise sum operation for the auxiliary term ($\gamma\hat{\mathbf{Z}}$) to refine the final results. To project the representations back to the 2D space , we first reshape the predictions ($n \times c \longrightarrow h \times w \times c$), and then conduct up-sampling with bilinear interpolation to obtain the final segmentation maps with original spatial resolution size ($h_0 \times w_0 \times c$).

3.3.4. Variants of SCG-Net models

Table 3 shows some variants of the SCG-Net model that were investigated in this work. Note that we utilize the same backbone CNN (customized ResNet50) for all these models. Our proposed SCG-Net framework can be easily extended and tailored to various deep CNNs and GNN-like networks with flexible configurations w.r.t the depth (e.g. the number of layers), width (e.g. the size of inputs) and density (e.g. the number of nodes and hidden features) for different problems.

Table 3. Variants of SCG-Net Models with different settings for evaluations in this paper. Note that the superscripts ^{1,2} indicate the 1st and 2nd GNN layers and the subscript $_{\text{\_ns}}$ means no sum of the auxiliary term.

SCG-Net Variants	SCG Layer	GNN^1	GNN^2	Sum(opt)
SCG-GCN	SCG	GCN	GCN	✓
SCG-GIN	SCG	GIN	GIN	✓
SCG-GCN-GIN	SCG	GCN	GIN	✓
SCG$_{\text{ae}}$-GCN	SCG$_{\text{ae}}$	GCN	GCN	✓
SCG$_{\text{dir}}$-GCN	SCG$_{\text{dir}}$	GCN	GCN	✓
SCG$_{\text{dir}}$-GCN-GIN	SCG$_{\text{dir}}$	GCN	GIN	✓
SCG-GCN$_{\text{\_ns}}$	SCG	GCN	GCN	✗

4. Benchmark datasets

We evaluate our proposed methods on two public benchmark datasets, namely the ISPRS 2D semantic labeling contest datasets (Rottensteiner et al. 2012). The ISPRS datasets are comprised of aerial images over two cities in Germany: Potsdam[5] and Vaihingen[6]. They have been labelled with six common land cover classes: impervious surfaces, buildings, low vegetation, trees, cars and clutter.

The Potsdam dataset consists of 38 tiles of size 6000 × 6000 pixels with a ground resolution of 5cm, where 14 of these are used as hold-out test images. The tiles consist of Red-Green-Blue-Infrared (RGB-IR) four-channel images. While both the digital surface model (DSM) and normalized DSM (nDSM) data are also included in the dataset, we only use RGB images in this paper in order to fairly compare with other work.

The Vaihingen dataset contains 33 tiles of varying size (on average approximately 2100 × 2100 pixels) with a ground resolution of 9cm, of which 17 tiles are used as hold-out test images. The tiles are composed of Infrared-Red-Green (IRRG) 3-channel images. Though DSMs and nDSMs data are also available for all images in the dataset, we only focus on the 3-channel IRRG data in this paper to fairly compare with other work.

[5]http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html
[6]http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html

5. Experiments and results

In this section, we investigate our proposed methods and networks on the Potsdam and Vaihingen (Section 5.3) datasets. We first present the training and evaluation details, and then report both qualitative and quantitative results for the multi-class semantic labeling task.

5.1. *Training details*

Following our previous work (Liu, Salberg, and Jenssen 2018), we train the models using Adam (Kingma and Ba 2014) with AMSGrad (Reddi, Kale, and Kumar 2018) as the optimizer with weight decay 2×10^{-5} applied to all learnable parameters except biases and batch-norm parameters, and polynomial learning rate (LR) decay $(1 - \frac{cur\_iter}{max\_iter})^{0.9}$ with the maximum iterations of 10^8. The learning rate of the bias parameters is $2 \times$ LR. We use initial LRs of $\frac{8.5 \times 10^{-5}}{\sqrt{2}}$ and utilized a step-wise LR schedule method that reduces the LR by a factor of 0.85 every 15 epochs. Based on our training observations to achieve fast and stable convergence, we apply a dice loss function (Milletari, Navab, and Ahmadi 2016) defined as

$$\mathcal{L}_{\text{dice}} = 1 - \frac{1}{|\mathrm{Y}|} \sum_{i \in \mathrm{Y}} \frac{2 \sum_{j \in \mathrm{C}} y_{ij} \tilde{y}_{ij}}{\sum_{j \in \mathrm{C}} y_{ij} + \sum_{j \in \mathrm{C}} \tilde{y}_{ij}} , \tag{15}$$

where Y contains all the labeled nodes, C denotes the label set, and $\tilde{y}_{ij}$ is the prediction of the ij-th node with its ground-truth label to be y_{ij}.

Together with the two regularization terms $\mathcal{L}_{\text{kl}}$ and $\mathcal{L}_{\text{dl}}$ as defined in Eq. 9 and Eq. 11, respectively, the final cost function of our model is defined as

$$\mathcal{L} = \mathcal{L}_{\text{dice}} + \mathcal{L}_{\text{kl}} + \mathcal{L}_{\text{dl}} . \tag{16}$$

We train and validate the networks for both the Potsdam and Vaihingen datasets with 4000 randomly sampled patches of size 448×448 as input and using a batch size of 4. The training data is sampled uniformly and randomly shuffled for each epoch. We conduct all experiments in this paper using PyTorch (Paszke et al. 2017) on a single computer with one NVIDIA 1080Ti GPU.

5.2. *Augmentation and evaluation methods*

We randomly flip or mirror images for data augmentation (with probability 0.5). The albumentations library (Buslaev et al. 2020) for data augmentation is utilized in this work. Please note that all training images are normalized to [0.0, 1.0] after data augmentation. We apply test time augmentation (TTA) via flipping and mirroring (Liu et al. 2019a). And also we use sliding windows (with 448×448 size at a 100-pixel stride) on a test image and stitch the results together by averaging the predictions of the over-lapping TTA regions to form the output. The performance is measured by both the F1-score, and the mean Intersection over Union (IoU).

Table 4. Comparisons between our method with other published methods on the hold-out RGB test images of Potsdam dataset.

