UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Physics and Technology

Advancing Land Cover Mapping in Remote Sensing with Deep Learning

Qinghui Liu

A dissertation for the degree of Philosophiae Doctor - November 2021

Abstract

Automatic mapping of land cover in remote sensing data plays an increasingly significant role in several earth observation (EO) applications, such as sustainable development, autonomous agriculture, and urban planning. Due to the complexity of the real ground surface and environment, accurate classification of land cover types is facing many challenges. This thesis provides novel deep learning-based solutions to land cover mapping challenges such as how to deal with intricate objects and imbalanced classes in multi-spectral and high-spatial resolution remote sensing data.

The first work presents a novel model to learn richer multi-scale and global contextual representations in very high-resolution remote sensing images, namely the dense dilated convolutions' merging (DDCM) network. The proposed method is light-weighted, flexible and extendable, so that it can be used as a simple yet effective encoder and decoder module to address different classification and semantic mapping challenges. Intensive experiments on different benchmark remote sensing datasets demonstrate that the proposed method can achieve better performance but consume much fewer computation resources compared with other published methods.

Next, a novel graph model is developed for capturing long-range pixel dependencies in remote sensing images to improve land cover mapping. One key component in the method is the self-constructing graph (SCG) module that can effectively construct global context relations (latent graph structure) without requiring prior knowledge graphs. The proposed SCG-based models achieved competitive performance on different representative remote sensing datasets with faster training and lower computational cost compared to strong baseline models.

The third work introduces a new framework, namely the multi-view self-constructing graph (MSCG) network, to extend the vanilla SCG model to be able to capture multi-view context representations with rotation invariance to achieve improved segmentation performance. Meanwhile, a novel adaptive class weighting loss function is developed to alleviate the issue of class imbalance commonly found in EO datasets for semantic segmentation. Experiments on benchmark data demonstrate the proposed framework is computationally efficient and robust to produce improved segmentation results for imbalanced classes.

To address the key challenges in multi-modal land cover mapping of remote sensing data, namely, 'what', 'how' and 'where' to effectively fuse multi-source features and to efficiently learn optimal joint representations of different modalities, the last work presents a compact and scalable multi-modal deep learning framework

(MultiModNet) based on two novel modules: the pyramid attention fusion module and the gated fusion unit. The proposed MultiModNet outperforms the strong baselines on two representative remote sensing datasets with fewer parameters and at a lower computational cost. Extensive ablation studies also validate the effectiveness and flexibility of the framework.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my main supervisor Professor Robert Jenssen for his valuable guidance, support and optimism while conducting this research work as well as in the writing of this thesis. Thank you for guiding me through this journey and shaping my research development.

I would also like to extend my deepest thanks to my co-supervisors Dr. Arnt-Børre Salberg and Dr. Michael Kampffmeyer, who always had time to guide me with constant patience and encouragement during my Ph.D study and research. I really appreciate your involvement, valuable insight and constant attention to detail in this research work of this thesis. I would never have been able to finish this without your great advice and helpful discussions. Thank you!

I am also very grateful to the Research Council of Norway (RCN) and Norwegian Computing Center (NR) for funding this research project, specially to Rune Solberg and André Teigland for supporting this research work. I would like to thank all my colleagues at NR. It has been a great experience and privilege to be part of NR and JO Group where I had the opportunity to learn from a group of highly skilled researchers.

I would also like to express my gratitude to everyone in the Machine Learning Group at UiT. It was always inspiring and memorable while I was there for a few days with so many brilliant and helpful people.

I also would like to thank my committee members for taking the time to read my thesis and attending the defense.

Last but definitely not the least, I would especially like to thank my family, parents, siblings, and friends for your unconditional love and continued support throughout this experience.

Qinghui Liu,
Oslo, November 2021

Contents

List of Figures

1

Introduction

In recent years, the advances in remote sensing technologies and the fast-growing volume of remotely sensed data have dramatically changed the way we observe the Earth. One of the key applications in Earth observation is the classification[1] of the land cover and further monitoring of changes. Land cover mapping of the Earth is highly valuable in environmental monitoring [1, 2, 3], agriculture [4] and urban planning [5, 6], predicting natural disasters and hazardous events [7, 8, 9], etc. Figure. 1.1 shows some illustrative examples of land cover mapping for various remote sensing data. With the improvement of sensor technology, the quality of remotely sensed data has been greatly improved in terms of the spatial, spectral and temporal resolution. The availability of high-resolution remote sensing data can be significantly more effective to automatically extract on-Earth objects and map land cover and uses.

However, to effectively extract and exploit meaningful information from such big remote sensing data, special tools and methods are required [1]. Traditional approaches that mainly rely on hand-crafted features are very expensive, labor-intensive, and time-consuming. In the past few years, deep learning [13] techniques have demonstrated astounding capabilities in signal and data processing and often progressed beyond state-of-the-art performance on various tasks, such as, image classification [14] and segmentation [15, 16], object detection [17], speech recognition [18], and natural language understanding [19]. Currently, various deep learning approaches have been increasingly adapted for the intelligent interpretation of remote sensing data. As one of the key methods for automatic analysis and interpretation of remote sensing data, semantic mapping, or segmentation, aims to attribute each pixel to a single semantic label that is corresponding to a type of

[1] The term 'classification' in the field of remote sensing is often preferred instead of the term 'semantic segmentation' that is commonly used in computer vision [1]. In the thesis, the term 'classification' represents the same meaning as the term 'semantic segmentation' that is the pixel-wise classification or semantic mapping.

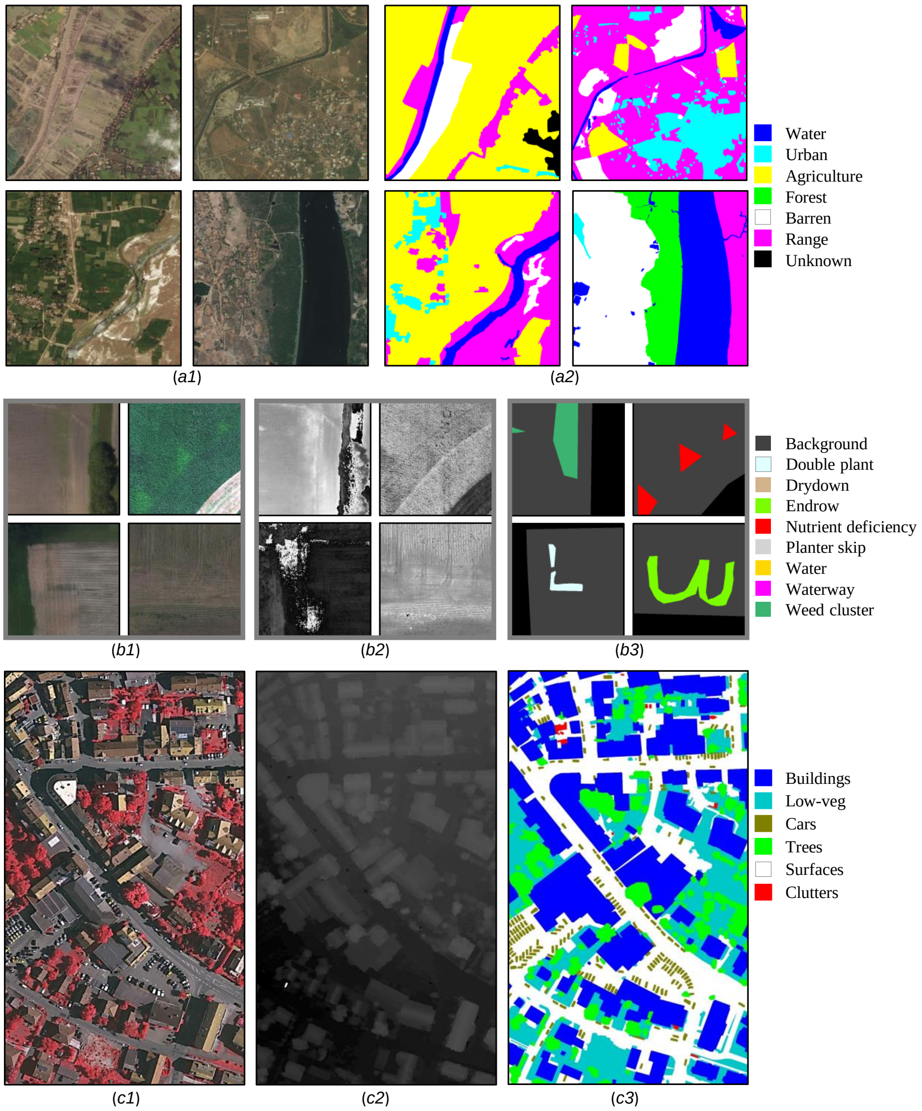


Figure 1.1: Examples of land cover mapping samples from three different datasets. Top: (a1) RGB images from the DeepGlobe dataset [10], (b1) the corresponding ground-truth images. Middle: (b1) RGB images from the Agriculture-vision dataset [11], (b2) the corresponding NIR-band images, (c1) the corresponding ground-truth images. Bottom: (c1) IRRG image from the Vaihingen dataset [12], (c2) the corresponding DSM image, (c3) the corresponding ground-truth image.

land cover[2].

The aim of this thesis is to contribute to the advances of deep learning methodologies for land cover mapping in remote sensing, and to find full or partial answers to some key challenges in the automatic analysis and interpretation of remote sensing data. These challenges are briefly outlined in Section 1.2 and will be treated in more detail in the corresponding papers.

1.1 Remote sensing imagery

Remote sensing is the process of capturing the physical characteristics of an area from a distance [20] with remote sensors or instruments on e.g. satellites, airplanes or UAVs (unmanned aerial vehicles). Many sensors acquire data at different spectral wavelengths, known as the electromagnetic spectrum, that range from short wavelengths (such as X-rays: $10^{-2} - 10$ nm) to long wavelengths (such as radio waves: $10 - 10^3$ m). Each region or segment of the spectrum is referred to as a band or channel. Our human eyes are only able to see small portion of the full spectrum from about 380 to about 750 nanometers [21], such as RGB bands: Red-band (0.45 – 0.51 um), Green-band (0.53 – 0.59 um) and Blue-band (0.64 – 0.67 um). Different bands can be combined together to produce imagery of the data in order to reveal different features in the landscape as shown in Figure 1.3.

Once remote sensing data are processed into imagery with varying band combinations, they allow us to visualize, analyze, and interpret objects and features on the Earth's surface for urban planning, measuring land cover and land-use change, tracking biodiversity, managing natural resources, and assessing disasters [20]. When we want to capture and evaluate remote sensing imagery for earth observation, we need to consider its resolution, referring to the potential detail provided by the imagery. In remote sensing, there are three types of resolution: spatial, spectral and temporal.

- **Spatial resolution** is defined by the size of each pixel in an image and the corresponding area on Earth's surface represented by that pixel. For example, an image that has a spatial resolution of 10m means that each pixel in the image represents a 10 × 10 meters area on the ground. Figure 1.2 shows examples of different spatial resolution images over the same area, ranging from 0.1 meter to 10 meter.

- **Spectral resolution** refers to the ability of a sensor to measure finer wavelengths of the electromagnetic spectrum, that is, having more and narrower bands. The major difference between **multi-spectral** and **hyper-spectral**[3] is the number of bands and how narrow the bands are. In this work, we

[2] The set of land cover classes varies between various applications and tasks.

[3] In general, multi-spectral images have between 3 to 10 wider bands, where each band commonly has a descriptive band title such as red, green, blue, near-infrared, short-wave infrared, and so on. Hyper-spectral images generally consist of hundreds or thousands of much narrower bands (10 – 20 nm) without specific band names.

mainly focus on multi-spectral remote sensing data. Figure 1.3 illustrates four commonly used bands in optical remote sensing data with different combinations.

- **Temporal resolution** is the amount of time it takes for a satellite to revisit and acquire data for the exact same geographical area. Airplanes or UAVs are flexible. But for satellites, this resolution depends on the orbit and latitude, the sensor's characteristics, and the swath width. For example, polar orbiting satellites have a temporal resolution that can vary from about 1 day to 16 or more days [20]. But it is also common that the orbiting satellites visit the same place twice a day but in different directions.

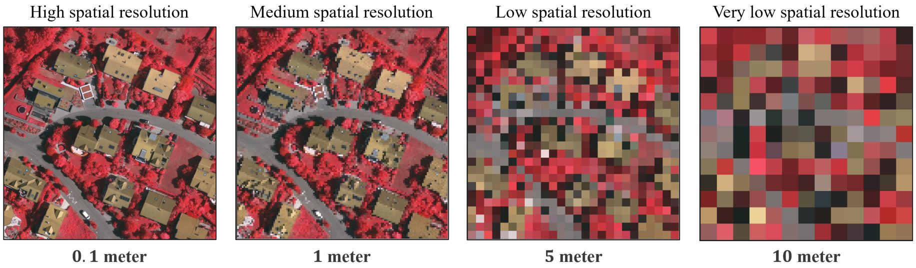


Figure 1.2: Examples of different spatial resolution images over the same region. From left to right, high spatial resolution (0.1 meter), medium spatial resolution (1 meter), low and very low spatial resolution (5 - 10 meter).

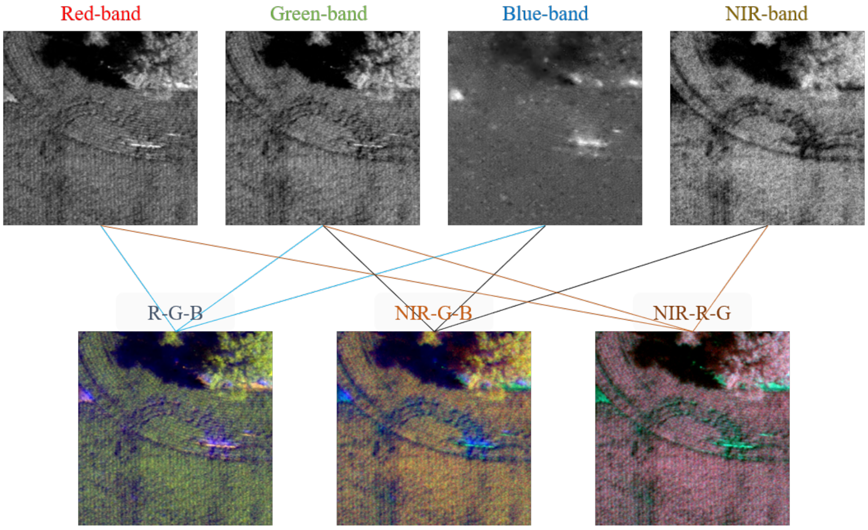


Figure 1.3: Examples of combining different spectral bands, i.e., Red-band, Green-band, Blue-band and NIR-band (near-infrared band), to produce different images, namely R-G-B (RGB), NIR-R-B (IRRB), and NIR-R-G (IRRG).

