

SELF-CONSTRUCTING GRAPH CONVOLUTIONAL NETWORKS FOR SEMANTIC LABELING

Qinghui Liu^{1,2}, Michael Kampffmeyer², Robert Jenssen^{2,1}, Arnt-Børre Salberg¹

¹Norwegian Computing Center, Dept. SAMBA, NO-0314 OSLO, Norway

²UiT Machine Learning Group, UiT the Arctic University of Norway, Tromsø, Norway

ABSTRACT

Graph Neural Networks (GNNs) have received increasing attention in many fields. However, due to the lack of prior graphs, their use for semantic labeling has been limited. Here, we propose a novel architecture called the Self-Constructing Graph (SCG), which makes use of learnable latent variables to generate embeddings and to self-construct the underlying graphs directly from the input features without relying on manually built prior knowledge graphs. SCG can automatically obtain optimized non-local context graphs from complex-shaped objects in aerial imagery. We optimize SCG via an adaptive diagonal enhancement method and a variational lower bound that consists of a customized graph reconstruction term and a Kullback-Leibler divergence regularization term. We demonstrate the effectiveness and flexibility of the proposed SCG on the publicly available ISPRS Vaihingen dataset and our model SCG-Net achieves competitive results in terms of F1-score with much fewer parameters and at a lower computational cost compared to related pure-CNN based work.

Index Terms— Self-Constructing Graph (SCG), Graph Convolutional Networks (GCNs), semantic labeling

1. INTRODUCTION

Recently, graph neural networks (GNNs) [1] and Graph Convolutional Networks (GCNs) [2] have received increasing attention, partially due to their superior performance for many node or graph classification tasks in the non-Euclidean domain, including graphs and manifolds. Variants of GNNs and GCNs have been applied to computer vision tasks, among others, image classification [3], few-shot and zero-shot classification [4], point clouds classification [5] and semantic segmentation [6]. However, graph reasoning for vision tasks is quite sensitive to how the graph of relations between objects is built and previous approaches commonly rely on manually built graphs based on prior knowledge. Inspired by variational graph auto-encoders [7], we instead propose a novel Self-Constructing Graph module (SCG) to *learn* how a 2D fea-

ture 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 our proposed self-constructing graph convolutional network (SCG-Net), the SCG is followed by Graph Convolutional Networks (GCNs) [2] to update the node features along the edges of the graph. After K-layer of GCNs, the vertices are projected back onto the 2D plane. The SCG module can be easily embedded into existing CNN and GCN networks for computer vision tasks. Our model can be trained end-to-end since every step is fully differentiable. Our experiments demonstrate that the network achieves robust and competitive results on the representative ISPRS 2D semantic labeling Vaihingen benchmark datasets [8].

2. METHODS

We first briefly revisit some concepts of graph convolutions. We then present the details of the proposed self-constructing graph (SCG) algorithm and our end-to-end trainable model SCG-Net for semantic labeling tasks.

2.1. Graph Convolution

Definitions: We consider an undirected graph $G = (A, X)$, which consists of n vertices, where $A \in \mathbb{R}^{n \times n}$ is the adjacency matrix with (i, j) entry A_{ij} is 1 if there is an edge between i and j and 0 otherwise, and $X \in \mathbb{R}^{n \times d}$ is the node feature matrix for all vertices assuming each node has d features. Given a set of labeled nodes $D = \{(x_i, y_i)\}_{i=1}^m$ where $x_i \in X$, and $y_i \in Y$ contains the labels for the labeled nodes.

GCN: Graph Convolutional Networks (GCNs) [2] were originally proposed for semi-supervised classification ($m = |Y| \leq n$). Thus, $m = |Y| \leq n$ for semi-supervised node classification settings. GCN implements a "message-passing" function by a combination of linear transformations over one-hop neighbourhoods followed by a non-linearity:

$$Z^{(l+1)} = \sigma \left(\hat{A} X^{(l)} \theta^{(l)} \right), \quad (1)$$

where \hat{A} is the symmetric normalization of A with self-loops:

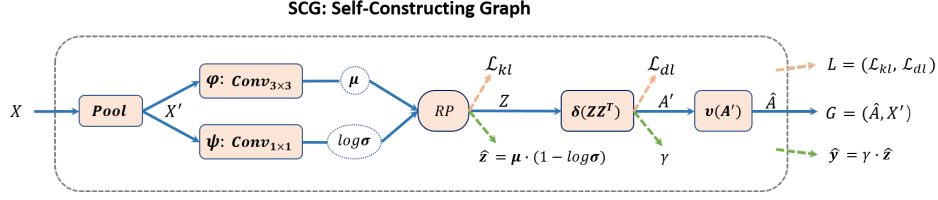


Fig. 1. The illustration diagram of the SCG module, where \hat{A} is the normalized adjacency matrix, X' corresponds to node features, Z are the latent embeddings, ε are learnable weights, \mathcal{L}_{kl} is the KullbackLeibler divergence loss and \mathcal{L}_{dl} is the diagonal log loss. γ is the adaptive factor, \hat{z} are auxiliary embeddings, RP is the re-parameterization operations and \hat{y} are residual predictions.