Models	OA	Surface	Building	Low-veg	Tree	Car	mF1
HED+SEG.H-Sc1 (Marmanis et al. 2016)	0.851	0.850	0.967	0.842	0.686	0.858	0.846
RGB+I-ensemble (Kampffmeyer, Salberg, and Jenssen 2018)	0.900	0.870	0.936	0.822	0.845	0.892	0.873
Hallucination (Kampffmeyer, Salberg, and Jenssen 2018)	0.901	0.873	0.938	0.821	0.848	0.882	0.872
SegNet (Audebert, Le Saux, and Lefèvre 2017)	0.897	0.930	0.929	0.850	0.851	0.951	0.902
DST_2 (Sherrah 2016)	0.903	0.925	0.964	0.867	0.880	0.947	0.917
FuseNet+OSM (Audebert, Le Saux, and Lefèvre 2017)	**0.923**	**0.953**	0.959	0.863	0.851	**0.968**	0.918
DDCM-R50 (Liu et al. 2019a)	0.908	0.929	**0.969**	**0.877**	**0.894**	0.949	**0.923**
SCG-GCN	0.903	0.924	0.952	0.873	0.893	**0.960**	0.920

5.3. *Test results*

Following previous work (Kampffmeyer, Salberg, and Jenssen 2018; Liu et al. 2019a) for fair evaluation and comparison, the labeled part of the Potsdam dataset is split into a training set (19 images), a validation set (2 images of 4_10 and 7_10), and a local test set (3 images of areas 5_11, 6_9 and 7_11). The Vaihingen dataset is similarly divided into training (10 images), validation (2 images of areas 7 and 9) and local test set (4 images of areas 5, 15, 21 and 30). While the hold-out test sets contain 14 images (areas: 2_13, 2_14, 3_13, 3_14, 4_13, 4_14, 4_15, 5_13, 5_14, 5_15, 6_13, 6_14, 6_15 and 7_13) and 17 images (areas: 2, 4, 6, 8, 10, 12, 14, 16, 20, 22, 24, 27, 29, 31, 33, 35 and 38) for the Potsdam and Vaihingen datasets, respectively.

Table 5. Comparisons between our method with other published methods on the hold-out IRRG test images of Vaihingen Dataset.

Models	OA	Surface	Building	Low-veg	Tree	Car	mF1
ADL_3 (Paisitkriangkrai et al. 2015)	0.880	0.895	0.932	0.823	0.882	0.633	0.833
DNN_HCRF (Liu et al. 2019b)	0.878	0.901	0.932	0.814	0.872	0.720	0.848
DST_2 (Sherrah 2016)	0.891	0.905	0.937	0.834	0.892	0.726	0.859
UOA (Lin et al. 2016)	0.876	0.898	0.921	0.804	0.882	0.820	0.865
ONE_7 (Audebert, Le Saux, and Lefèvre 2016)	0.898	0.910	0.945	**0.844**	0.899	0.778	0.875
DLR_9 (Marmanis et al. 2016)	0.903	0.924	0.952	0.839	0.899	0.812	0.885
GSN (Wang et al. 2017a)	0.903	0.922	0.951	0.837	**0.899**	0.824	0.887
RWSNet (Jiang et al. 2020)	0.899	0.916	0.947	0.840	0.893	0.860	0.891
DDCM-R50 (Liu et al. 2019a)	**0.904**	**0.927**	**0.953**	0.833	0.894	**0.883**	**0.898**
SCG-GCN	**0.904**	0.924	0.948	0.839	0.897	0.880	**0.898**

5.3.1. Comparison with other work

We compare our results to other related published work on the ISPRS Potsdam RGB dataset and Vaihingen IRRG dataset. These results are shown in Table 4 and 5 respectively. Our single model achieves an overall F1-score (92.0%) on the Potsdam RGB dataset, which is comparable to state-of-the-art (0.3% point lower compared to the best model DDCM-R50 (Liu et al. 2019a), but 0.2% $\sim$ 6.0% higher than the remaining models). The performance gap between our model with DDCM-R50 may be explained by the fact that DDCM-R50 utilized multi-scale (both low-resolution and high-resolution) feature maps for fusion and prediction to achieve the best test results, while our model only used single-level (the last low-resolution) features for prediction. Further research is required to improve the SCG-Net's performance through the addition of multi-scale architectures in the future. Similarly, our model trained on the Vaihingen IRRG images, also obtained very competitive performance with 89.8% F1-score, which is around +1.1% higher than GSN (Wang et al. 2017a) and the same as the best model DDCM-R50. Fig. 5 and 6 show the qualitative comparisons of the land cover segmentation results from our model and the ground truths on the test set.

5.3.2. *Comparison of computational efficiency*

We also compared our methods to some popular architectures on the local Potsdam RGB test set (Liu, Salberg, and Jenssen 2018) in terms of parameter size, computational cost (FLOPs), inference time on both CPU and GPU, and mIoU evaluated on the full reference ground truths of the dataset. Table 6 details the quantitative results of our SCG-Net against others. Compared to PSPNet (Zhao et al. 2016) and SegNet (Badrinarayanan, Kendall, and Cipolla 2017), our model consumes about 10x and 13x less FLOPs with 5x and 4.6x fewer parameters and 21x and 42x faster inference speed on CPU, but achieves 2.1% and 2.9% higher mIoU respectively. In addition, compared with the best model DDCM-R50, our model has about 12% fewer parameters with around 40% faster inference speed on GPU, and achieves better performance on our local Potsdam test set.

Table 6. Quantitative Comparison of parameters size, FLOPs (measured on input image size of $3\times256\times256$), Inference time on CPU and GPU separately, and mIoU on Potsdam RGB dataset.