1.2 Key challenges

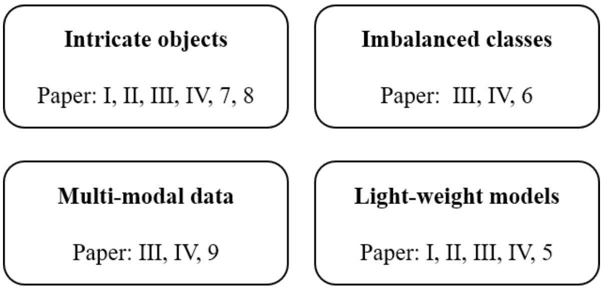
With recent advances in deep learning for image processing and pattern recognition, land cover classification of remote sensing data has progressed tremendously in the last few years. Nonetheless, there are still challenges related to the unique characteristics of remote sensing data and an inherent complexity in the pixel-wise classification tasks that strongly impact the classification performance. Some of these challenges include:

- **Intricate objects:** Remote sensing data consist of a variety of objects with intricate variations in aspect-ratio, size, and color-texture, such as roads, roofs, building shadows, low plants and tree branches, and so on. Furthermore, as shown in Figure 1.1, many high-spatial resolution remote sensing imagery are entirely composed of "stuff" classes, i.e. amorphous regions such as forest, vegetation, agricultural fields, water, and so on. Because such natural objects are not generally surrounded by well-defined borders in many remote sensing images with low spatial resolutions, pixel-wise annotation for learning models raises more difficulties.

- **Imbalanced classes:** The imbalanced nature of most remotely sensed data leads to a high asymmetric distribution of thematic classes, where some classes are frequent in the training dataset, while others have little appearance [22]. The acquisition of training data containing balanced class frequencies is often unfeasible in remote sensing. These highly imbalanced classes and samples cause one of the major issues for the application of deep learning for land cover mapping. The learning of deep neural networks is based on minimizing an objective or loss function. Because the minority class contribute less to the minimization of the objective function, a bias towards the majority class if often introduced. Hence, a model trained with an imbalanced class distribution will often have low accuracy for rare land cover classes. Consequently, as typical classification algorithms are designed to work with reasonably balanced datasets, learning the decision boundaries between imbalanced classes becomes a very challenging task [23].

- **Multi-modal data:** Multi-modal data is becoming more available in remote sensing [1]. Additional sensed data, such as light detection and ranging (LiDAR) data that can supplement common multi-spectral imagery with additional information about the same land, is also used for semantic mapping. For instance, in many applications, topographical information extracted from LiDAR data is used to improve discrimination of land cover classes with similar spectral characteristics [24]. Effective fusion of this different modality information is thus important for various application in remote sensing, but also very challenging due to large domain differences, high noises, and redundancies [25]. There are main three open questions, namely 'what', 'how' and 'where' to effectively fuse multi-modal features [26] for learning optimal joint representations of different modalities.

- **Light-weight models:** Remote sensing also faces the big data challenge [27]. Algorithms must be fast and scalable to deal with very large and ever-growing data volumes. However, many advanced deep learning models have millions of parameters and require a massive labeled dataset for training and high-performance GPUs. These increased model scale and computational burden severely limit the application and deployment of deep learning based methods in most scenarios with real-time requirements, such as on airborne or satellite-borne embedded systems. Hence, designing light-weight but highly effective deep learning models is highly value and demanding in the remote sensing domain.



Intricate objects

Paper: I, II, III, IV, 7, 8

Imbalanced classes

Paper: III, IV, 6

Multi-modal data

Paper: III, IV, 9

Light-weight models

Paper: I, II, III, IV, 5

Figure 1.4: Categorization of publications (see Section 1.5 and 1.6) according to the challenges they deal with.

1.3 Research objectives

In this thesis, we leverage various deep learning methods to provide solutions to some of the challenges mentioned in Section 1.2. Figure. 1.4 provides an overview of how the different publications relate to the challenges. The main focus of this research is to develop novel deep learning models to improve performance of land cover classification. Our main objectives are:

- Develop novel light-weight models that can effectively learn rich and local-global contextual representations for better interpretation of very high resolution remote sensing data.
- Propose new loss functions to address the issue of class imbalance commonly found in remote sensing data and also in many other domains.
- Design novel scalable multi-modal frameworks that can learn and fuse complementary information from multiple remote sensing modalities in order to deal with more complex scenarios.

1.4 Proposed approaches

The work presented in this thesis provides novel developments across a variety of deep learning approaches. In order to address the research challenges (Section 1.2), the work makes methodological contributions e.g. for convolution-based segmentation models, autoencoders/ variational autoencoders (AE/VAE) based latent representation learning and graph-based attention mechanisms associated with the land cover classification problem of remote sensing data. In Fig. 1.5, we have categorized the publications based on the types of deep learning approaches they mainly investigate and contribute to.

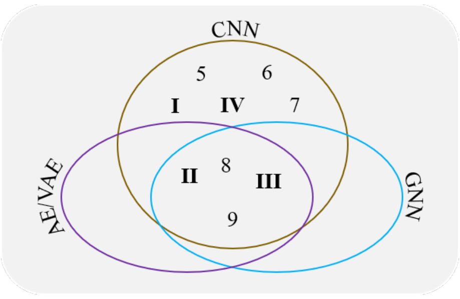


Figure 1.5: Methodological categorization of publications (see Section 1.5 and 1.6) according to the type of deep learning networks they mainly explore. CNN denotes convolutional neural network, GNN means graph neural network, and AE/VAE are autoencoders and variational autoencders.

In Paper I, we propose a novel architecture based on dilated convolutions, the so called dense dilated convolutions' merging (DDCM) network, that effectively utilizes rich combinations of dilated convolutions that enlarge the network's receptive fields with fewer parameters and features. Specifically, the DDCM network has three major differences compared to the state-of-the-art approaches in the remote sensing domain. First, we sequentially stack the output of each layer with its input features before feeding it to the next layer in order to alleviate context information loss. Second, the final output is computed on all features generated by intermediate layers, which can effectively aggregate the fused receptive field of each layer and maximally utilize multi-scale context information. Third, the method is highly flexible and extendable with the group and strided convolutions to address different domain problems.

We then explore how graph neural networks (GNNs) can be used in remote sensing to model long-range context dependencies. As a solution, we develop a novel Self-Constructing Graph module (SCG) that learns how to transform a 2D feature map into a latent graph structure and how to assign pixels to the graph's vertices from the available training data. In a nutshell, we model relations between pixels that are spatially close 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. The SCG-Net model can explicitly employ different types of GNNs to not only learn global context representations but also directly output the predictions (Paper II).

We extend the SCG to MSCG that considers multiple views for explicitly exploiting rotation invariance in remote sensing images to achieve improved segmentation performance. More specifically, we augment the input features to obtain multiple rotated views before fusing the multi-view global contextual information and projecting the features back onto the 2-D spatial domain. Furthermore, to address the issue of class imbalance commonly found in semantic segmentation datasets, we propose a novel adaptive class weighting loss function based on iterative batch-wise class re-balancing, rather than pre-computing the fixed weights across the entire dataset (Paper III).

In an effort to find a more effective method to extract and fuse information from multi-modal remote sensing data, we develop a novel and scalable framework in Paper IV, called MultiModNet, that is based on a novel pyramid attention fusion (PAF) modules and gated fusion units (GFU). The proposed PAF module is a lightweight network with a built-in cross-hierarchical-scale and cross-view attention fusion mechanism that can obtain rich and robust contextual representations. It can be used as a stand-alone decoder for a unimodal model to improve segmentation performance, or as a vital fusion mechanism to merge several modalities when combined with the GFU module.

1.5 Brief summary of included papers

This section briefly summarizes the papers included in this thesis. A list of other articles published over the course of the PhD project is presented in the next section. Figure. 1.5 provides an overview of the publications according to the types of deep learning models they mainly explore. The following papers are included in this thesis:

I. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Dense dilated convolutions' merging network for land cover classification.**" IEEE Transactions on Geoscience and Remote Sensing, vol 58.9, pp 6309-6320, doi:10.1109/TGRS.2020.2976658, 2020.

II. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Self-constructing graph neural networks to model long-range pixel dependencies for semantic segmentation of remote sensing images.**" International Journal of Remote Sensing, vol 42.16, pp 6184-6208, doi:10.1080/01431161.2021.1936267, 2021.

III. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Multi-View self-constructing graph convolutional networks with adaptive class weighting loss for semantic segmentation.**" IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 199-205, doi:10.1109/CVPRW50498.2020.00030, 2020.

IV. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Multi-modal land cover mapping of remote sensing images using pyramid attention and gated fusion networks.**" Submitted to International Journal of Remote Sensing, September, 2021.

Paper I: Develops a novel computationally light-weight and scalable network architecture, called the dense dilated convolutions merging network (DDCM-Net), for land cover classification of remote sensing images. The proposed DDCM module learns with densely linked dilated convolutions and outputs a fusion of all intermediate features without losing resolutions during the extraction of multi-scale features. This significantly reduces the computational redundancies and costs. It also allows for the efficient enlargement of the network's receptive fields by utilizing rich combinations of dilated and grouped convolutions with varying strided operations. The DDCM-Net and its variants demonstrated better performance on three different representative remote sensing datasets and are more computational efficient compared to other published methods.

Paper II: CNNs are commonly limited by their efficiency and ability to obtain long-range non-local contextual information due to their local valid receptive fields. For improved capturing of non-local representations, which has been shown to improve segmentation performance in remote sensing images, we propose the Self-Constructing Graph (SCG) module that learns a pixel-wise dependency graph directly from the image data and uses it to capture local-global contextual information efficiently to improve land cover mapping. 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 (GNNs) and CNNs. The SCG-Net model can achieve competitive performance with much fewer parameters and lower computational cost compared to related state-of-the-art models that rely on deep and wide multi-scale CNN architectures.

Paper III: Presents a new architecture called the Multi-view Self-Constructing Graph Convolutional Networks (MSCG-Net) that extends the SCG (proposed in Paper II) to explicitly exploit the rotation invariance in airborne images, by fusing multi-orientation information with deep-feature augmenting mechanisms. Moreover, we also develop an adaptive class weighting (ACW) loss that addresses the common class imbalance issue in remote sensing data. Unlike most existing methods that weighted loss functions with pre-computed class weights based on the pixel frequency of the entire training data, the ACW loss can compute the class weights automatically during iterative training and dynamically weigh the positive and negative regularization function. This provides an auto-dynamic-weighting solution that can reduce the class imbalance effect while also putting more emphasis on difficult samples (both positive and negative) during learning. Our experiments demonstrate that the MSCG-Net with the ACW loss achieves very robust and competitive performance and produces more accurate segmentation results for both larger and smaller classes on multi-spectral aerial images.

Paper IV: This paper focuses on DL based multi-modal fusion and classification problems of remote sensing data. Current multi-modal classification methods mostly use two independent encoders in parallel to extract features separately that tends to overlook the effects of noise and redundant features from very different multi-modal data. Therefore, we introduce a new gated fusion unit (GFU) that enables supplementary modalities. The GFU effectively extract the most valuable and complementary information via early gating feature merging, and thereby diminishing hidden redundancies and noise. By incorporating a novel pyramid attention fusion (PAF) module that can effectively extract a rich contextual representation from each modality by a deeply fused cross-view and cross-level pyramid attention mechanism, we develop a light-weight multi-modal segmentation network (MultiModNet). Extensive experiments on two publicly available remote sensing benchmark datasets demonstrate the effectiveness and superiority of the MultiModNet for multi-modal land cover classification.

1.6 Other papers

During the course of the PhD work, the following papers were also published:

5. Liu, Qinghui; Salberg, Arnt Børre; Jenssen, Robert. "**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. (ISBN 978-1-5386-7150-4). pp 6943-6946. 2018.

6. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Road Mapping in Lidar Images Using a Joint-Task Dense Dilated Convolutions Merging Network.**" In: IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium Proceedings. (ISBN 978-1-5386-9154-0). pp 5041-5044. 2019.

7. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Dense Dilated Convolutions Merging Network for Semantic Mapping of Remote Sensing Images.**" In: Joint Urban Remote Sensing Event, JURSE 2019. (ISBN 978-1-7281-0009-8). 2019.

8. Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre. "**Self-Constructing Graph Convolutional Networks for Semantic Labeling.**" In: IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium. Proceedings. (ISBN 9781728163741). 2020.

9. Chiu, Mang Tik; Xingqiang, Xu; others...; Liu, Qinghui; Kampffmeyer, Michael; Jenssen, Robert; Salberg, Arnt Børre; others... "**The 1st Agriculture-Vision Challenge: Methods and Results.**" In: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops 2020. (ISBN 978-1-7281-9360-1). pp 212-218. 2020.

1.7 Reading guide

The remainder of this thesis is organized into the following three parts: i) *methodology and context*, ii) *summary of research*, and iii) *included papers*.

The *methodology and context* part aims to provide the reader with the theoretical background that builds the foundation for the research presented in this thesis. To that end, this part is organized into five chapters. *Chapter 2* provides a short overview of learning methodologies and introduces deep feedforward neural networks. This is relevant background material for all papers. *Chapter 3* provides an introduction to convolutional neural networks. This is also relevant for all papers.

Chapter 4 provides an introduction to Autoencoders and Variational Autoencoders that are relevant for Paper II and III. *Chapter 5* provides an introduction to Graph Neural Networks, which is relevant for Papers II-IV. *Chapter 6* provides a short overview of segmentation networks and applications in remote sensing. This is relevant for all papers.

In the *summary of research and conclusion* part, we present a brief summary of the included papers and the author's main contributions to the works. Further, we provide concluding remarks and a discussion of future directions. The research papers are included in the *included papers* part.

Part I

Methodology and context

2

Basic learning methodologies

In this chapter, we briefly review the key concepts and notations of the learning methodologies that are used through this thesis. The work by Goodfellow et al. [28] and Zhang et al. [29] are the main references for this part. When no other references are explicitly cited, we kindly refer to these works for more details.

2.1 Machine learning basics

A machine learning (ML) algorithm is an algorithm that is able to learn from data to solve difficult tasks, such as classification, regression, dimension reduction, density estimation, and so on. The training data are often represented as a matrix[1], e.g. $\mathbf{X} = [\boldsymbol{x}_1^{\mathrm{T}}, \cdots, \boldsymbol{x}_m^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{m \times d}$, which contains m training samples $\boldsymbol{x}_i \in \mathbb{R}^d$ in each row $i = \{1, 2, \cdots, m\}$, and d different features x_{ij} for each sample where $j = \{1, 2, \cdots, d\}$. The dataset can also be represented as a set containing m samples: $\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(m)}\}$ that does not imply that any two feature vectors $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ have the same dimension like in the above matrix description $\mathbf{X}$.

Generally speaking, *supervised learning*, *unsupervised learning*, and *reinforcement learning* are the three main types of machine learning algorithms. While reinforcement learning methods do not only learn from a fixed dataset, but also interact with an environment and train themselves through trial and error, they are outside the scope of this work. Please see the work in [30, 31] for detailed information about reinforcement learning algorithms.

[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.