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

where $D_{ii} = \sum_j (A + I)_{ij}$ is a diagonal matrix with node degrees and I is an identity matrix, and σ denotes the non-linearity function (e.g. *ReLU*).

In the following sections, we will use $Z^{(K)} = \text{GCN}(A, X)$ to denote an arbitrary GCN module implementing K steps of message passing based on some adjacency matrix A and input node features X , where K is common in the range 2-6 in practice.

2.2. Self-Constructing Graph

We propose the Self-Constructing Graph (SCG) framework for learning latent graph representations directly from 2D feature maps. This model makes use of re-parameterized latent variables and is capable of constructing undirected graphs without relying on prior graph information (see Figure 1). We assume an input 2D feature map X of size $h \times w$ with d features. $X \in \mathbb{R}^{h \times w \times d}$ are usually the high-level features learned by deep convolutional networks. The main goal of our SCG module is to learn a latent graph from the input feature maps to capture the long-range relations among vertices. Formally, $G = \text{SCG}(X)$, where $G = (\hat{A}, X')$, and $\hat{A} \in \mathbb{R}^{n \times n}$ is a weighted adjacency matrix, $X' \in \mathbb{R}^{n \times d}$ is the node features, and $n = h' \times w'$ denotes the number of nodes. Note that usually $(h' \times w') \leq (h \times w)$ in practice.

In this work, we take a parameter-free pooling operation (e.g. adaptive avg pooling) to transform X to X' and then constraint the size of vertices to be n .

Encoder to a latent space: In the encoding part, Gaussian parameters (the mean matrix $\mu \in \mathbb{R}^{n \times c}$ and the standard deviation matrix $\sigma \in \mathbb{R}^{n \times c}$, where c denotes the number of labels.) are learned from two single-layer Conv networks (where the subscript indicates of Conv denote the size of the filters as shown in the following two formulas).

$$\mu \leftarrow \varphi(X') = \text{Conv}_{3 \times 3}(X')$$

$$\sigma \leftarrow \exp(\psi(X')) = \exp(\text{Conv}_{1 \times 1}(X'))$$

Note that X' is compatible with regular convolutional networks by simply reshaping it from $\mathbb{R}^{n \times d}$ to $\mathbb{R}^{h' \times w' \times d}$.

And similarly, the outputs of Conv are reshaped back from $\mathbb{R}^{h' \times w' \times c}$ to $\mathbb{R}^{n \times c}$.

Reparameterization: In order to keep the proposed architecture end-to-end trainable, we perform a reparameterization of the latent embeddings. The latent embedding Z is computed as

$$Z \leftarrow \mu + \sigma \cdot \varepsilon$$

, where $\varepsilon \in \mathbb{R}^{n \times c}$ is an auxiliary noise variable that is initialized from a standard normal distribution ($\varepsilon \sim \text{N}(0, I)$).

Here, we also introduce auxiliary embeddings \hat{z} which is defined as: $\hat{z} \leftarrow \mu \cdot (1 - \log \sigma)$, which will be used later to computer the residual predictions.

We further, during the training phase, regularize the latent variables by minimizing the Kullback-Leibler divergence between the embedding and a centered isotropic multivariate Gaussian prior distribution [9] $\mathcal{L}_{kl} \leftarrow KL(\mu, \sigma)$, which is given as

$$KL(\mu, \sigma) \simeq -\frac{1}{2n} \sum_{i=1}^n \left(1 + \log(\sigma_i)^2 - \mu_i^2 - \sigma_i^2 \right). \quad (3)$$

Decoder to output space: The learned graph A' is given by the inner product between the latent embeddings

$$A' \leftarrow \delta(ZZ^T) = \text{ReLU}(ZZ^T)$$

Note, $A'_{ij} > 0$ denotes that there is an edge between nodes i and j . Intuitively, we consider A'_{ii} shall be > 0 . We therefore introduce the following diagonal log regularization term

$$\mathcal{L}_{dl} \leftarrow DL(A') = -\frac{\gamma}{n^2} \sum_{i=1}^n \log(|A'_{ii}|_{[0,1]} + \epsilon), \quad (4)$$

where γ is an adaptive factor which is defined as

$$\gamma = \sqrt{1 + \frac{n}{\sum_{i=1}^n (A'_{ii}) + \epsilon}}. \quad (5)$$

We also propose an adaptive diagonal enhancement approach to better maintain the learned neighborhoods information resulting in

$$A' \leftarrow A' + \gamma \cdot \text{diag}(A'). \quad (6)$$

We finally obtain the symmetric normalized \hat{A} w.r.t the enhanced A' by

$$\hat{A} \leftarrow \nu(A') = D^{-\frac{1}{2}} (A' + \gamma \cdot \text{diag}(A') + I) D^{\frac{1}{2}}. \quad (7)$$

Additionally, we also propose a residual term, the so-called adaptive residual prediction \hat{y} which is defined as: $\hat{y} \leftarrow \gamma \cdot \hat{z}$, to be used later for refining the final predictions of the networks.