Models	Backbones	Parameters (Million)	FLOPs (Giga)	Inference time (ms - CPU/GPU)	mIoU*
U-Net (Ronneberger, Fischer, and Brox 2015)	VGG16	31.04	15.25	1460 / 6.37	0.715
FCN8s (Long, Shelhamer, and Darrell 2015)	VGG16	134.30	73.46	6353 / 20.68	0.728
SegNet (Badrinarayanan, Kendall, and Cipolla 2017)	VGG19	39.79	60.88	5757 / 15.47	0.781
GCN (Peng et al. 2017)	ResNet50	23.84	5.61	593 / 11.93	0.774
PSPNet (Zhao et al. 2016)	ResNet50	46.59	44.40	2881 / 81.08	0.789
DUC (Wang et al. 2017b)	ResNet50	30.59	32.26	2086 / 68.24	0.793
DDCM-R50 (Liu et al. 2019a)	ResNet50†	9.99	4.86	238 / 10.23	0.808
SCG-GCN	ResNet50†	**8.74**	**4.47**	**161 / 6.08**	**0.810**

† denotes only the first three bottleneck layers of ResNet50 were used in the model.

5.4. *Analysis and ablations*

In our analysis and ablation studies, we explore how the graph size and other components of our framework such as the adaptive auxiliary term and regularization terms affect the training and final performance. We also analyze the learned node graphs and point out some limitations of our model. For the study we choose the SCG-GCN model with ResNet50 backbone and train and evaluate it on the Vaihingen IRRG dataset.

5.4.1. *Effect of the adaptive auxiliary and regularization terms*

We evaluated the effectiveness of the proposed adaptive auxiliary term and the regularization. The test performance is presented in Table 7. Generally, without using adaptive auxiliary terms ($\gamma\hat{\mathbf{Z}}$) and regularization ($\mathcal{L}_{\text{kl}} + \mathcal{L}_{\text{dl}}$), the number of training steps required to reach convergence is increased by over 4 time as shown in Fig 3, and, when regularization is used alone, the training converges faster, but the test performance is significantly decreased on small objects (i.e, cars -2.0%) and overall (-0.8%). Similarly, when only applying the auxiliary term, the convergence speed is even faster than the default setting (0.6x training steps) while the test results also became worse both on big objects like buildings (-0.7%) and overall classes (-0.7%). Fig. 3 illustrates the training performance of the first 18 training epochs with different settings. Note, both the auxiliary term and the regularization loss can stabilize the training process and speed up the convergence. However, when the auxiliary term or the regularization loss is used alone, we observe some degradation in performance. Note that the validation accuracy shown in the learning curve may have some bias and will not perfectly estimate the performance of the model due to the fact that our validation set is relatively small and contains only 2 images compared to the 17 images in the test set.

Table 7. Test performance of different settings on Vaihingen test set.

$\gamma\hat{\mathbf{Z}}$	$\mathcal{L}_{\mathrm{kl}} + \mathcal{L}_{\mathrm{dl}}$	SCG	OA	Δ%	Building	Δ%	Car	Δ%	mF1	Δ%	Steps (K)	Δ
✗	✗	✓	0.899	-0.5	0.946	-0.2	-0.877	-0.3	0.893	-0.5	97	4.1x
✗	✓	✓	0.899	-0.5	0.946	-0.2	0.860	-2.0	0.890	-0.8	46	1.9x
✓	✗	✓	0.895	-0.9	0.941	-0.7	0.878	-0.2	0.891	-0.7	15	0.6x
✓	✓	✗	0.887	-1.7	0.929	-1.9	0.829	-5.1	0.874	-2.4	120	5.0x
✓	✓	✓	**0.904**	-	**0.948**	-	**0.880**	-	**0.898**	-	**24**	-

[*] Steps (K) denote training iterations with K=1000. Note that 1K steps = 1 epoch in this work.

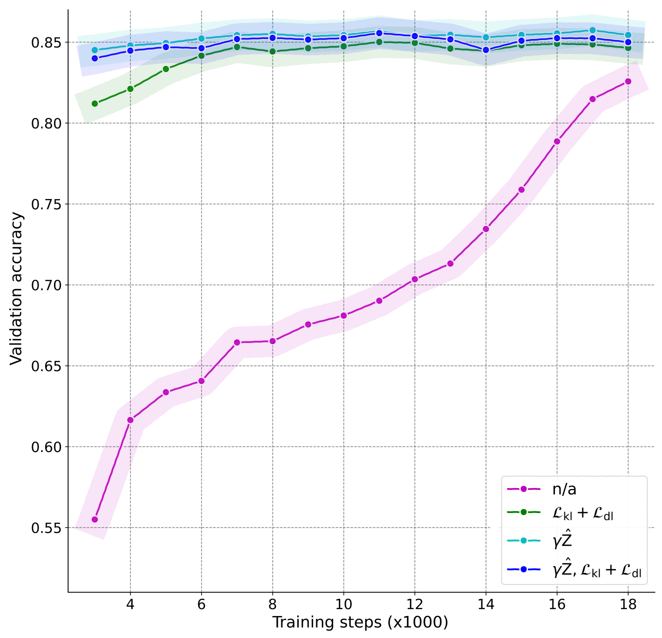

Figure 3. Training performance of different settings. Here n/a denotes the training without using adaptive auxiliary terms ($\gamma\hat{\mathbf{Z}}$) and regularization ($\mathcal{L}_{\mathrm{kl}} + \mathcal{L}_{\mathrm{dl}}$).

5.4.2. Effect of the graph size

We investigated the effect of node size and corresponding input image size on the performance. Note that, our backbone-CNN outputs the last feature map with a down-sample rate of 16 with respect to its input size (i.e., $h_0 \times w_0$). The node size is thus set to $\frac{h_0 w_0}{16^2}$ accordingly, which could give us the best results based on our experiments. Changing the input size when the node size remains unchanged (smaller or larger than $\frac{h_0 w_0}{16^2}$) will lead to a performance decline. Table 8 presents the details of the evaluation results where four models are trained on various input and node settings. Smaller graph size (nodes $= 16^2$) and input size ($256 \times 256 \times 3$) can lead to very fast inference speed but also result in significantly worse accuracy on small objects such as cars. Larger node size (nodes $= 32^2$or 48^2) considerably slows down the inference speed without obtaining better performance (-0.4% in terms of mF1-score). Hence, our model with node size of 28^2 achieves the best results on the Vaihingen dataset with very fast inference speed (113 FPS on a GPU). Note that the node size has no effect on the number of parameters of the model.