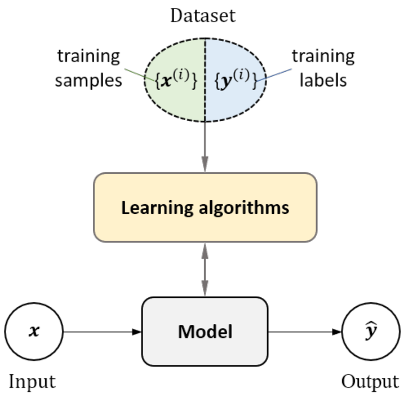


Figure 2.1: General learning process from data (a training dataset) that aim to produce a model that maps any input $\boldsymbol{x}$ to an output $\hat{\boldsymbol{y}}$. Each training sample $\boldsymbol{x}^{(i)}$ typically consists of a set of attributes called features, from which the model can be trained by certain learning algorithms to make its output $\hat{\boldsymbol{y}}$ (prediction). For supervised learning problems, there is a target value or vector that is designated as the label $\boldsymbol{y}^{(i)}$ (or target, ground-truth), while for unsupervised learning there are no corresponding (supervision) labels available in the dataset.

2.1.1 Supervised learning

In supervised learning, the models are learned using a dateset that contains both training samples and associated labels (i.e., target outputs/ground truths), so called feature-label pairs $\{\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}\}$. Figure 2.1 shows the supervised learning process. Regression and classification are the two most common supervised tasks. The purpose of both problems is to create a model that can predict the values of the dependent $\boldsymbol{y}$ from attribute variables $\boldsymbol{x}$. The primary difference between the two tasks is that the dependent attribute in regression is a real or continuous value, such as "salary" or "weight", whereas the dependent attribute in classification is categorical, such as {'cat', 'dog'}.

One of the most classical supervised ML approaches is the support vector machine (SVM) [32] used for classification problems. One important innovation associated with the SVM is the kernel trick that conceptually implements a non-linearly mapping from the input vector to a high-dimension feature space, such that a linear decision surface can be constructed in this feature space. However, kernel-based methods commonly suffer from a high computational cost of training, and the cost of evaluating the decision function is linear in the number of training samples. The k-nearest neighbors (KNN) algorithm [33] is another classical supervised learning algorithm that can be used to solve both classification and regression problems. We will not go into more detail about KNN or any of the other conventional supervised ML approaches such as the decision tree [34], random forests [35] and its many variants. This thesis primarily focuses on deep learning methods for supervised learning, which will be discussed in detail in the following sections.

2.1.2 Unsupervised learning

Unsupervised learning methods learn useful properties from the data that do not contain any supervision signal (label). Unsupervised learning is commonly used to perform tasks such as clustering, which divides the data into different groups of similar samples, as well as denoising or dimensionality reduction to compress the data.

A classic unsupervised learning algorithm is k-means clustering that divides the training dataset into k different clusters of samples. A cluster refers to a collection of data aggregated together based on certain similarities. The 'means' in the k-means refers to averaging of the data, that is, finding the centroid. Specifically, to perform the learning, the k-means algorithm starts with a first set of randomly initialized centroids $\{\boldsymbol{u}^{(1)}, \cdots, \boldsymbol{u}^{(k)}\}$ which are used as the starting points for every cluster, and then performs iterative calculations to optimize the values of the centroids until convergence.

Principal component analysis (PCA) is another popular unsupervised learning algorithm that learns a low-dimensional representation whose elements have no linear correlation with each other. PCA is commonly used as a simple and effective dimensionality reduction method that aims to preserve as much of the information in the data as possible, measured by the least-squares reconstruction error. There are also many other dimensionality reduction algorithms such as Linear Discriminant Analysis (LDA) [36], t-distributed Stochastic Neighbor Embedding (t-SNE) [37] and Autoencoders (that will be further discussed later in Section 4.1).

When we don't have human-annotated ground truths, we can still perform supervised learning by using some of the input as supervision targets, for example, by predicting some masked out part of the input using the remaining part of the input. This is referred to as *self-supervised learning*, which can be viewed as a special type of unsupervised learning. Self-supervised learning has proven to be quite effective in natural language processing [19, 38]. When working with unlabeled image data, contrastive learning has recently become one of the most powerful approaches in self-supervised learning and achieved great success in learning image representations [39]. Self-supervised learning is outside of the scope of this work but is a very import research direction in the future. For more details about self-supervised learning we refer the reader to [38, 39, 40].

2.2 Deep feedforward networks

In the last years, deep neural networks have set the state-of-the-art on many computer vision tasks. In the following, we address the basics of neural networks, activation functions, cost functions and optimization.

2.2.1 Model architecture

Deep feedforward networks, also called feedforward neural networks or Multi-layer Perceptrons (MLPs), represent the general foundation of deep learning architectures. A feedforward network aims to learn a mapping function $\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \Theta)$ that maps the input data to $\boldsymbol{y}$ by adjusting the parameters Θ to result in the best prediction . As shown in Fig. 2.2, the mapping function $f(\cdot)$, is commonly composed of a number of intermediate functions $f^{(1)}, \ldots, f^{(n)}$ that are parametrized by $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(n)}$ respectively. It can be represented as a chain function as

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \Theta) = f^{(n)}\left(f^{(n-1)}\left(\ldots\left(f^{(1)}(\boldsymbol{x})\right)\ldots\right)\right) \ . \tag{2.1}$$

Here we omit the arguments $\{\boldsymbol{\theta}^{(l)}\}_{l=1}^{n}$ to shorten notation and $\Theta = \{\boldsymbol{\theta}^{(1)}, ..., \boldsymbol{\theta}^{(n)}\}$ is the parameter set of the network. The model is thus called feedforward because there is no feedback connections between the output $\hat{\boldsymbol{y}}$ and the input $\boldsymbol{x}$. Note that dimensions of $\boldsymbol{x}$ and $\hat{\boldsymbol{y}}$ do not need to be equal, and $\hat{\boldsymbol{y}}$ can also be a scalar $\hat{y}$. The overall length of the chain function defines the **depth** (the number of layers) of the feedforward model. In this case, we have a n-depth or n-layer model. The last layer of the network $f^{(n)}$ is called the output layer, the other $n-1$ layers from $f^{(1)}$ to $f^{(n-1)}$ are called hidden layers that produce hidden features $\boldsymbol{h}^{(l)}$, and the input layer connects to the input variables, as shown in Figure 2.2.

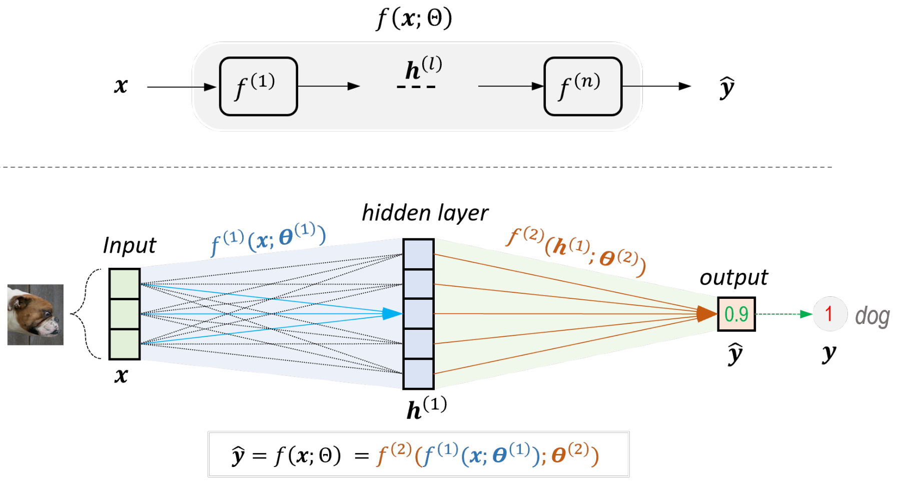

$f(\boldsymbol{x}; \Theta)$
$\boldsymbol{x}$
$f^{(1)}$
$\boldsymbol{h}^{(l)}$
$f^{(n)}$
$\hat{\boldsymbol{y}}$
hidden layer
Input
$f^{(1)}(\boldsymbol{x}; \boldsymbol{\theta}^{(1)})$
$f^{(2)}(\boldsymbol{h}^{(1)}; \boldsymbol{\theta}^{(2)})$
output
0.9
1
dog
$\boldsymbol{x}$
$\boldsymbol{h}^{(1)}$
$\hat{\boldsymbol{y}}$
$\boldsymbol{y}$
$\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \Theta) = f^{(2)}(f^{(1)}(\boldsymbol{x}; \boldsymbol{\theta}^{(1)}); \boldsymbol{\theta}^{(2)})$

Figure 2.2: Top: A general structure of deep feedforward networks that consist of a broad class of feedforward mapping functions, i.e., layers $\{f^{(l)}(\cdot; \boldsymbol{\theta}^{(l)})\}_{l=1}^{n}$, that map from an input $\boldsymbol{x}$ to the output $\hat{\boldsymbol{y}}$. Bottom: A simple example of a two-layer feedforward network, i.e., a binary classifier that outcomes as either a 1 'dog' or 0 'not'. This simple model consists of an input layer, i.e. a 3-dimensional vector $\boldsymbol{x}$ representing an image of 'dog', a hidden layer containing five units $\boldsymbol{h}^{(1)}$ as the hidden representation, and an output layer containing a single unit as the final prediction. In this example, the model outputs $\hat{\boldsymbol{y}} = 0.9$ as the probability of class 'dog', while the ground truth $\boldsymbol{y} = 1$.

A given layer $\boldsymbol{h}^{(l)} \in \mathbb{R}^{m_l}$, consists of many parallel units or neurons $\{h_1^{(l)}, h_2^{(l)}, \ldots, h_{m_l}^{(l)}\}$. Each neuron $h_i^{(l)}$ represents a single vector-to-scalar function $f_i^{(l)}(\cdot)$ that takes units from its previous layer, i.e. $\boldsymbol{h}^{(l-1)} \in \mathbb{R}^{m_{l-1}}$, as input to computer its own activation value. Thus, each unit in the l-th layer of the deep feedforward network can be defined as follows:

$$h_i^{(l)} = f_i^{(l)}\left(\boldsymbol{h}^{(l-1)}; \boldsymbol{w}_i^{(l)}, b_i^{(l)}\right) = \delta^{(l)}\left(\boldsymbol{w}_i^{(l)\mathrm{T}}\boldsymbol{h}^{(l-1)} + b_i^{(l)}\right) , \tag{2.2}$$

where $\boldsymbol{w}_i^{(l)} \in \mathbb{R}^{m_{l-1}}$ is the weight vector, $b_i^{(l)}$ is the bias parameter, $\delta^{(l)}(\cdot)$ denotes the non-linear activation function at the l-th layer, $i = 1, 2, \ldots, m_l$, and $\boldsymbol{h}^{(0)} = \boldsymbol{x}$. We can thus summarize all learnable parameters of the n-layer deep network as:

$$\Theta = \left\{\boldsymbol{\theta}^{(l)} = \left(\boldsymbol{W}^{(l)} \in \mathbb{R}^{m_{l-1}\times m_l}, \boldsymbol{b}^{(l)} \in \mathbb{R}^{m_l}\right) : l = 1, 2, \ldots, n\right\} , \tag{2.3}$$

where m_l and m_{l-1} denote the of the number of units at the l-th layer and the l-1-th layer respectively.

2.2.2 Non-linearity

The non-linear activation function $\delta(\cdot)$ is a key component of neural networks since it enables the network to learn complex non-linear mappings between the network's inputs and its outputs, which are essential for modeling complex high dimensionality data, such as images, video, audio and so on. Without a non-linear activation function, a deep neural network would behave just like a linear model regardless of how complex its architecture is, because summing all its layers would just result in a simple linear transformation from input to output.

Modern neural network models may use linear activation functions in the output layer, while in other layers they often apply non-linear activation functions such as the rectified linear unit (ReLU [41]) that is defined as

$$\delta(z_i) = \mathrm{ReLU}(z_i) = \max(0, z_i) , \tag{2.4}$$

where $z_i = \boldsymbol{w}_i^{(l)\mathrm{T}}\boldsymbol{h}^{(l-1)} + b_i^{(l)}$, denotes one unit of the l-th layer. Note that we omit the layer superscript $^{(l)}$ of z_i and δ to simplify the notation.

The ReLU activation function is the default option in many deep networks since it is computationally efficient and yet maintains better gradient flow compared to sigmoid (eq. 2.7) and tanh, i.e., $\frac{e^{z_i}-e^{-z_i}}{e^{z_i}+e^{-z_i}}$, which are prone to the vanishing gradient problem [42]. However, ReLU tends to result in dead neurons. For example, if the units are not activated initially, then they are always in the off-state as zero gradients flow through them. This can be addressed by enforcing a small negative gradient flow through the network, such as the Leaky ReLU [43] activation function. Another popular activation function is the PReLU [44] given as

$$\delta(z_i; \alpha) = \mathrm{PReLU}(z_i; \alpha) = \begin{cases} \alpha z_i & \text{if } z_i < 0 \\ z_i & \text{otherwise} \end{cases} , \tag{2.5}$$

where the parameter α is a learnable parameter.

There are many other types of activation functions more commonly found in the output units such as softmax that is defined as

$$\delta(z)_i = \mathrm{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=0}^{m_l} e^{z_j}} \ , \tag{2.6}$$

which ensures that the output values are in the range $(0, 1)$ and always sum to 1. When we are performing multi-class classification, we commonly use softmax in the output layer of our model. For binary or multi-label classification tasks, *sigmoid* activation is a default choice that is defined as

$$\delta(z_i) = \mathrm{sigmoid}(z_i) = \frac{1}{1 + e^{-z_i}} \ . \tag{2.7}$$

Thus, when using the sigmoid as the output activation function and ReLU in the hidden layer, our 2-layer toy model for dog classification (shown in Figure 2.2) can be written as

$$\begin{aligned} \hat{y} &= f\,(\boldsymbol{x}; \Theta) \\ &= \mathrm{sigmoid}\left(\boldsymbol{W}^{(2)\mathrm{T}}\ \mathrm{ReLU}\left(\boldsymbol{W}^{(1)\mathrm{T}}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right) + b^{(2)}\right) \\ &= \left(1 + e^{-\left(\boldsymbol{W}^{(2)\mathrm{T}}\ \max\left(0, \boldsymbol{W}^{(1)\mathrm{T}}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right) + b^{(2)}\right)}\right)^{-1} \ , \end{aligned} \tag{2.8}$$

where $\Theta = \{\boldsymbol{W}^{(1)} \in \mathbb{R}^{3\times5}, \boldsymbol{b}^{(1)} \in \mathbb{R}^5, \boldsymbol{W}^{(2)} \in \mathbb{R}^{5\times1}, b^{(2)} \in \mathbb{R}^1\}$, and $\boldsymbol{x} \in \mathbb{R}^3$. The parameter Θ set of this 2-layer model contains a total of 26 learnable weights.