2.3. The SCG-Net

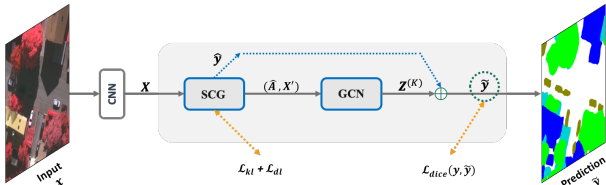


Fig. 2. Model architecture of SCG-Net for semantic labeling includes the CNN-based feature extractor (e.g. customized ResNet50 output 1024-channel), SCG module and K-layer GCNs (K=2 in this work), the fused (element-wise sum) output is projected back to 2D maps for final prediction.

The SCG module can be easily incorporate into existing CNN and GCN architectures in order to exploit the advantages of both the CNN to learn feature detectors, while at the same time exploit the ability of GCNs to model long-range relations. Fig. 2 shows our so-called SCG-Net, which combines SCG with CNNs and GCNs to address the semantic labeling task. Following our previous work [10], we utilize the first three bottleneck layers of a pretrained ResNet50 [11] as the backbone CNN to learn the high-level representations. A 2-layer GCN (Equation 1) is used in our model and we utilize ReLU activation and batch normalization only in the first layer of the GCN.

3. EXPERIMENTS AND RESULTS

We train and evaluate our proposed methods on a publicly available benchmark dataset, namely the ISPRS 2D Vaihingen semantic labeling contest dataset. The Vaihingen dataset contains 33 tiles of varying size (on average approximately 2100×2100 pixels) with a ground resolution of 9cm, of which 17 are used as hold-out test images. We follow the training settings of our previous work [12] to train our model, and apply a dice loss function [13] and two regularization terms \mathcal{L}_{kl} and \mathcal{L}_{dl} as defined in the equations 3 and 4. The overall cost function of our model is therefore defined as

$$\mathcal{L} \leftarrow \mathcal{L}_{dice} + \mathcal{L}_{kl} + \mathcal{L}_{dl}. \quad (8)$$

We train and validate the networks with 4000 randomly sampled patches of size 448×448 as input and train it using minibatches of size 4. The training data is sampled uniformly and randomly shuffled for each epoch.

Results: We evaluated our trained model on the hold-out test sets (17 images) in order to fairly compare to other related published work on the same test sets. These results are shown in Table 1. Our model obtained very competitive performance with 89.8% F1-score which is around 1.1% higher than GSN [14] and the same as the best performing model DDCM-R50 [10]. However, the proposed model consists of fewer training parameters (8.74 million vs. 9.99 million for the DDCM-R50 model) and has lower computational cost (4.37 Giga FLOPs vs. 4.86 Giga FLOPs for the DDCM-R50 model), resulting in faster training performance. Fig. 3 shows the qualitative comparisons of the land cover mapping results from our model and the ground truths on the test set.

Table 1. Comparisons between our method with other published methods on the hold-out IRRG test images of ISPRS Vaihingen Dataset.

Models	OA	Surface	Building	Low-veg	Tree	Car	mF1
ONE_7 [15]	0.898	0.910	0.945	0.844	0.899	0.778	0.875
DLR_9 [16]	0.903	0.924	0.952	0.839	0.899	0.812	0.885
GSN [14]	0.903	0.922	0.951	0.837	0.899	0.824	0.887
DDCM-R50 [10]	0.904	0.927	0.953	0.833	0.894	0.883	0.898
SCG-Net	0.904	0.924	0.948	0.839	0.897	0.880	0.898

4. CONCLUSIONS

In this paper, we presented a self-constructing graph (SCG) architecture which makes use of learnable latent variables to construct the hidden graphs directly from 2D feature maps with no prior graphs available. The proposed SCG network can be easily adapted and incorporated into existing deep CNNs and GCNs architectures to address a wide range of different problems. On the Vaihingen datasets, our SCG-Net model achieves competitive results, while making use of fewer parameters and being computationally more efficient.