Table 8. Test performance of different input sizes and node sizes on Vaihingen test set.

Input	Nodes	GFLOPs	FPS	OA	Δ%	Building	Δ%	Car	Δ%	mF1	Δ%
$256 \times 256 \times 3$	16^2	4.37	178	0.903	-0.1	0.944	-0.4	0.858	-2.2	0.893	-0.5
$512 \times 512 \times 3$	32^2	17.5	91	0.901	-0.3	0.948	-	0.875	-0.5	0.894	-0.4
$768 \times 768 \times 3$	48^2	39.4	44	0.901	-0.3	0.946	-0.2	0.870	-1.0	0.893	-0.5
$448 \times 448 \times 3$	28^2	13.4	113	**0.904**	-	**0.948**	-	**0.880**	-	**0.898**	-

*FPS means frames per second on GPU (i.e., NVIDIA 1080Ti GPU in this work).

5.4.3. The learned graphs

We consider that the effectiveness and efficiency of our SCG-Net is mainly relying on the proposed SCG module that is able to construct the underlying non-local contextual relations in an end-to-end learnable manner. We visualized the SCG modules learned node graphs as 2D relation maps (see Figure 4). We highlight 6 representative nodes/squares (labeled with 8, 86, 165, 210, 624 and 738 separately) marked with 6 circles on both the input image and the ground truth. There are 6 relation maps from left to the right and top to bottom representing the learned relation graphs for the six nodes respectively, where dark blue regions represent weak dependency to the corresponding node, while light color blocks indicate strong relations to the target node. The relation-map visualizes the learned top-9 weighted-relationships of the target node (circled square patch) w.r.t its global contextual nodes (light-colored patches in the images), with gradient color palettes from light green to dark blue indicating the transition from strong dependency to zero-dependency. We observe that the target node has been able to interact with long-range neighbor nodes via the learned relation graph. In addition, when the SCG (the learned node graph) is discarded from the model as shown in Table 7, the performance of the model is greatly reduced (e.g. on Car -5.1% and mF1 -2.4%), and also the number of training steps dramatically increases by 5 times. While the learned graphs by SCG can benefit semantic segmentation tasks in terms of training process and performance, future studies are required to enhance the interpretability of the learned dependencies.

Table 9. Comparisons of SCG-Net variants on the hold-out IRRG test images of Vaihingen Dataset.

SCG-Net Variants	OA	Surface	Building	Low-veg	Tree	Car	mF1
SCG-GIN	0.901	0.922	0.947	0.836	0.894	0.877	0.895
SCG-GCN-GIN	0.899	0.920	0.946	0.828	0.893	**0.888**	0.895
SCG_{ae}-GCN	0.897	0.919	0.942	0.827	0.892	0.880	0.892
SCG_{dir}-GCN	0.902	0.923	0.947	0.833	0.895	0.881	0.896
SCG_{dir}-GCN-GIN	0.900	0.920	0.946	0.832	0.895	0.886	0.896
$\text{SCG-GCN}_{\text{\_ns}}$	0.899	0.919	0.946	0.834	0.892	0.860	0.890
SCG-GCN	**0.904**	**0.924**	**0.948**	**0.839**	**0.897**	0.880	**0.898**

5.4.4. SCG-Net variants

Additionally, we evaluated variants of the SCG-Net with different configuration settings as shown in Table 3. All models except SCG_{ae}-GCN and $\text{SCG-GCN}_{\text{\_ns}}$ demonstrated very close performance ($\Delta \sim \pm 0.3\%$ in terms of mF1-score) on the test set of Vaihingen dataset as shown in Table 9, and we further observe that when GCN and GIN are combined together in the framework, the models, such as SCG-GCN-GIN and