2.2.3 Cost functions

The choice of the cost function is an important aspect for designing a deep neural network. The cost function makes it possible to train a deep learning model using gradient-based optimizers via backpropagation to update the parameters through minimizing the cost function.

Figure 2.3 illustrates the training process of a deep model. Generally, one single training iteration consists of two propagation processes: forward propagation (forwardprop) and back-propagation (backprop)[2]. Forward and backward propagation depend on each other. During training the forwardprop traverses the model onward and computes all the variables on its path. These are used by backprop where the compute order on the path is reversed with a gradient-based algorithm, such as stochastic gradient descent (SGD), which adjust the model's parameters Θ in the direction of its gradient ∇_Θ.

The cost function can be written as an average over the training set (the data-generating distribution p_{data}), such as

$$J(\Theta) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_{\mathrm{data}}} L(f(\boldsymbol{x};\Theta), y) \ , \tag{2.9}$$

[2] Note that backpropagation needs to reuse the stored intermediate values from forward propagation to avoid duplicate calculations. The computer thus needs to retain the intermediate values until backpropagation is finished. That is one of the reasons why training requires significantly more memory and easily results in out of memory issues in particular when training deeper models with larger batch size.

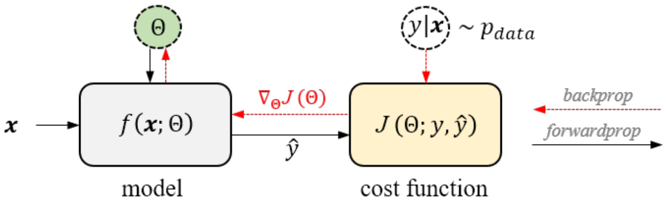


Figure 2.3: An overview of model training process. When training neural networks, we alternate forward propagation with backpropagation and updating model parameters using gradients $\nabla_\Theta J(\Theta)$ given by a cost function $J(\Theta)$.

where L is the per-sample loss function, $f(\boldsymbol{x};\Theta)$ is our model that produces the output $\hat{y}$ when the input is $\boldsymbol{x}$, and p_{data} is the training sample generating distribution.

In practice, we can compute these expectations by randomly sampling a mini-batch of samples $\{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(m)}\}$ with corresponding ground truth $y^{(i)}$ from the dataset p_{data}, then taking the average over only these mini-batch samples. When both $\boldsymbol{x}^{(i)}$ and $y^{(i)}$ are discrete, the mini-batch cost function can be written as

$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} L\left(f\left(\boldsymbol{x}^{(i)};\Theta\right), y^{(i)}\right) . \tag{2.10}$$

Hence, the mini-batch[3] gradient ∇_Θ of the loss with respect to the parameter set Θ can be expressed as

$$\nabla_\Theta J(\Theta) = \frac{1}{m}\nabla_\Theta\left[\sum_{i=1}^{m} L\left(f\left(\boldsymbol{x}^{(i)};\Theta\right), y^{(i)}\right)\right] . \tag{2.11}$$

A common used loss function L in classification settings is the binary cross-entropy loss, also called Bernoulli cross-entropy, defined as

$$L = L_{bce}(y, \hat{y}) = -\left(y\log(\hat{y}) + (1-y)\log(1-\hat{y})\right) , \tag{2.12}$$

where y is the label (e.g., 1 for “dog” and 0 for “not-dog” in our dog classification model), and $\hat{y}$ is the predicted class (e.g., "dog") probability for the input sample.

For multi-class cases, the cross-entropy loss, also called categorical cross-entropy, is given as

$$L = L_{cce}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{c}\sum_{j=1}^{c} y_j \log(\hat{y}_j) , \tag{2.13}$$

[3] The mini-batch approach is the default method to implement the gradient descent algorithm in deep learning. Because the mini-bath gradient descent often provides more stable and faster convergence towards the global minimum since an average gradient over m samples results in less noise. However, a new hyperparameter m, known as the mini-batch size, is introduced, which often has a significant impact on the neural network’s overall performance.

where $c > 1$ denotes the number of classes (i.e., the number of scalar values in the model output $\hat{\boldsymbol{y}}$ or the target vector $\boldsymbol{y}$: one-hot-encoding), $\hat{y}_j$ is j-th scalar value in $\hat{\boldsymbol{y}}$, y_j is the corresponding target value, and each input sample $\boldsymbol{x}$ only belongs to just one class. Thus, the mini-batch cost function for a multi-class classification model can be given as

$$J(\Theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{c} y_j^{(i)} \log(\hat{y}_j^{(i)}) \ . \tag{2.14}$$

A cost-function for multi-label cases, i.e. multi-label bernoulli cross-entropy loss, is given as

$$L = L_{mbce}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{c} \sum_{j=1}^{c} y_j \log(\hat{y}_j) + (1 - y_j) \log(1 - \hat{y}_j) \ . \tag{2.15}$$

Here we assume there are multiple classes (i.e., c labels), and the model tries to decide for each class whether the input belongs to or contains that class or not. In other words, each input data $\boldsymbol{x}$ can belong to multiple labels. This is called multi-label classification that differs from multi-class classification tasks.

2.2.4 Optimization

The training of a deep model is usually performed by minimizing the cost function using a form of gradient descent through backpropation. A commonly utilized gradient-based optimization algorithm for deep learning is the mini-batch stochastic gradient descent (SGD) which is given as

$$\Theta = \Theta - \eta \nabla_{\Theta} \ , \tag{2.16}$$

where η is the learning rate, i.e., the step size of the update per mini-batch training iteration. One may combine SGD with algorithms such as the momentum algorithm [45] defined as

$$\begin{aligned} \Lambda &= \epsilon \Lambda - \eta \nabla_{\Theta} \ , && \text{(step-1)} \\ \Theta &= \Theta + \Lambda \ , && \text{(step-2)} \end{aligned}$$

where Λ is a velocity variable to accumulate the model's gradient with a momentum factor ϵ. In other words, the momentum algorithm incorporates previous gradients estimates for the current parameter update. The step size of the parameter update can be larger with a velocity variable when gradients point in the same direction. This allows for faster convergence than the SGD without applying the momentum.

Many other kinds of optimization algorithms with adaptive learning rates, such as RMSProp [28], AdaGrad [46], Adadelta [47] and Adam [48] are widely used. Each algorithm aims to address the challenge of optimizing deep models by adapting the learning rate for each model parameter, however, there is no consensus on which algorithm should be choose to use for a specific model. The choice generally depends on the user's familiarity with the optimizer and the model for ease of hyperparmeter tuning. A more detailed discussion of various optimization methods is provided in [28].

3

Convolutional neural networks

This chapter presents a brief introduction to Convolutional Neural Networks (CNNs or ConvNets for short) [49], which is relevant background material for all papers.

ConvNets are a powerful type of neural network for recognizing patterns in data that has a grid-like structure, such as time-series data (1D grid), image data that consists of 2D grid of pixels, and videos (3D-grid). On the one hand, ConvNets are also made up of neurons with learnable weights and biases, just as standard deep feedforward networks (explained in the previous chapter 2.2). Each neuron takes some inputs, does a dot product, and then executes the activation function. The entire CNN network still defines a single differentiable function from input to output, which still has a cost function for training. All of the methods developed for training deep feedforward networks are still applicable to ConvNets.

ConvNet architectures, on the other hand, make the explicit assumption that the input neurons in the network are locally connected, as opposed to fully connected neural networks (described in chapter 2.2) in which all units are connected. The convolutional operator simply encodes local connectivity, and its weight parameters can be shared across the entire grid-like data to detect hidden features, implying that the convolution operation is independent of input size. Figure 3.1 illustrates an example of a convolution operation on an input image with single kernel. These then make the forward function more efficient to implement and considerably reduce the amount of parameters in the network. Furthermore, the ConvNet provides translation equivariance that offers a mechanism to learn a model that takes into account the spatial property of the input data. By combining convolutions with spatial pooling operators (see Section 3.4), an approximative translation invariance can also be achieved in neural networks.

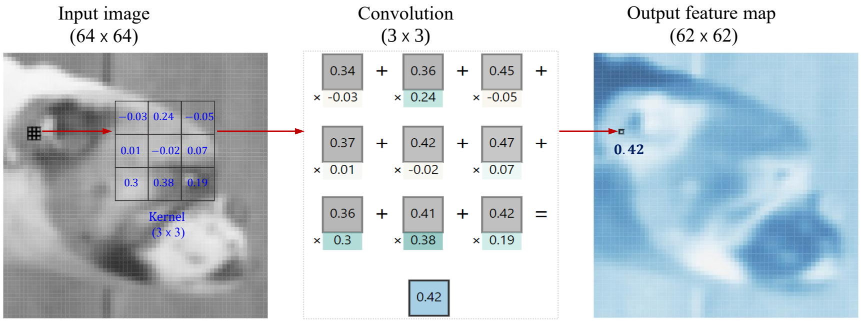


Figure 3.1: An example of a convolutional operation.

3.1 Standard convolution

Mathematically, a convolution is an integration of the product of two functions (x and w) assuming one function $x(\cdot)$ is reversed and shifted by a value of t over another function $w(\cdot)$, that is typically denoted with an asterisk as $(x * w)(t)$:

$$x(t) * w(t) = \underbrace{\int_{-\infty}^{\infty} x(\tau)w(t-\tau)d\tau}_{(x*w)(t)} \ . \tag{3.1}$$

In convolutional network terminology, the function $x(\cdot)$ is referred to as the input, such a 2D image, and the second function $w(\cdot)$ as the kernel (vector/matrix) or filter[1]. The output is referred to as the feature map that is detected by the parameters (kernel weights plus an optional bias term) of the filter.

In practice, we can implement the infinite integration as a summation over a finite number of array elements. Assume our input consists of 2D image data $\mathbf{X}$ with elements $X_{i,j}$, and the output $\mathbf{Z}$ with the same format as $\mathbf{X}$, given a 2D kernel $\mathbf{K}$ with elements $K_{m,n}$, we thus can define the discrete convolution (convolving $\mathbf{K}$ across $\mathbf{X}$) as:

$$\begin{aligned} Z_{i,j} &= (\mathbf{X} * \mathbf{K})(i,j) \\ &= \sum_{m}\sum_{n} X_{i-m,j-n}K_{m,n} \\ &:= \sum_{m}\sum_{n} X_{i+m,j+n}K_{m,n} \qquad \text{; without flipping the kernel, also called cross-correlation.} \end{aligned} \tag{3.2}$$

Note that the mathematical definition of a convolution is not equivalent to the dot product between image region and filter kernel, but to the dot product between the image region and the flipped kernel. The convolution operation, as used in most deep learning libraries, is referred to as cross-correlation (i.e., sliding dot product). However, since the filter kernels contain the weights that are eventually

[1] A filter is actually a set of kernels with an optional bias term, although we sometimes use filter and kernel interchangeably in the context of convolutional networks. The number of filters always equals to the number of feature maps in next layer, while the number of kernels in each filter commonly equals to the number of feature maps in this layer.

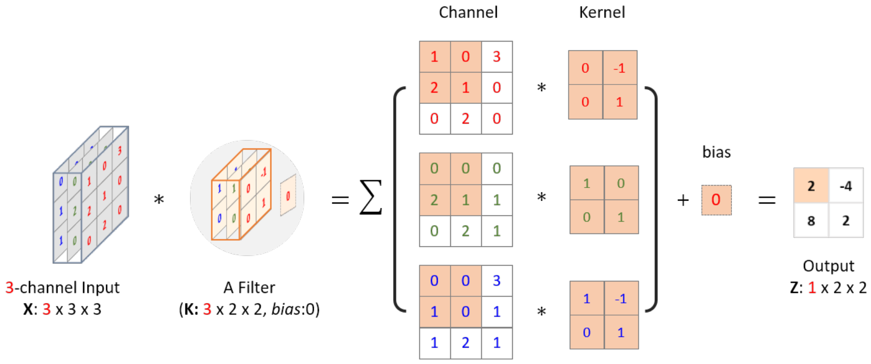


Figure 3.2: An example of multi-channel 2D input. The convolution computation (cross-correlation) uses one filter including 3 kernels (size of 2×2) and 1 bias term, with 3 input channels (spatial resolution of 3×3, i.e. height × width) and 1 output channel (2×2). The input and output have different height and width due to no padding.

learned throughout the training, the convolutional network can learn either filter weights, which correspond to a flipped or a regular kernel. The depth of a filter kernel is often needed to be equal to the depth (i.e., channel) of the input data (for example, the channel of an RGB image is three), As illustrated in Figure 3.2, a filter kernel produces a two-dimensional output with a depth of one, which we refer to as a feature map (also called activation map).

3.2 Variants of the convolution

There are currently many convolutional variants that aim to improve the convolutional operation with fewer parameters and better computational efficiencies, such as grouped convolutions [41, 50, 51, 52], dilated or altrous convolutions [53, 54, 55], and depthwise separale convolutions [56, 57, 58], among others [59, 60, 61].

3.2.1 Grouped convolution

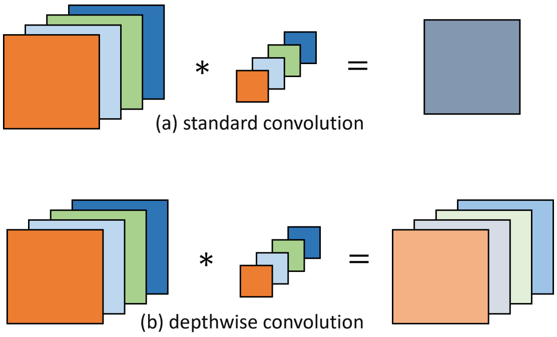
The use of grouped convolutions dates back to AlexNet [41] with the motivation for distributing the model over two GPUs. It was used later by architectures such as ResNeXt [50] and many others [50, 51]. The input channels are commonly divided into C groups in a grouped convolution, and regular convolutions are performed separately within each group. Each output channel is thus just connected to a subset of the input channels in this manner. Compared to a regular convolution using a full connectivity pattern, grouped convolutions can reduce the parameter size and computational cost.

3.2.2 Dilated or altrous convolution

Dilated convolutions [53], also know as altrous convoluions [54], have been demonstrated to improve performance in many classification and segmentation tasks [62, 52, 63, 64]. One of the main advantages of using dilated convolutions is that it allows us to adjust the filter's receptive field flexibly to obtain multi-scale information without resorting to down- and up-scaling operations. A 2D dilated convolution operator can be defined as

$$\begin{aligned} Z_{i,j} &= (\mathbf{X} *_r \mathbf{K})(i, j) \\ &:= \sum_m \sum_n X_{i+rm, j+rn} K_{m,n} \quad , \end{aligned} \tag{3.3}$$

where $*_r$ denotes a dilated convolution operator and dilation rate $r \in \mathbb{Z}^+$ is a positive integer. In a dilated convolution, a kernel size, e.g. $(M \times N)$, is effectively enlarged to $(M + (M-1)(r-1)) \times (N + (N-1)(r-1))$ with the dilation factor r. As a special case, a dilated convolution with dilation rate $r = 1$ corresponds to a standard convolution.