5. REFERENCES

- [1] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst, “Geometric deep learning: going beyond euclidean data,” *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.
- [2] Thomas N Kipf and Max Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
- [3] Boris Knyazev, Xiao Lin, Mohamed R Amer, and Graham W Taylor, “Image classification with hierarchical multigraph networks,” *arXiv preprint arXiv:1907.09000*, 2019.

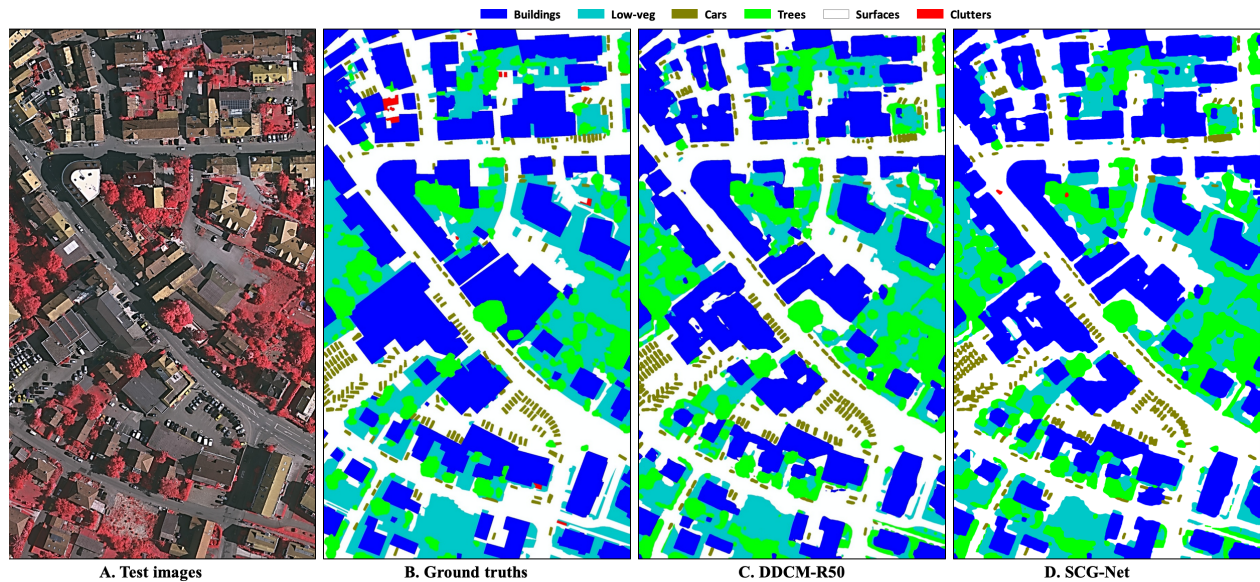


Fig. 3. Mapping results for test images of Vaihingen tile-27. From the left to right, the input images, the ground truths and the predictions of DDCM-R50, and our SCG-Net.

- [4] Michael Kampffmeyer, Yinbo Chen, Xiaodan Liang, Hao Wang, Yujia Zhang, and Eric P Xing, “Rethinking knowledge graph propagation for zero-shot learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11487–11496.
- [5] 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 (TOG)*, vol. 38, no. 5, pp. 146, 2019.
- [6] Xiaodan Liang, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P Xing, “Symbolic graph reasoning meets convolutions,” in *Advances in Neural Information Processing Systems*, 2018, pp. 1853–1863.
- [7] Thomas N Kipf and Max Welling, “Variational graph auto-encoders,” *arXiv preprint arXiv:1611.07308*, 2016.
- [8] International Society for Photogrammetry and Remote Sensing (ISPRS), “2D Semantic Labeling Contest,” online, 2018.
- [9] Diederik P Kingma and Max Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [10] Qinghui liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Borre Salberg, “Dense dilated convolutions merging network for semantic mapping of remote sensing images,” *2019 Joint Urban Remote Sensing Event (JURSE)*, May 2019.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [12] Q. Liu, A. Salberg, and R. Jenssen, “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 2018, pp. 6943–6946.
- [13] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi, “V-net: Fully convolutional neural networks for volumetric medical image segmentation,” in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 565–571.
- [14] Hongzhen Wang, Ying Wang, Qian Zhang, Shiming Xiang, and Chunhong Pan, “Gated convolutional neural network for semantic segmentation in high-resolution images,” *Remote Sensing*, vol. 9, no. 5, pp. 446, 2017.
- [15] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre, “Semantic segmentation of earth observation data using multimodal and multi-scale deep networks,” in *Asian Conference on Computer Vision*. Springer, 2016, pp. 180–196.
- [16] Dimitrios Marmanis, Konrad Schindler, Jan Dirk Wegner, Silvano Galliani, Mihai Datcu, and Uwe Stilla, “Classification with an edge: Improving semantic image segmentation with boundary detection,” *CoRR*, vol. abs/1612.01337, 2016.