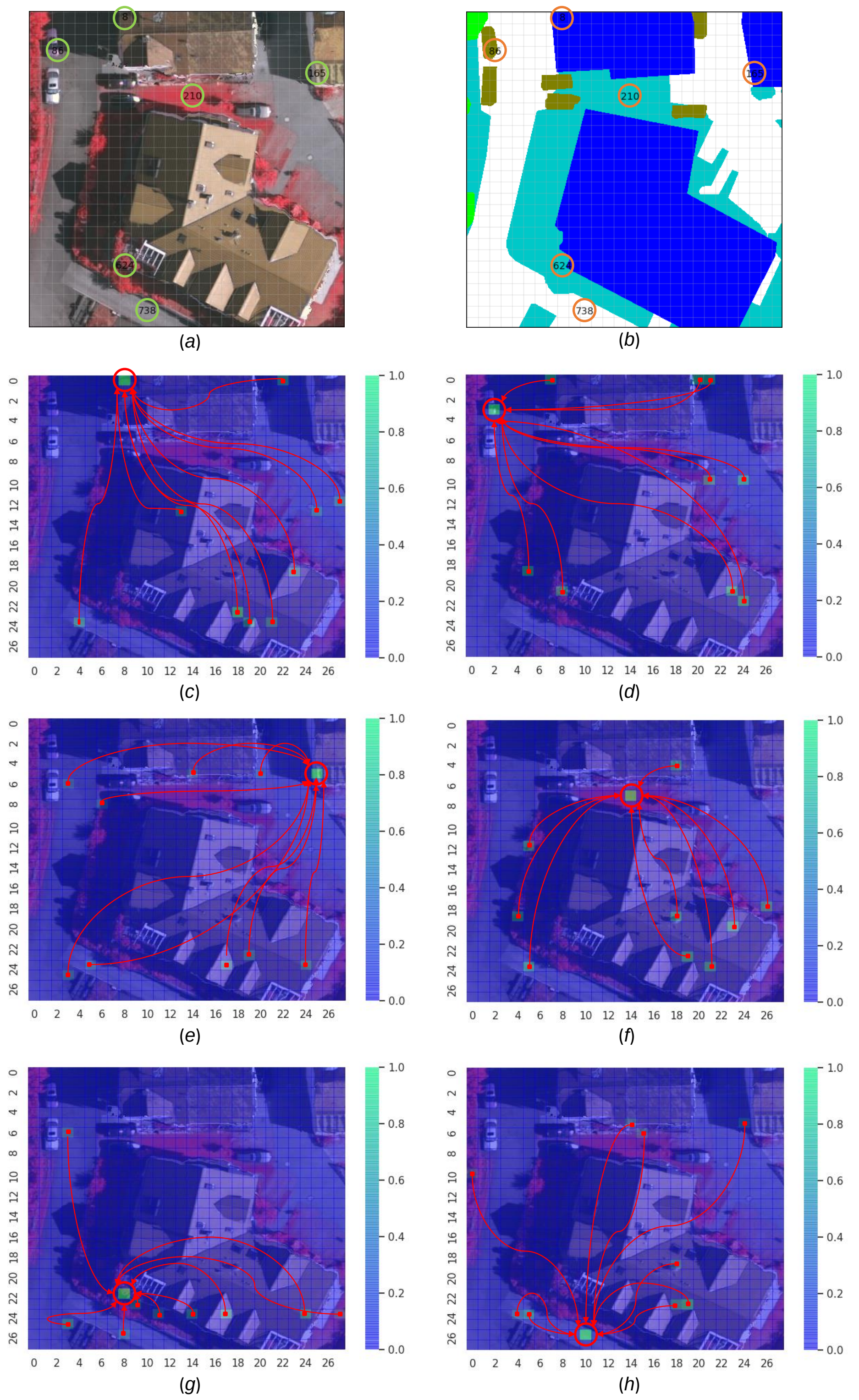

Figure 4. Visualization of the learned node graphs (28^2 nodes) onto 2D relation maps: (a) the input image, (b) the ground truth, (c) the visualized relation maps of node-8, (d) node-86, (e) node-165, (f) node-210, (g) node-624, (h) node-738. Dark blue regions represent weak dependency to the corresponding node, while light color blocks indicate strong relations to the node. Note that we normalized the edge weights to $[0.0, 1.0]$ and only illustrated the top nine largest weights associated to the target node.

SCG$_{\mathrm{dir}}$-GCN-GIN, obtained considerably better results on the small object (i.e, cars $+0.6 \sim 0.8\%$).

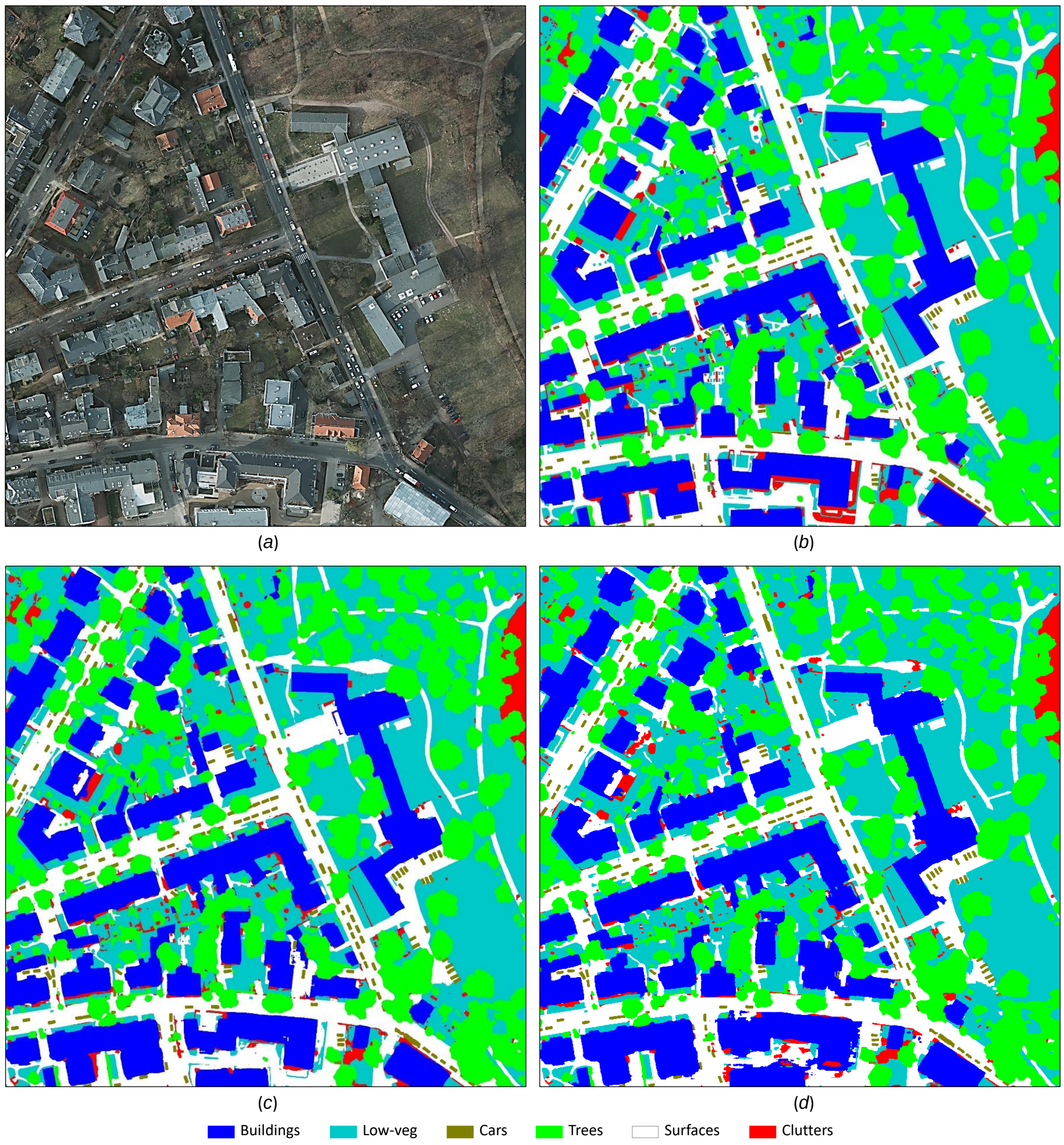

Figure 5. Segmentation results for the test image of Potsdam tile-3_14: (a) the test image, (b) the ground truth, (c) the predictions of DDCM-R50, (d) the predictions of our SCG-GCN.