(a) standard convolution

(b) depthwise convolution

Figure 3.3: An illustration of a standard convolution and a depthwise convolution.

3.2.3 Depthwise separable convolution

Depthwise separable convolutions were at first introduced in Xception [56], and were wildly used in MobileNet architectures [57, 65, 66]. Depthwise separable convolution is built on a depthwise convolution followed by a pointwise convolution (i.e., a 1×1 convolution). Unlike a standard convolution that conducts convolution on all channels at once, a depthwise convolution performs convolution on each channel separately. In other words, each output channel has an independent convolution kernel corresponding to each input channel as shown in Fig. 3.3. Roughly speaking, it's computational cost is the total of the costs of the depthwise and pointwise operations. It gives a computation reduction of around 8 times less operations when compared to a regular convolution with a kernel size of 3×3.

Depthwise and grouped convolutions have a lot in common. Depthwise convolutions use a set of independent kernels for each input channel, whereas grouped convolutions use a set of independent kernels for each channel group. These feature maps obtained by depthwise convolutions are stacked together and then the pointwise convolution is applied to output the desired number of channels.

3.3 Convolutional layers

The convolutional layer is the core building block of a ConvNet. Deep CNN models prove very effective with stacking many convolutional layers, which allow layers close to the input to learn low-level features (e.g., lines, edges) and layers deeper in the model to learn high-order or more abstract features, like shapes or specific objects.

A convolutional layer commonly consists of many convolutional filters that extract several different activation maps. The number of filters determines the depth dimension (channel) of the convolution layer's output that stacks all feature maps detected by the individual filters. Other properties, such as stride, padding and kernel size can be specified for each convolution layer in deep learning libraries. These hyperparameters affect the size of the output feature map. Figure 3.4 shows an example of convolutional layer with strides and padding operations.

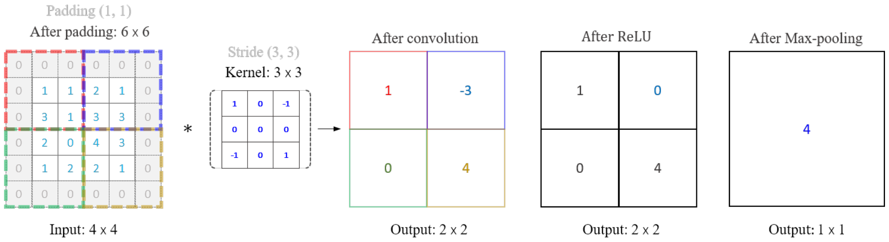


Figure 3.4: An example of convolutional layer (kernel: 3×3) with strides $(3, 3)$ and zero-padding $(1, 1)$, ReLU and following by a max-pooling to keep only the maximum value for 2×2 region in the feature map.

Specifically, the *Stride* means the number of rows and columns traversed per sliding kernel window. In our previous examples, we default to sliding one pixel at a time from the upper-left to the lower-right. However, sometimes we want to move our window more than one pixel at a time by skipping the intermediate pixels, such that we can perform downsampling. For example, stride 2 is often used to downsample the spatial dimension of the input by half.

The *Padding* method is used to conserve information at the borders of our input image or feature maps, which may lead to better performance. There exist many padding approaches, such as zero-padding (add zeros symmetrically around the edges of an input), reflection-padding (reflect the input values across the border axis) and replication-padding (extend by replicating the values along borders) [67], and so on. There is no consensus on which padding scheme is the best yet, but the most commonly used method is zero-padding because of its simplicity, computational efficiency and performance. This approach is adopted by many high-performing CNNs models such as the ResNet [14].

The *Kernels* used in a convolution layer commonly have odd number size, such as 1×1, 3×3 , 5×5 , or 7×7 as the height and width values. In practice, we choose odd number kernel sizes and padding to precisely preserve spatial dimensionality while padding with the same number of rows on top and bottom, and the same number of columns on left and right. In other words, for any 2D input $\mathbf{X}$, when the kernel's size is an odd number and the number of padding rows and columns on all sides are the same, producing an output with the same height and width as the input. Choosing an appropriate kernel size will be dependent on your task and data, but smaller kernel sizes (e.g. 3×3) in general lead to better performance for the image classification task because we are allowed to stack deeper convention layers together to learn more rich and robust features [14].

3.4 Pooling and fully-connected layers

A pooling operator, which is a parameter-free down sampling operation, is often added after the convolution layers, specifically, after a non-linearity (e.g. ReLU, see Section 2.2.2) has been applied to the convolution feature maps. Pooling computes a summary value for each small local patch on the feature map, thereby making the feature representations become robust (approximately invariant) to small translations of the input. This is referred to by the technical phrase "local translation invariance" that means that if we translate the input by a small amount, the values of most of the pooled outputs do not change.

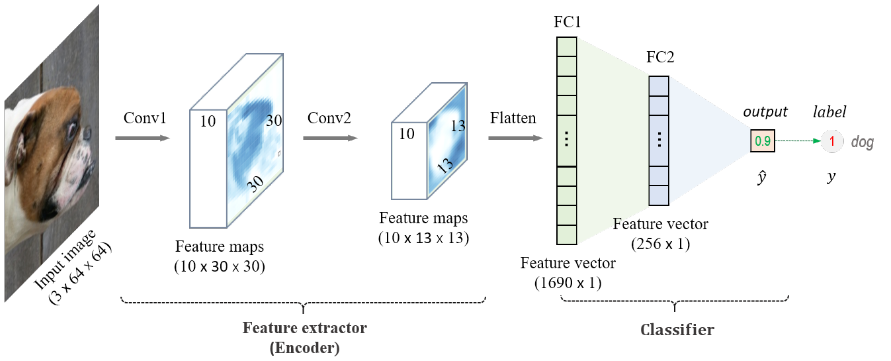


Figure 3.5: A simple example of 2D CNN network for classification of RGB images with fully connected layers. Conv1 and Conv2 layers consist of Conv3×3+ReLU+Conv3×3+ReLU+Max-pooling. FC1 and FC2 denote the fully connected (FC) layer 1 and layer 2. Here y is the label, i.e., 1 for class "dog" and 0 for "not-dog", and $\hat{y}$ is the predicted probability of class "dog" for the input image.

Like convolutional layers, pooling operators require a fixed-shape kernel (known as the pooling window) that is sliding over all regions in the input according to a selected stride, computing a single value for each location traversed by the pooling window. The pooling layers are deterministic and parameter-free, typically calculating either the maximum (as illustrated in Figure 3.4) or the average value of the features in the pooling window. These operations are therefore called max-pooling and average-pooling respectively, and are two commonly used pooling methods in many modern deep ConvNet models.

The fully connected layers in a convolutional network, that are commonly used as the final layers in image classification models, are essentially feedforward neural networks (generally a two or three layer MLPs). They are typically used to map the activation features produced by the concatenation of convolutional, nonlinearity, rectification, and pooling layers into a class probability vector. With the exception of the input layer, individual fully connected layers operate identically to the layers of the MLP as discussed in Section 2.2. Figure 3.5 shows a simple example of 2D CNN network with fully connected layers for classification of RGB images.

4 Autoencoders

In this chapter, we will briefly introduce autoencoders and focus on two types of autoencoders: traditional autoencoders and variational autoencoders, to provide mathematical background on how these neural architectures work. The material covered in this chapter serves as the foundation for Papers II and III.

4.1 Traditional autoencoders

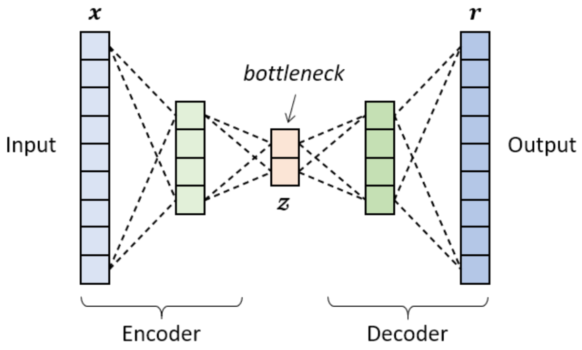


Figure 4.1: The concept structure of an traditional/undercomplete autoencoder, mapping an input $\boldsymbol{x}$ to an output $\boldsymbol{r}$ through an internal/latent representation or code $\boldsymbol{z}$. In this simple example, it consists of a two layer encoder that maps the input (nine dimensional vector) into a two dimensional code. A two layer decoder is then used to map the code back to the nine dimensional output.

An autoencoder (AE) is a special type of neural network that uses input data to learn a latent-space representation (also known as a bottleneck) and then reconstructs the output from this representation. This latent representation represents the data's most essential features/characteristics. The data can take the form of speech, text, picture, or video, among other things.

Internally, a standard AE consists of two main parts, the encoder that maps the input data $\boldsymbol{x}$ to its latent representation $\boldsymbol{z}$ (described as a code that is present at the bottleneck), and the decoder that maps the code back to the output $\boldsymbol{r}$ (Figure 4.1). The traditional autoencoder whose code dimension is much less than the input dimension is called undercomplete [28]. Learning an undercomplete representation forces the autoencoder to extract the most salient features from the training samples. Hence, the bottleneck is the key attribute of a standard AE model. It constrains the model to learn compression of the input data rather than memorize a training sample to accurately build its reconstruction.

The AE model can be trained by the standard backpropagation procedure as show in Figure 4.2. In practice, both the encoder and decoder functions can be represented as one or more feedforward neural layers or convolutional layers. The encoder function f maps the input data $\boldsymbol{x}$ with learnable parameters $\boldsymbol{\phi}$ to its code $\boldsymbol{z}$, i.e., $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\phi})$. The decoder function, denoted by g with learnable parameters $\boldsymbol{\theta}$, maps the latent space to the output $\boldsymbol{r} = g(\boldsymbol{z}; \boldsymbol{\theta})$. The loss function $\mathcal{L}$, such as the mean squared error (MSE), measures the difference between the input $\boldsymbol{x}$ and its reconstruction $\boldsymbol{r}$. Therefore, autoencoders learn unsupervised (or self-supervised) and can be trained by gradient methods (such as gradient decent or stochastic gradient descent) through backpropagation to update the weights (the learnable parameters) to minimize the reconstruction loss, e.g., $\mathcal{L}(\boldsymbol{x}, \boldsymbol{r}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - r_i)^2$, where $\boldsymbol{x}, \boldsymbol{r} \in \mathbb{R}^n$.

There are many various techniques in order to improve their performance to capture important features and learn richer representations, such as the Denoising Auto-Encoder (DAE) [68] which changes the learning objective of the AE from reconstruction to denoising of the input, the Sparse Auto-Encoder (SAE) [69] that adds sparsity regularization to avoid that the model can learn the identity mapping, the Contractive Auto-Encoder (CAE) [70] that tries to learn more robust representations, and so on. For a review and more details about these AE variations and extensions, we refer the reader to [28].

4.2 Variational autoencoders

Variational autoencoders (VAE) [71] belong to the families of variational Bayesian methods with a multivariate distribution as prior, and a posterior approximated by an artificial neural network, forming the so-called variational encoder-decoder structure [72]. VAEs have emerged as one of the most popular methods to unsupervised learning in generating many kinds of complicated distributions, including handwritten digits [71], CIFAR images [73], faces [74, 75], and segmentation [76].

From a systemic point of view, both the classical autoencoder and the variational autoencoders take a collection of high dimensional data as input. Then they encode the input into a latent space, which they then decode in order to reconstruct it as precisely as possible. Figure 4.2 illustrates the block schemes of both AE and VAE

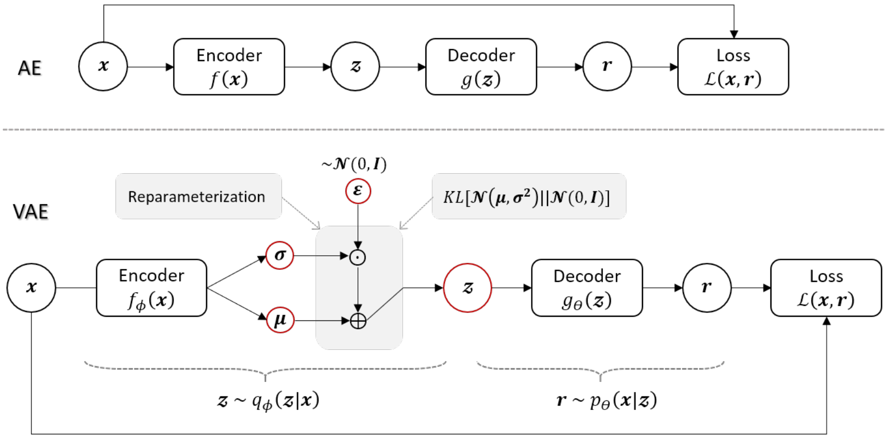


Figure 4.2: The systemic pipelines of an original autoencoder (AE) and a variational autoencoder (VAE). Top: AE where $f(x)$ is the encoding function, $g(z)$ is the decoding function, and $\mathcal{L}(x,r)$ is a loss function. Bottom : VAE where the encoder learns a vector of means μ and standard deviations σ as the parameters of the variational distribution, then the latent code z is sampled from the latent space through re-parameterization (the mixture of learned multivariate Gaussian and normal distributions), and the Kullback–Leibler (KL-loss) divergence between the latent space and normal Gaussians is introduced accordingly to the VAE model. Both AEs and VAEs are appealing because they can be constructed on top of a variety of neural networks and trained using stochastic gradient descent via backpropagation.

models. Despite their architectural similarities, the mathematical formulas differ substantially. The main difference is that in VAEs, the latent space is composed of a mixture of distributions rather than a fixed vector as in AEs.

Specifically from a statistical point of view, the goal of the VAE is to learn a distribution $p_\theta\ (x)$ over a multi-dimensional variable x characterized by an unknown probability density function. One of the main reasons for modelling distributions is that we can draw samples from the learned distribution to generate new plausible values of x. The probability of $p_\theta(x)$ can be described as a marginalization of a joint probability density function, i.e. $p_\theta(x,z)$ of the data x and an latent variable z, so that

$$p_\theta(x) = \int_z p_\theta(x,z)dz\ , \tag{4.1}$$

where θ denotes the set of the learnable network parameters. We typically use the product of the likelihood $p_\theta(x|z)$ and the prior $p(z)$ to describe the joint probability $p_\theta(x,z)$, i.e.,

$$p_\theta(x) = \int_z p_\theta(x|z)p(z)dz\ . \tag{4.2}$$

The likelihood $p_\theta(x|z)$ describes how to compute the distribution over the observed data x given latent variable z. However, we may wish to obtain the latent variable z (feature representations) given input x, which is described in the posterior distribution $p_\theta(z|x)$. Based on Bayes' theorem, the posterior can be defined as:

$$p_\theta(z|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|z)p(z)}{p_\theta(\boldsymbol{x})} \; . \tag{4.3}$$

Unfortunately, computing $p_\theta(\boldsymbol{x})$ to obtain $\boldsymbol{z}$, as defined as Eq. 4.2, is quite difficult, or even intractable in many cases. It is thus necessary to approximate the posterior distribution $p_\theta(z|\boldsymbol{x})$ by another parametric distribution $q_\phi(z|\boldsymbol{x})$ with $\boldsymbol{\phi}$ as the learnable parameters. In this way, the overall problem can be translated into the autoencoder framework, in which

- the approximation function $q_\phi(z|\boldsymbol{x})$ plays a similar role as the encoder function $f(\boldsymbol{x};\boldsymbol{\phi})$ in AE, but involving a re-parameterization process to generate samples from the latent space.
- the conditional probability $p_\theta(\boldsymbol{x}|z)$ defines a generative model, also know as probabilistic decoder that servers as the decoder function $g(z;\boldsymbol{\theta})$ in AE.

4.2.1 ELBO loss function

For variational autoencoders, we need to define a differentiable loss function in order to optimize both the generative model (decoder) parameters $\boldsymbol{\theta}$ and estimation (encoder) parameters $\boldsymbol{\phi}$ by minimizing the reconstruction error and the distance between the estimated distribution $q_\phi(z|\boldsymbol{x})$ and the real one $p_\theta(z|\boldsymbol{x})$. In practice, mean squared error and cross entropy represent good options for the reconstruction loss $\mathcal{L}\left(\boldsymbol{x},\boldsymbol{r}\right)$, and the reverse Kullback-Leibler (KL) divergence[1] is a good choice to measure the distance between the two distributions. The reverse KL divergence between $q_\phi\left(z|\boldsymbol{x}\right)$ and $p_\theta\left(z|\boldsymbol{x}\right)$ is given as

$$\begin{aligned}
&KL\left(q_\phi\left(z|\boldsymbol{x}\right) \;||\; p_\theta\left(z|\boldsymbol{x}\right)\right) \\
&= \int q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})}{p_\theta(z|\boldsymbol{x})} dz \\
&= \int q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})p_\theta(\boldsymbol{x})}{p_\theta(z,\boldsymbol{x})} dz && \text{; Because } p(z|x){=}p(z,x)/p(x) \\
&= \int q_\phi(z|\boldsymbol{x}) \left(\log p_\theta(\boldsymbol{x}) + \log \frac{q_\phi(z|\boldsymbol{x})}{p_\theta(z,\boldsymbol{x})}\right) dz \\
&= \log p_\theta(\boldsymbol{x}) + \int q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})}{p_\theta(z,\boldsymbol{x})} dz && \text{; Because } \textstyle\int q(z|x)dz{=}1 \\
&= \log p_\theta(\boldsymbol{x}) + \int q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})}{p_\theta(\boldsymbol{x}|z)p(z)} dz && \text{; Because } p(z,x){=}p(x|z)p(z) \\
&= \log p_\theta(\boldsymbol{x}) + \mathbb{E}_{z\sim q_\phi(z|\boldsymbol{x})}[\log \frac{q_\phi(z|\boldsymbol{x})}{p(z)} - \log p_\theta(\boldsymbol{x}|z)] \\
&= \log p_\theta(\boldsymbol{x}) + KL\left(q_\phi\left(z|\boldsymbol{x}\right) \;||\; p(z)\right) - \mathbb{E}_{z\sim q_\phi(z|\boldsymbol{x})} \log p_\theta(\boldsymbol{x}|z) \; .
\end{aligned} \tag{4.4}$$

[1] KL divergence is not symmetric. Commonly, $KL(p\;||\;q)$ is defined as the forward KL divergence and $KL(q\;||\;p)$ is the reverse KL divergence, where p denotes a prior/true probability distribution and q is a 'prediction' distribution.

Once re-arranging the left and right hand side of the above equation, we can write the loss function as

$$\begin{aligned}\mathcal{L}_{\theta,\phi} &= -\log p_\theta(\boldsymbol{x}) + KL\left(q_\phi\left(z|\boldsymbol{x}\right) \,||\, p_\theta\left(z|\boldsymbol{x}\right)\right) \\ &= -\mathbb{E}_{z\sim q_\phi(z|\boldsymbol{x})} \log p_\theta(\boldsymbol{x}|z) + KL\left(q_\phi\left(z|\boldsymbol{x}\right) \,||\, p\left(z\right)\right) \ .\end{aligned} \tag{4.5}$$

In Variational Bayesian methods, this loss function is also known as evidence lower bound (ELBO) loss function. The "lower bound" part in the name comes from the fact that KL divergence is always non-negative and it is thus correct to assert that $-\mathcal{L}_{\theta,\phi} \leq \log p_\theta(\boldsymbol{x})$.

4.2.2 Re-parameterization

The expectation term in the ELBO loss function invokes generating stochastic samples from the latent space to feed the probabilistic decoder. Such stochastic sampling is a non-differentiable operation and thus we cannot backpropagate the gradient. To make the ELBO formulation suitable for training, the re-parameterization trick is introduced with an assumption that the latent space can be considered as multivariate Gaussian distributions, i.e.,

$$\begin{aligned}&z \sim q_\phi(z|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2) \\ &z = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}) \quad \text{; Re-parameterization.}\end{aligned} \tag{4.6}$$

The re-parameterization trick also works for other distributions different from Gaussian. In the multivariate Gaussian case, we make the VAE model trainable by learning the mean and standard deviation of the variational distribution, i.e. $(\boldsymbol{\mu}, \boldsymbol{\sigma}) = f_\phi(\boldsymbol{x}; \boldsymbol{\phi})$, explicitly using the re-parameterization trick, while the stochasticity remains in the random variable $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ that is excluded from the updating process as shown in Figure 4.2.

The overall differentiable loss function for this VAE model can thus be given as

$$\mathcal{L}_{\theta,\phi} = \mathcal{L}\left(\boldsymbol{x}, \boldsymbol{r}\right) + KL\left(\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\right) \,||\, \mathcal{N}\left(0, \mathbf{I}\right)\right) \ . \tag{4.7}$$

It consists of two terms. The first one represents the reconstruction loss as discussed earlier in AEs and the second term (KL-loss) ensures that our learned distribution $q_\phi(z)$ is close to the prior distribution $p(z)$, i.e. our assumption in this case, a multivariate normal distribution.

5 Graph neural networks

Graph neural networks (GNNs) are neural network models that capture the dependence of graphs via message passing between the nodes of graphs. In recent years, variants of GNNs have demonstrated salient performances on many deep learning tasks, such as social networks [77, 78], bio-chemistry [79, 80], and so on. Variants of GNNs have also been increasingly explored in various image analysis tasks that include image classification [81, 82], semantic segmentation [83, 84, 85, 86, 87, 88], few-shot and zero-shot learning [89, 90, 91], and have demonstrated very promising performance for various image-level reasoning tasks while significantly reducing the computational cost [81].

In this chapter, we briefly review the graph neural networks that provide the backbone of Papers II-IV. Here we will limit the discussion to some relevant background about the graph theory and a few representative GNN models.

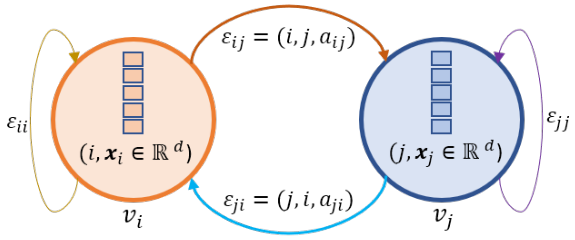
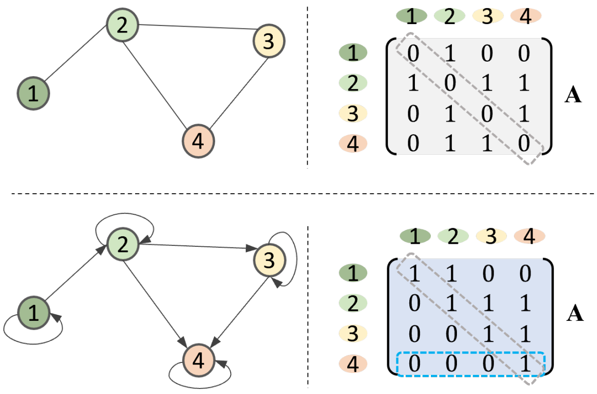

$\varepsilon_{ij} = (i, j, a_{ij})$
ε_{ii}
$(i, \boldsymbol{x}_i \in \mathbb{R}^d)$
$(j, \boldsymbol{x}_j \in \mathbb{R}^d)$
ε_{jj}
$\varepsilon_{ji} = (j, i, a_{ji})$
v_i
v_j

Figure 5.1: The concept diagram of graph nodes and edges

5.1 Graph definition

Consider a graph $G = (V, E)$ that consists of a set $V = \{v_i = (i, \boldsymbol{x}_i) : i = 1, 2, \ldots, n\}$ 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 $\mathrm{E} = \{\varepsilon_{ij} = (i, j, \alpha_{ij}) : i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, n\}$, where α_{ij} represents the weight associated to the node pair (v_i, v_j) directed from v_i to v_j as shown in Figure 5.1. The graph, G, can be also represented by $(\mathbf{A}, \mathbf{X})$, where the adjacency matrix[1] $\mathbf{A} = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nn} \end{bmatrix} \in \mathbb{R}^{n \times n}$ is composed of each link weight $\alpha_{ij} \geq 0 \in \mathbb{R}$, and the feature matrix $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n] \in \mathbb{R}^{n \times d}$ contains feature vectors for each node.

Figure 5.2 illustrates simplified directed and undirected graphs with their associated adjacency matrices having a self-loop (i.e., all diagonal entries are 1) and no self-loop (i.e., all diagonal entries are 0), respectively.



$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 0 & 1 & 0 & 1 \\ 4 & 0 & 1 & 1 & 0 \end{array} \quad \mathbf{A}$$

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 0 \\ 2 & 0 & 1 & 1 & 1 \\ 3 & 0 & 0 & 1 & 1 \\ 4 & 0 & 0 & 0 & 1 \end{array} \quad \mathbf{A}$$

Figure 5.2: Examples of directed and undirected graphs with their adjacency matrices. Top: an undirected graph with its representing adjacency matrix (symmetric) without self-loop. Bottom: a directed graph with its adjacency matrix (asymmetric) having self-loop.

5.2 Message passing

A GNN generalizes the convolution operator to irregular/structure domains. This may be expressed as a "message passing" or a "neighborhood aggregation" scheme in the graph structures [92]. 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

$$\mathbf{Z}^{(l)} = \mathrm{MP}\left(\tilde{\mathbf{A}}, \mathbf{Z}^{(l-1)}; \boldsymbol{\theta}^{(l)}\right), l = 1, 2, \ldots, k\ , \tag{5.1}$$

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

[1] The adjacency matrix **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.

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 $\tilde{\mathbf{A}}$ based on the specific GNN variant [93, 79].

The most common GNNs follow the message-passing strategy that can be generalized as Eq. 5.1. A simplified message passing process on a 4-node directed graph with self-loop is shown in Figure 5.3.

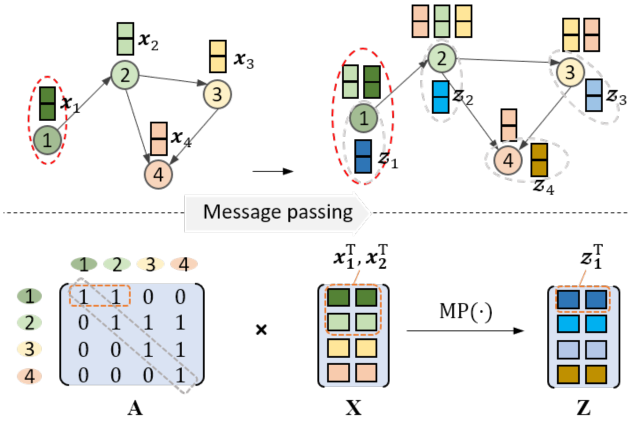


Figure 5.3: An illustration of message passing process on a simple directed graph. Top left: the graph containing 4 nodes with corresponding feature vectors, i.e., x_1, x_2, x_3 and x_4. Bottom left: the representing adjacency matrix $\mathbf{A}$ and the associated feature matrix $\mathbf{X}$, note that each row of $\mathbf{X}$ represents the features of each node. Top right: the message passing process, e.g., for node-1, first gathers (e.g., sum) all each node's neighborhood features (e.g., $x_1 + x_2$), and then learns a new representation for each node (e.g., z_1). Bottom right: the message passing pipeline with a dense representation, e.g., $\mathbf{Z} = \mathrm{MP}(\mathbf{AX}; \boldsymbol{\theta})$.

There are many kinds of implementations of the message-passing function. In this section, we mainly exploit two representative GNN variants: the spectral-based method - Graph Convolutional Network (GCN) [93], and the spatial-based method - Graph Isomorphism Network (GIN) [79].

5.3 Spectral GCNs

Spectral GCNs were first proposed by Bruna et al. [94] to define parameterized filters derived from spectral graph theory that can be used in a multi-layer neural network model, akin to "classical" CNNs. Defferrard et al. [95] improved it by approximating smooth filters in the spectral domain using Chebyshev polynomials. Simplifications were further introduced by Kipf and Welling [93] to allow for faster training times and higher performance in many cases. We will follow the notation of Kipf and Welling to briefly summarize the main idea behind GCNs.

The GCN proposed by Kipf and Welling [93] was presented as the first-order approximation of the spectral GNN [96], and implements a message-passing function by a combination of linear transformations over one-hop neighbourhoods and non-linearities. It is defined as

$$\mathbf{Z}^{(l)} = \delta\left(\tilde{\mathbf{A}}\mathbf{Z}^{(l-1)}\boldsymbol{\theta}^{(l)}\right) , \tag{5.2}$$

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

$$\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{\frac{1}{2}} . \tag{5.3}$$

Here $\mathbf{D}$ denotes 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. 5.3) 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 [98].

5.4 Spatial GNNs

Unlike the spectral-based GNN, GIN [79] 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

$$z_i^{(l)} = \mathrm{MLP}^{(l)}\left(\left(1+\omega^{(l)}\right)z_i^{(l-1)} + \sum_{j\in \mathrm{N}_{v_i}} z_j^{(l-1)}\right) , \tag{5.4}$$

where $z_i^{(l)} \in \mathbf{Z}^{(l)}$ 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 MLP at layer l, and $\omega^{(l)}$ is a learnable parameter or a fixed scalar at layer l. Eq. 5.4 can be converted to a dense/matrix representation as

$$\mathbf{Z}^{(l)} = \delta\left(\left(\omega^{(l)}\mathbf{I} + (\mathbf{I}+\mathbf{A})\right)\mathbf{Z}^{(l-1)}\boldsymbol{\theta}^{(l)}\right) . \tag{5.5}$$

Comparing Eq. 5.5 to Eq. 5.2, the major difference between the GIN and the GCN is that the normalized adjacency matrix $\tilde{\mathbf{A}}$ 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.

[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.

[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 [97].