5.4.5. Limitations

The performance of GNNs is known to gradually decrease with increasing number of GNN layers partly due to its over-smoothing issue, where repeatedly applying graph convolutions eventually makes features of vertices indistinguishable. Our model therefore exploits only 2-layer GNNs as the decoder and final semantic prediction simultaneously. Despite its promising performance, stacking more GNN layers either in the decoder or for the prediction significantly hurts the training and test performance of our model. In addition, we observed that the segmentation performance on the boundaries of small and dense objects (e.g, cars) was not as good as the baseline DDCM

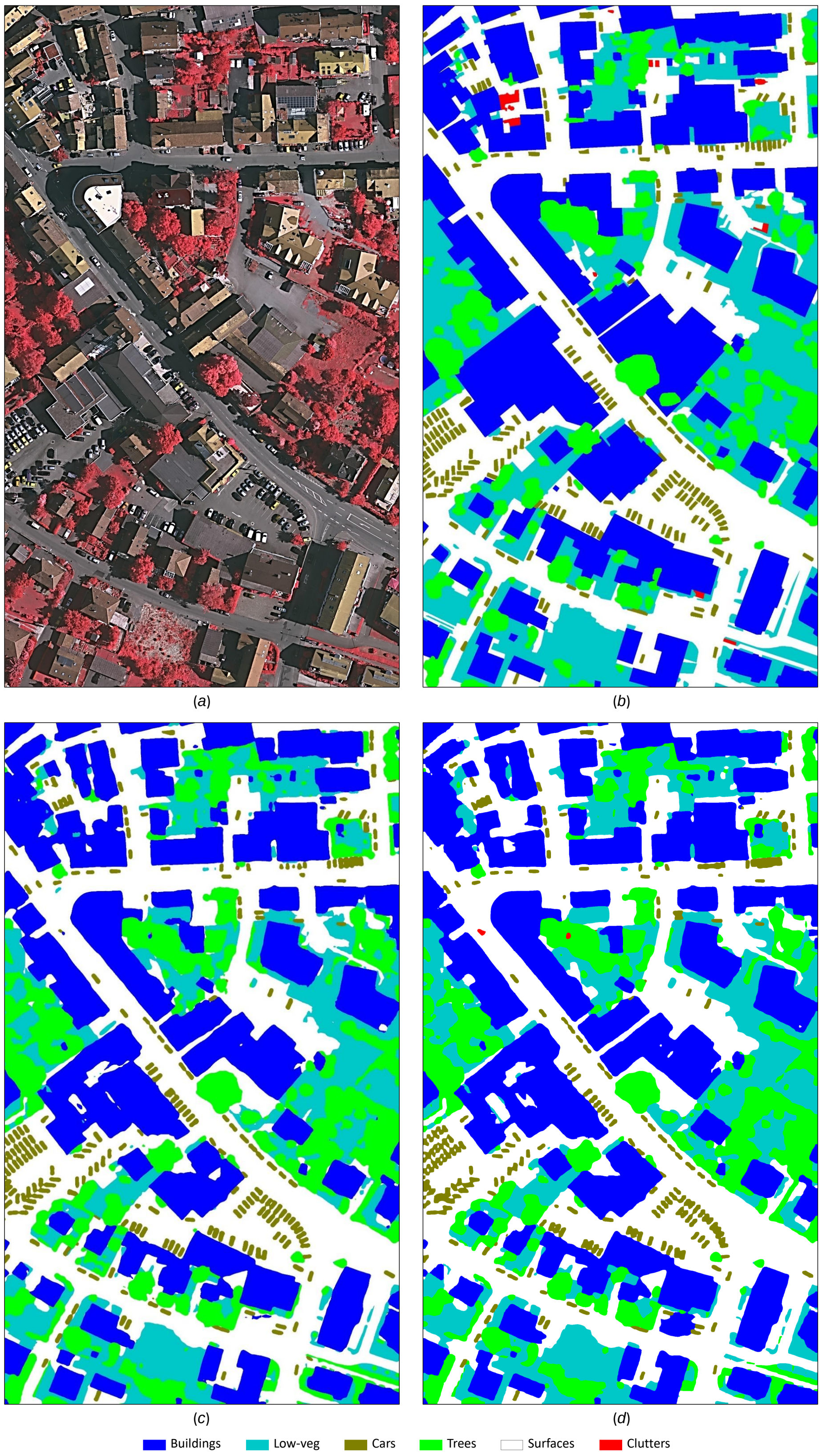

Figure 6. Segmentation results for the test image of Vaihingen tile-27: (a) the test image, (b) the ground truth, (c) the predictions of DDCM-R50, (d) the predictions of our SCG-GCN.

model. The closely located small objects tend to be segmented as a whole big object (see Figure 6). Future studies are required to further improve the model's performance on small objects.

6. Conclusions

We presented a novel self-constructing graph (SCG) framework, which makes use of learnable latent variables to effectively construct the global context relations (latent graph structure) directly from the input feature maps. SCG can be used to efficiently model pixel-wise contextual dependencies at a local and global scale without relying on manually built prior knowledge graphs. Built upon SCG and graph convolutional networks (GCN), our end-to-end SCG-GCN model achieves very competitive performance on the publicly available ISPRS Potsdam and Vaihingen datasets, with much fewer parameters, and at a lower computational cost compared to strong baseline models. Our comprehensive ablation experiments on remote sensing images also demonstrate that our methods are able to efficiently obtain long-range contextual information and improve performance by fully leveraging the benefits of both CNNs and variants of GNNs.

Acknowledgement(s)

The benchmark datasets were provided by the International Society for Photogrammetry and Remote Sensing (ISPRS). This work was supported by the foundation of the Research Council of Norway under Grant 272399 and Grant 309439.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the foundation of the Research Council of Norway under Grant 272399 and Grant 309439.

References

Achanta, R., A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. 2012. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (11): 2274–2282.

Adelipour, Sadjad, and Hassan Ghassemian. 2019. "Building extraction from very high-resolution synthetic aperture radar images based on statistical and structural information fusion." *International Journal of Remote Sensing* 40 (18): 7113–7126. https://doi.org/10.1080/01431161.2019.1601280.

Audebert, Nicolas, Bertrand Le Saux, and Sébastien Lefèvre. 2016. “Semantic segmentation of earth observation data using multimodal and multi-scale deep networks.” In *Asian Conference on Computer Vision*, 180–196. Springer.

Audebert, Nicolas, Bertrand Le Saux, and Sébastien Lefèvre. 2017. “Joint learning from earth observation and openstreetmap data to get faster better semantic maps.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 67–75.

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. 2017. “SegNet: A deep convolutional encoder-decoder architecture for image segmentation.” *IEEE Trans. Pattern Anal. Mach. Intell.* 39 (12): 2481–2495.

Bronstein, Michael M, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. “Geometric deep learning: going beyond euclidean data.” *IEEE Signal Processing Magazine* 34 (4): 18–42.

Buslaev, Alexander, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. 2020. “Albumentations: Fast and Flexible Image Augmentations.” *Information* 11 (2). https://www.mdpi.com/2078-2489/11/2/125.

Cao, Yue, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. 2019. “GCNet: Non-Local Networks Meet Squeeze-Excitation Networks and Beyond.” In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 1971–1980.

Chen, Jie, Yarong Han, Li Wan, Xing Zhou, and Min Deng. 2019. “Geospatial relation captioning for high-spatial-resolution images by using an attention-based neural network.” *International Journal of Remote Sensing* 40 (16): 6482–6498. https://doi.org/10.1080/01431161.2019.1594439.

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2018. “DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs.” *IEEE Trans. Pattern Anal. Mach. Intell.* 40 (4): 834–848.

Chiang, Wei-Lin, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. “Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks.” *arXiv preprint arXiv:1905.07953* .

Chung, Fan, Linyuan Lu, and Van Vu. 2003. “Spectra of random graphs with given expected degrees.” *Proceedings of the National Academy of Sciences* 100 (11): 6313–6318. https://www.pnas.org/content/100/11/6313.

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. 2016. “Convolutional neural networks on graphs with fast localized spectral filtering.” In *Advances in neural information processing systems*, 3844–3852.

Fey, Matthias, and Jan E. Lenssen. 2019. “Fast Graph Representation Learning with PyTorch Geometric.” In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, .

Fu, Jun, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. 2019. “Dual attention network for scene segmentation.” In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3146–3154.

Garcia, Victor, and Joan Bruna. 2017. “Few-shot learning with graph neural networks.” *arXiv preprint arXiv:1711.04043* .

Gori, Marco, Gabriele Monfardini, and Franco Scarselli. 2005. “A new model for learning in graph domains.” In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, Vol. 2, 729–734. IEEE.

Hamilton, Will, Zhitao Ying, and Jure Leskovec. 2017. “Inductive representation learning on large graphs.” In *Advances in Neural Information Processing Systems*, 1024–1034.

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

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep residual learning for image recognition.” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hu, Han, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. 2018. “Relation networks for object detection.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3588–3597.

Huang, Wenbing, Tong Zhang, Yu Rong, and Junzhou Huang. 2018. "Adaptive sampling towards fast graph representation learning." In *Advances in neural information processing systems*, 4558–4567.

Islam, Md. Amirul, Sen Jia, and Neil D. B. Bruce. 2020. "How much Position Information Do Convolutional Neural Networks Encode?" In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net. https://openreview.net/forum?id=rJeB36NKvB.

Jiang, Jie, Chengjin Lyu, Siying Liu, Y. He, and Xuetao Hao. 2020. "RWSNet: a semantic segmentation network based on SegNet combined with random walk for remote sensing." *International Journal of Remote Sensing* 41: 487 – 505.

Kampffmeyer, Michael C. 2018. "Advancing Segmentation and Unsupervised Learning Within the Field of Deep Learning." PhD dissertation, UiT The Arctic University of Norway. https://munin.uit.no/handle/10037/14264.

Kampffmeyer, Michael C., Yinbo Chen, Xiaodan Liang, Hao Wang, Yujia Zhang, and Eric P Xing. 2019. "Rethinking knowledge graph propagation for zero-shot learning." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11487–11496.

Kampffmeyer, Michael C., Arnt-Børre Salberg, and Robert Jenssen. 2018. "Urban Land Cover Classification With Missing Data Modalities Using Deep Convolutional Neural Networks." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11 (6): 1758–1768.

Kingma, Diederik P., and Jimmy Ba. 2014. "Adam: A Method for Stochastic Optimization." *CoRR* abs/1412.6980. http://arxiv.org/abs/1412.6980.

Kingma, Diederik P, and Max Welling. 2013. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* .

Kipf, Thomas N, and Max Welling. 2016a. "Semi-supervised classification with graph convolutional networks." *arXiv preprint arXiv:1609.02907* .

Kipf, Thomas N, and Max Welling. 2016b. "Variational graph auto-encoders." *arXiv preprint arXiv:1611.07308* .

Knyazev, Boris, Xiao Lin, Mohamed R Amer, and Graham W Taylor. 2019. "Image Classification with Hierarchical Multigraph Networks." *arXiv preprint arXiv:1907.09000* .

Landrieu, Loic, and Martin Simonovsky. 2018. "Large-scale point cloud semantic segmentation with superpoint graphs." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4558–4567.

Li, Yansheng, Ruixian Chen, Yongjun Zhang, and Hang Li. 2020. "A CNN-GCN Framework for Multi-Label Aerial Image Scene Classification." In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, 1353–1356.

Liang, Xiaodan, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P Xing. 2018. "Symbolic graph reasoning meets convolutions." In *Advances in Neural Information Processing Systems*, 1853–1863.