6

Deep segmentation networks

This chapter provides a short overview of deep segmentation networks and applications in remote sensing. This is relevant for all papers.

The semantic segmentation task consists of classifying each pixel of an image into a specific thematic class. This is a key challenge in scene understanding and interpretation. In the remote sensing domain, image segmentation can be used for land cover mapping that refers to the assignment of a specific land-cover (semantic) category to every pixel in remote sensing imagery as shown in Figure 1.1. Although traditional shallow models (e.g., PCNN [99], SVMs [100], Random Forest [101]) for image segmentation have been explored in the past, they heavily relied on hand-crafted features or super-pixel maps to generate pixel-wise predictions. In this work, we will limit the discussion to deep learning based methods, such as CNNs, that are end-to-end trainable for image segmentation.

6.1 Architectures

6.1.1 Fully convolutional networks

Long et al. [102] proposed a fully convolutional network (FCN) for semantic segmentation, which was one of the first high impact CNN-based segmentation models. The FCN does not contain any fully connected layers and directly produces pixel-wise predictions in an end-to-end trainable way. The authors utilized deconvolutions (fractional strided convolutions) or bilinear interpolation for gradual upsampling, from which the pixel-wise output can be generated. Furthermore, shallower layer activation maps were proposed to be fused into the output to retain high-spatial-resolution information as the convolved input data flows deeper into the network.

Zhao et al. [103] improved the feature fusing operation in the FCN by using a spatial pyramid pooling module, which is the so called pyramid scene parsing network (PSPNet). The proposed spatial pyramid pooling module aims to encode multi-scale contextual information. It makes use of multiple pooling operations and CNNs followed by upsampling and concatenation layers to produce a multi-scale contextual representation from the incoming features, commonly the activation maps from the last convolution layer in the encoder. Finally, the multi-scale contextual representation is fed into a convolution layer to calculate the final pixel-wise prediction. Further, Chen et al. [62] proposed DeepLabv3+ that encodes multi-scale contextual information by using several parallel dilated convolutions with different rates (called Atrous Spatial Pyramid Pooling). The encoder features are first bilinearly upsampled and then concatenated with the corresponding low-level features. Because the low-level features typically contain a large number of channels, which may outweigh the importance of the rich encoder features, there is 1×1 convolution applied on the low-level features to reduce the number of channels before concatenation. Finally, a few 3×3 convolutions are used to refine the features and produce the predictions.

6.1.2 Encoder-decoder networks

Current popular approaches, such as SegNet [104] and U-Net [105], make us of so-called encoder-decoder segmentation architectures [106], and often consist of an encoder network (a sequence of non-linear processing layers, such as CNN+BN+ReLU+Pooling), and a corresponding decoder layers (typically CNNs plus Upsampling) that is followed by a final pixel-wise classification layer (such as CNN+Softmax). This architecture is illustrated in Figure 6.1.

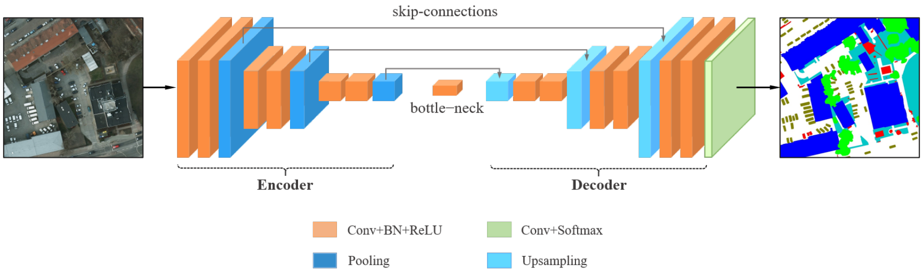


Figure 6.1: Overview of the encoder-decoder segmentation network. Skip connections, via addition or concatenation, are commonly used to pass features from the encoder path to the decoder path in order to recover spatial information lost during pooling (downsampling).

Specifically, SegNet [104] has a symmetric encoder-decoder structure. The decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling. This eliminates the need for learning to upsample. The upsampled maps are sparse and are then convolved with trainable filters to produce dense feature maps. Instead of keeping track of the pooling indices, U-Net [105] proposed skip connections between the encoder and decoder modules. The spatial information is able to be gradually recovered in the decoder module by fusing skipped connections (e.g. via concatenation) from the encoder module with upsampling (e.g. via transpose convolution) layers to improve the segmentation details. Adding skip-connections to the encoder-decoder networks do not only improve the model's accuracy but also address the problem of vanishing gradients.

There are many modified versions, e.g. adding extra fusion or attention blocks in encoder-decoder networks, which have been widely applied to semantic segmentation of nature images, i.e. on the PASCAL VOC 2012 benchmark [107]. Examples are the ExFuse network [108], the Dual attention network [109], and the HRNet-OCR [110].

6.2 Applications to remote sensing

The FCNs and encoder-decoder architectures have also been widely adapted and applied to remote sensing domain, such as the ISPRS [111] Semantic Labeling Contest [112, 113, 114, 115, 116, 16, 117, 118, 119, 120], and the DeepGlobe CVPR-2018 [10] challenge of automatic classification of land cover types [121, 122, 123, 124, 125, 126, 127].

For instance, Sherrah [113] applied FCNs for semantic labelling of aerial imagery and illustrated that higher accuracy can be achieved than with more traditional patch-based approaches. Similarly, the stacked U-Nets architecture proposed in [125] for land cover segmentation in remote sensing imagery merges high-resolution details and long distance context information captured at low-resolution to generate segmentation maps. Further, Kuo et al. [126] introduced an aggregation decoder in combination with DeepLabV3 architecture to fuse different-level features progressively from the encoder for final prediction, while the authors of [127] proposed a dense fusion classmate network (DFCNet) that tried to fuse auxiliary training data as "classmate" to capture supplementary features for land cover classification. One of the main ideas behind all the architectures is to take into account the multi-level context to improve the prediction of the segmentation. In general, these models differ from each other in how they capture rich contextual information at multiple scales, such as how to model local and non-local information of complex-shaped and context-dependable objects. For instance, the car is more likely found on the road than on the roof of a building.

In addition, deep learning has also been exploited for multi-modal data processing in remote sensing. For example, Audebert et al. [16] proposed a multi-scale SegNet approach (so-called FuseNet) to leverage both a large spatial context and the high resolution data, while early and late fusion strategies of multi-modality data are also exploited. However, such fusion techniques require that all modalities to be available to the classification during both training and testing. Later, Kampffmeyere eta al. [119] presented a novel CNN architecture based on so-called hallucination networks for urban land cover classification that were able to replace missing data modalities in the test phase. This enables fusion capabilities even when data modalities are missing in the test phase.

6.3 Evaluation metrics

Intuitively, a successful segmentation model is one which maximizes the overlap between the predicted and true regions, since semantic segmentation is simply the act of differentiating (recognizing) objects (regions) in an image based on their different semantic properties. There are two popular overlap-based evaluation metrics for this goal, the Dice (i.e. F1-score) and Jaccard coefficients or index (also known as the intersection over union - IoU):

$$Dice(A,B) = \frac{2\|A \cap B\|}{\|A\| + \|B\|}\,, \qquad Jaccard(A,B) = \frac{\|A \cap B\|}{\|A \cup B\|}\,. \tag{6.1}$$

Here, A and B denote two segmentation masks for a given class, $\|\cdot\|$ denotes the norm of a given mask (for images, the area in pixels), and $\cap$, $\cup$ are the intersection and union operators. Both the Dice and Jaccard indices are bounded between 0 (when there is no overlap) and 1 (when A and B overlaps). Figure 6.2 gives an simple illustration of the Dice and Jaccard metrics.

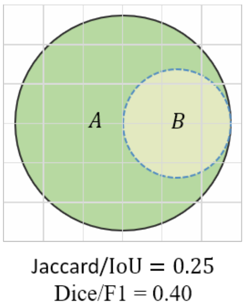


Figure 6.2: An illustration of the Dice and Jaccard coefficients given two circles representing the ground truth (A) and the predicted masks (B) for an arbitrary object.

In terms of the confusion matrix, the metrics can be calculated as following:

$$\text{Dice} = \text{F1-score} = \frac{2TP}{2TP + FP + FN}\,, \qquad \text{Jaccard} = \text{IoU} = \frac{TP}{TP + FP + FN} \tag{6.2}$$

Here, TP - true positive, TN - true negative, FP - false positive, and FN - false negative. In general, Dice and IoU are numerically quite comparable and both are widely used in computer vision applications.

Part II

Summary of research and conclusion

7

Paper I

Dense Dilated Convolutions Merging Network for Land Cover Classification

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
Journal of IEEE Transactions on Geoscience and Remote Sensing, vol 58.9, pp 6309-6320, doi:10.1109/TGRS.2020.2976658, 2020.

In this work, we propose a novel DL architecture called the Dense Dilated Convolutions Merging Network (DDCM-Net) for land cover classification of remote sensing images. The proposed DDCM-Net consists of dense dilated convolutions' merging (DDCM) modules with varying dilation rates. The DDCM module learns with densely linked multiple dilated convolutions and outputs a fusion of all intermediate features while extracting multi-scale features at varying dilation rates without reducing resolutions. This decreases computational redundancy and costs substantially. DDCM's computational efficiency and performance could be improved further by combining grouped convolutions and striated operations as demonstrated in our experiments. Hence, the proposed DDCM module is very light-weighted and scalable that can be used as a simple, yet effective, encoder or decoder module for semantic segmentation tasks.

Figure. 7.1 illustrates the end-to-end pipeline of the DDCM-Net combined with a pre-trained model for land cover classification. We demonstrate the effectiveness, robustness and flexibility of the proposed DDCM-Net on the publicly available ISPRS Potsdam and Vaihingen data, as well as the DeepGlobe land cover dataset. Our single model, trained on 3-band Potsdam (RGB-band) and Vaihingen (IRRG-band) data, achieves better accuracy in terms of both mean intersection over union (mIoU) and F1-score compared to other published models trained with more than 3-band data (such as RBG/IRRG + DSM). The variants of our DDCM-Net by using different combinations of dilations and densities for the DeepGlobe data set also demonstrated better performance but consumed much fewer computation resources compared with other published methods.

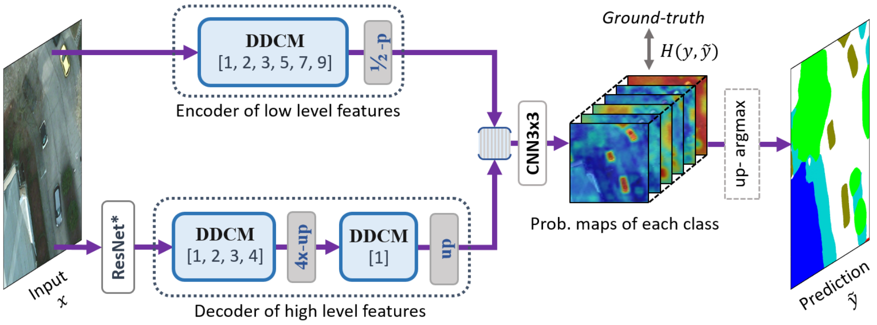


Figure 7.1: End-to.end pipeline of DDCM-Net for semantic mapping of VHR Potsdam images. The encoder of low level features encodes multi-scale contextual information from the initial input images by a DDCM module (output 3-channel) using 3×3 kernels with 6 different dilation rates $[1, 2, 3, 5, 7, 9]$. The decoder of high level features decodes highly abstract representations learned from a ResNet-based backbone (output 1024-channel) by 2 DDCM modules with rates $[1, 2, 3, 4]$ (output 36-channel) and $[1]$ (output 18-channel) separately. The transformed low-level and high-level feature maps by DDCMs are then fused together to infer pixel-wise class probabilities. Here, 'p' and 'up' denote pooling and up-sampling respectively.

7.1 Contributions by the author

- The idea was mainly conceived and developed by me with suggestions from my co-authors.
- I made all implementations and ran all experiments.
- I wrote the first draft of the manuscript and it was improved together with my co-authors.

/8

Paper II

Self-constructing Graph Neural Networks to Model Long-range Pixel Dependencies for Semantic Segmentation of Remote Sensing Images

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg

International Journal of Remote Sensing, vol 42.16, pp 6184-6208,
doi:10.1080/01431161.2021.1936267, 2021

Global dependency is very important for many dense classification tasks. It has been shown to improve semantic mapping performance by capturing rich non-local contextual representations in remote sensing images. However, ConvNets do not explicitly model the non-local information, particularly for pixel-wise prediction tasks, because pure CNN models, in general, are building blocks that process one local neighborhood at a time. Deep and dilated CNN layers are thus commonly used to increase the field of view in current approaches aimed at obtaining rich non-local context information. This significantly raises the model's complexity and memory consumption. Graph Neural Networks (GNNs) have recently emerged as more powerful and efficient models for capturing global dependencies than CNNs, but GNNs have been rarely deployed in dense prediction tasks in computer vision domain due to lack of prior knowledge or dependency graphs.

In this work, we propose the Self-Constructing Graph (SCG) module that learns long-range pixel-wise dependencies directly from image data. Various Graph models can then be applied on top of it to efficiently capture global contextual features for improving semantic segmentation. In other words, the SCG module offers a high degree of flexibility for building segmentation networks that seamlessly combine the benefits of GNNs and CNNs. For example, the SCG-GCN model, that is built upon SCG and graph convolutional networks (GCN) as shown in Figure 8.1,

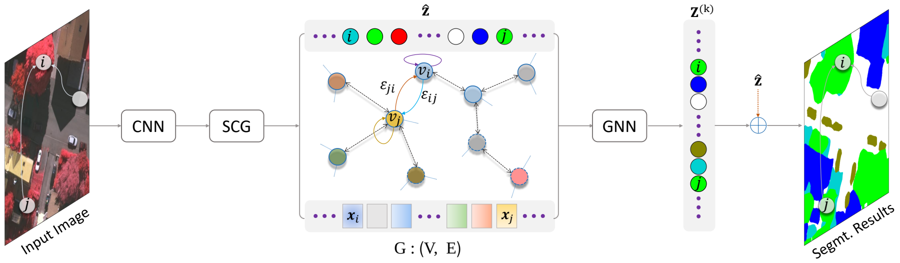


Figure 8.1: SCG-Net uses a conventional CNN backbone to learn a 2D feature map of an input image. The SCG module then learns to transform the 2D feature map into a latent graph structure G : (V, E), construct the global context relations (ε_{ij} ∈ E) and assign feature vectors ($\boldsymbol{x}_i$) to the vertices (v_i ∈ 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.

performs semantic segmentation in an end-to-end manner. It outperforms many related models based on pure CNNs on the publicly available ISPRS Potsdam and Vaihingen datasets while using fewer parameters and paying much lower computational costs. Our comprehensive ablation experiments 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.

8.1 Contributions by the author

- The idea was mainly developed by me, but with inputs from my co-authors.
- I made all implementations and ran all experiments.
- I wrote the first draft of the manuscript which was improved together with the co-authors.