Lin, Guosheng, Chunhua Shen, Anton Van Den Hengel, and Ian Reid. 2016. "Efficient piecewise training of deep structured models for semantic segmentation." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3194–3203.

Liu, Qinghui, Michael C. Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. 2019a. "Dense Dilated Convolutions Merging Network for Semantic Mapping of Remote Sensing Images." *2019 Joint Urban Remote Sensing Event (JURSE)* .

Liu, Qinghui, Michael C. Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. 2020a. "Dense Dilated Convolutions' Merging Network for Land Cover Classification." *IEEE Transactions on Geoscience and Remote Sensing* 58 (9): 6309–6320.

Liu, Qinghui, Michael C. Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. 2020b. "Multi-View Self-Constructing Graph Convolutional Networks With Adaptive Class Weighting Loss for Semantic Segmentation." In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 199–205.

Liu, Qinghui, Michael C. Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. 2020c. "Self-Constructing Graph Convolutional Networks for Semantic Labeling." In *Proceedings of*

IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium, .

Liu, Qinghui, Arnt-Børre Salberg, and Robert Jenssen. 2018. “A Comparison of Deep Learning Architectures for Semantic Mapping of Very High Resolution Images.” In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, July, 6943–6946.

Liu, Yansong, Sankaranarayanan Piramanayagam, Sildomar T Monteiro, and Eli Saber. 2019b. “Semantic segmentation of multisensor remote sensing imagery with deep ConvNets and higher-order conditional random fields.” *Journal of Applied Remote Sensing* 13 (1): 016501.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell. 2015. “Fully Convolutional Networks for Semantic Segmentation.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Marino, Kenneth, Ruslan Salakhutdinov, and Abhinav Gupta. 2016. “The more you know: Using knowledge graphs for image classification.” *arXiv preprint arXiv:1612.04844* .

Marmanis, Dimitrios, Konrad Schindler, Jan Dirk Wegner, Silvano Galliani, Mihai Datcu, and Uwe Stilla. 2016. “Classification With an Edge: Improving Semantic Image Segmentation with Boundary Detection.” *CoRR* abs/1612.01337. http://arxiv.org/abs/1612.01337.

Milletari, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. “V-net: Fully convolutional neural networks for volumetric medical image segmentation.” In *2016 Fourth International Conference on 3D Vision (3DV)*, 565–571. IEEE.

Mou, Lichao, and Xiao Xiang Zhu. 2018. “RiFCN: Recurrent Network in Fully Convolutional Network for Semantic Segmentation of High Resolution Remote Sensing Images.” *CoRR* abs/1805.02091. http://arxiv.org/abs/1805.02091.

Niepert, Mathias, Mohamed Ahmed, and Konstantin Kutzkov. 2016. “Learning convolutional neural networks for graphs.” In *International conference on machine learning*, 2014–2023.

Ouyang, Song, and Yansheng Li. 2021. “Combining Deep Semantic Segmentation Network and Graph Convolutional Neural Network for Semantic Segmentation of Remote Sensing Imagery.” *Remote Sensing* 13 (1). https://www.mdpi.com/2072-4292/13/1/119.

Paisitkriangkrai, Sakrapee, Jamie Sherrah, Pranam Janney, Van-Den Hengel, et al. 2015. “Effective semantic pixel labelling with convolutional networks and conditional random fields.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 36–43.

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. “Automatic differentiation in PyTorch.” In *NIPS-W*, .

Peng, Chao, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. 2017. “Large Kernel Matters–Improve Semantic Segmentation by Global Convolutional Network.” *arXiv preprint arXiv:1703.02719* .

Qi, Xiaojuan, Renjie Liao, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. 2017. “3d graph neural networks for rgbd semantic segmentation.” In *Proceedings of the IEEE International Conference on Computer Vision*, 5199–5208.

Reddi, Sashank J, Satyen Kale, and Sanjiv Kumar. 2018. “On the convergence of Adam and beyond.” *arXiv preprint arXiv:1904.09237* .

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. 2015. “U-Net: Convolutional networks for biomedical image segmentation.” In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234–241. Springer.

Rottensteiner, Franz, Gunho Sohn, Jaewook Jung, Markus Gerke, Caroline Baillard, Sebastien Benitez, and Uwe Breitkopf. 2012. “The ISPRS benchmark on urban object classification and 3D building reconstruction.” *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences I-3 (2012), Nr. 1* 1 (1): 293–298.

Sherrah, Jamie. 2016. “Fully Convolutional Networks for Dense Semantic Labelling of High-Resolution Aerial Imagery.” *CoRR* abs/1606.02585. http://arxiv.org/abs/1606.02585.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. “Attention is All you Need.” In *Advances in Neural Information Processing Systems 30*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, 5998–6008. Curran Associates, Inc. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Velickovic, Petar, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. "Deep Graph Infomax." In *ICLR (Poster)*, .

Wang, Hongzhen, Ying Wang, Qian Zhang, Shiming Xiang, and Chunhong Pan. 2017a. "Gated convolutional neural network for semantic segmentation in high-resolution images." *Remote Sensing* 9 (5): 446.

Wang, Panqu, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. 2017b. "Understanding convolution for semantic segmentation." *arXiv preprint arXiv:1702.08502* .

Wang, Xiaolong, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. "Non-local neural networks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7794–7803.

Wang, Yue, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. 2019. "Dynamic graph cnn for learning on point clouds." *ACM Transactions on Graphics (TOG)* 38 (5): 146.

Xu, Keyulu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. "How Powerful are Graph Neural Networks?" In *International Conference on Learning Representations*, .

Zhang, Hang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. 2018. "Context encoding for semantic segmentation." In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 7151–7160.

Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 2016. "Pyramid scene parsing network." *arXiv preprint arXiv:1612.01105* .

Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2015. "Object detectors emerge in deep scene cnns." *IEEE International Conference on Learning Representation (ICLR)* .

Zhou, Kaixiong, Xiao Huang, Yuening Li, Daochen Zha, Rui Chen, and Xia Hu. 2020. "Towards Deeper Graph Neural Networks with Differentiable Group Normalization." *arXiv preprint arXiv:2006.06972* .