9 Paper III

Multi-view Self-Constructing Graph Convolutional Networks with Adaptive Class Weighting Loss for Semantic Segmentation

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp 199-205, doi:10.1109/CVPRW50498.2020.00030, 2020

This paper is a direct successor of Paper II. In this work, we continue the exploration of self-constructing graph convolutional networks (SCG) for land cover classification, while we focus in this paper on the problem of rotational invariance and class imbalance in remote sensing. CNNs are empirically known to be invariant to moderate translation but not to rotation in image classification. Capturing rotation-invariance properties is desirable for deep learning applications in remote sensing because it can help to better predict the class regardless of the object's orientation.

Towards that end, we propose the Multi-view Self-Constructing Graph Convolutional Networks (MSCG-Net), a novel architecture that leverages multiple views to explicitly exploits rotational invariance for improving semantic mapping of multispectral airborne images. Our model utilizes deep-feature augmenting mechanisms (via rotating high-level features to different angles) with shared graph modules to learn robust representations by fusing multi-orientation information. Figure 9.1 shows the general architecture of our MSCG-Net.

Meanwhile, we also develop an adaptive class weighting loss to address the common class imbalance in remote sensing data. Unlike most existing weighted loss algorithms, which scale the loss for each class or pixel using pre-calculated class weights based on the full training data, our developed adaptive weighting algorithm can compute the class weights automatically and dynamically update during iterative training. We also introduce a novel PNC regularization (a proposed positive and negative class balanced function) combining a dice coefficient [128] into our adaptive class weighting loss. This provides an auto-dynamic-weighting solution that can reduce the class imbalance effect while also putting more emphasis on difficult both positive and negative examples during training.

We demonstrate the effectiveness and flexibility of the proposed network and loss function on the Agriculture-Vision challenge dataset and our model achieves very competitive results with much fewer parameters and at a lower computational cost compared to related work.

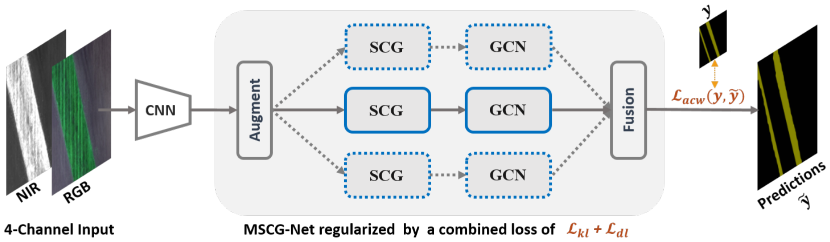

NIR
RGB
4-Channel Input
CNN
Augment
SCG
GCN
SCG
GCN
SCG
GCN
Fusion
y
$\mathcal{L}_{acw}(y,\tilde{y})$
Predictions
$\tilde{y}$
MSCG-Net regularized by a combined loss of $\mathcal{L}_{kl}+\mathcal{L}_{dl}$

Figure 9.1: Model architecture of MSCG-Net for semantic labeling includes the CNN-based feature extractor (e.g. customized Se_ResNext50_32x4d taking 4-channel input and output 1024-channel in this work), SCG module taking 3-view (augment the original input by 90°and180°) inputs and 2-layer GCNs, the Fusion block merging 3-view outputs together, the fused output is projected and upsampled back to 2D maps for final prediction.

9.1 Contributions by the author

- The idea was conceived by me and my co-authors and further developed by me.
- I implemented the proposed models and performed the experiments.
- I wrote the manuscript draft of the paper which was further improved in collaboration with the co-authors.

10

Paper IV

Multi-modal land cover mapping of remote sensing images using pyramid attention and gated fusion networks

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
In submission to Internationl Journal of Remote Sensing, September, 2021

Multi-modality data can provide complementary information of the same scene in earth observation (EO). Effective fusion of this multi-modality information is thus important for various EO applications, but also very challenging due to large domain differences, noise, and redundancies. Most existing multi-modal segmentation models mostly use two independent encoders in parallel to extract features separately from multi-modal data. However, there is a lack of effective and scalable fusion techniques for bridging multiple modality encoders in order to fully exploit complementary information and improve the model's overall performance.

To this end, we propose a new multi-modality segmentation network (MultiModNet) for land cover mapping of multi-modal remote sensing data. Our MultiModNet is built upon a novel pyramid attention fusino (PAF) module and gated fusion unit (GFU). Figure 10.1 shows the general concept architecture of MultiModNet. The PAF module is designed to efficiently obtain rich fine-grained contextual representations from each modality with a built-in cross-level and cross-view attention fusion mechanism, and the GFU module utilizes a novel gating mechanism for early features merging, thereby diminishing hidden redundancies and noise. This enables supplementary modalities to effectively extract the most valuable and complementary information for late feature fusion. Specifically, the PAF fused features from the primary/predecessor modality (e.g., the IRRG or RGB images) will be merged into the encoder of the supplementary/secondary modality (e.g., DSM or NIR) via a GFU module, which will automatically force the supplementary encoders to learn the most valuable information and diminish the influence of its redundancies and noises. Another PAF module (it can be optional shared weights with the main PAF module) will be further exploited by the secondary modality to capture fused features that will be combined together with the main PAF features.

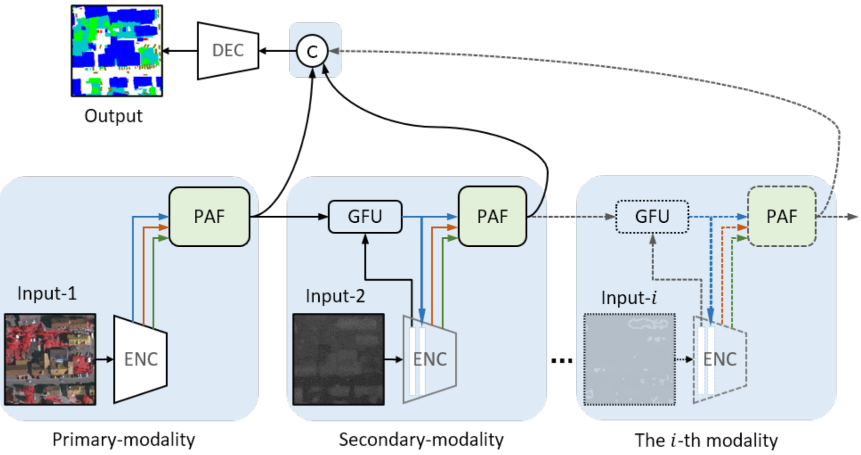


Figure 10.1: General concept structure of our MultiModNet. ENC denotes the feature encoder, GFU accounts for a gated fusion unit, PAF is our proposed pyramid attention fusion module, © denotes concatenation, and, DEC is the decoder layer to output the final classification.

Our experiments demonstrate that the network can achieve robust and accurate results on the ISPRS Semantic Labeling Contest Vaihingen dataset [12] and the Agriculture-Vision challenge dataset [11]. Particularly on the Agriculture-Vision dataset, our model outperforms the strong baselines with far fewer parameters and at a lower computational cost. Extensive ablation studies are also carried out to assess the effectiveness and robustness of our framework, such as how well the model can tolerate missing, noisy, or entirely interfering modality data during test.

10.1 Contributions by the author

- The idea was conceived by me and developed in collaboration with the co-authors of the paper.
- I made all implementations and ran all experiments.
- I wrote the main draft of the paper and it was improved together with the co-authors.

11

Concluding remarks

The research in this thesis develops novel deep learning methods for land cover mapping in the field of remote sensing. Specifically, we focused on developing CNN-based light-weight networks that can effectively extract rich multi-scale features to improve classification performance while keeping the model relatively simple and scalable. We thus proposed the DDCM network that makes use of multiple dilated convolutions with various dilation rates, and outputs a fusion of all intermediate features without losing resolutions during the extraction of multi-scale features that greatly reduces the computational redundancies and cost.

Further, we developed novel GNN-based methods that can efficiently model long-range pixel-wise dependency and capture non-local contextual representations with lower computational costs to achieve very competitive performance. Our proposed SCG-Net is able to directly learn long-range dependency from input images and fully leverage the benefits of both CNNs and variants of GNNs for improving semantic mapping performance. We later extended SCG-Net to MSCG-Net that is able to utilize multiple views to capture rotation-invariance properties in high-level representations for improving segmentation results. In addition, to address the class imbalanced issue, commonly found in remote sensing data, we developed a novel adaptive class weighting loss function built on an iterative-adaptive weighting technique with a new positive and negative regularization function. This can reduce the class imbalance effect while also gain more accurate predictions for both large and small difficult objects in remote sensing data.

Finally, we explored the task of multi-modal learning and proposed a novel light-weight and scalable architecture (MultiModNet) using pyramid attention and gated fusion methods to address the challenges related to multi-modal fusion. With our proposed MultiModNet, we can achieve cutting-edge performance on multi-modal and multi-spectral remote sensing land cover mapping at a low computational cost.

To summarize, the four research papers presented in this thesis are believed to advance remote sensing land cover mapping by exploring various state-of-the-art deep learning methodologies to address key challenges such as complex scenes and objects in very-high-resolution remote sensing imagery, highly imbalanced classes, multi-spectral and multi-modality learning, and light-weight yet effective modeling.

11.0.1 Limitations and future work

We acknowledge that every research paper has both strengths and weaknesses. Therefore, we discuss some limitations for the research presented in this thesis, and also suggest future work in this section.

Limitations

Paper I: Though our proposed method (DDCM) is able to capture larger multi-scale and richer context information and work well on different remote sensing data, it commonly demands delicate tuning of some sensitive hyperparameters (such as the dilation and densities settings). One has to manually fine-tune and make trade-offs on the dilation policies and the densities based on careful analysis of specific data and preliminary experiments. For example, since we observe that the DeepGlobe images contain more spatially chaotic objects with lower resolutions, larger scales, and less geometrical attributes than the ISPRS images, we thus configured a DDCM module for the DeepGlobe data with exponentially growing dilation rates ($[1, 2, 4, 8, 16, 32]$) rather than the linearly growing dilation rates ($[1, 2, 3, 5, 7, 9]$) used on ISPRS data. It would be interesting to design an approach to automatically or adaptively adjust these parameters [129] in the module to improve the model's robustness.

Paper II and III: Despite the promising performance of the proposed GNN based architecture, stacking more GNN layers in our model significantly hurts the training and test performance of our model. This is because the performance of GNNs is known to gradually decrease with increasing number of GNN layers, partly due to its over-smoothing issue. In other words, it means that repeatedly applying more graph convolutions eventually makes features of vertices indistinguishable.

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 model. Closely located small objects tend to be segmented together as a whole big object. Future studies are required to enhance the module's performance on small objects and the interpretability of the learned dependencies.

Though Paper III improved the SCG by considering multi-view inputs (augmented features with rotations), these augmented multi-view features are generated from one single-view raw image instead of multi-view raw data. It would be interesting to further extend the MSCG model to support raw multi-view images. Additionally, the adaptive class weighting (ACW) loss proposed in Paper III also needs to be carefully evaluated further on more datasets and compared with more related weighted loss functions.

Paper IV: One of the main limitations to the proposed multi-modal framework is that there is a prior assumption, i.e., one type of data with richer information should be manually selected as the primary modality that will be used to control the learning process of the secondary/supplementary modality. It works well when we know what modality contains more information. For instance, we know that RGB data commonly contains richer information than DSM. Thus, it is easy to confirm RGB images as the primary feature source rather than DSM. However, if the user does not know which type of information is better than others, such as SAR data vs LiDAR data, the proposed model may not work well or even fail. Therefore, it would be interesting to evaluate how the proposed method works with more than two modalities, such as including SAR data into the model, and further to improve the framework to be able to automatically learn and fuse multiple types of data without prior priority.

In addition, the proposed light-weight model still takes up a lot of memory usage on a GPU platform during training. Future work should be conducted to further compress the model using such as knowledge distillation-based methods [130] for less running time while remaining or even improving classification accuracy.

Future work

In this part, several thoughts are provided on potential research directions for the future in remote sensing.

A first promising research field is unsupervised and semi-supervised learning for the classification tasks in remote sensing to overcome the data-hungry issue. Since the most current classification algorithms are generally supervised deep learning models. This typically demands a large number of well-annotated data that are time-consuming and expensive to obtain. Most recently, the work by Castillo-Navarro et al. [131] introduces a novel large-scale dataset, the MiniFrance suite, for semi-supervised semantic segmentation in Earth Observation. The authors also present semi-supervised deep architectures based on multi-task learning and the first experiments on the dataset. These results will serve as very good baselines for future work on semi-supervised learning in the remote sensing domain.

A second important research direction is the development of domain adaptation methods that would be helpful to address another important problem, i.e., how to use the well-trained models by previous existing labeled datasets to accurately classify newly collected unlabeled data since a huge amount of new data is obtained every day by various kinds of remote sensors under different conditions. Because of different imaging platforms (e.g., satellites and unmanned aerial vehicles) or different imaging sensors and conditions (e.g., different time of the year), these variations between source and target domains are extremely prevalent in remote sensing images (optical, infrared , and SAR sensors). In general, these large gaps in data distribution between the source and the target domains will lead to large degradations in performance [132].

Finally, a last promising research direction is the study of few-shot or zero-shot learning that aims to learn a model that can quickly generalize to new tasks from very few labeled objects [91]. Few-shot learning might considerably alleviate the burden of data collection, particularly in the field of remote sensing, where collecting labeled instances is time-consuming and labor-intensive. The most existing works primarily focus on image-level classification tasks, while other typical interpretation tasks, such as semantic segmentation and object detection, the learning algorithms still suffer from the burden of data labeling. Hence, the few-shot learning for remote sensing data will be of great need for further improvements.

Part III

Included Papers

12

Paper I

Dense Dilated Convolutions' Merging Network for Land Cover Classification

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
IEEE Transactions on Geoscience and Remote Sensing, vol 58.9, pp 6309-6320, dio:10.1109/TGRS.202.2976658, 2020

/13

Paper II

Self-constructing Graph Neural Networks to Model Long-range Pixel Dependencies for Semantic Segmentation of Remote Sensing Images

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
International Journal of Remote Sensing, vol 42.16, pp 6184-6208, dio:10.1080/01431161.2021.1936267, 2021

14

Paper III

Multi-view Self-Constructing Graph Convolutional Networks with Adaptive Class Weighting Loss for Semantic Segmentation

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg
IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 199-205, doi:10.1109/CVPRW50498.2020.00030, 2020

Multi-view Self-Constructing Graph Convolutional Networks with Adaptive Class Weighting Loss for Semantic Segmentation

Qinghui Liu[1,2], Michael Kampffmeyer[2], Robert Jenssen[2,1], Arnt-Børre Salberg[1]
[1]Norwegian Computing Center, Oslo, NO-0314, Norway
[2]UiT Machine Learning Group, UiT the Arctic University of Norway, Tromsø, Norway
liu@nr.no, {michael.c.kampffmeyer, robert.jenssen}@uit.no, salberg@nr.no

Abstract

We propose a novel architecture called the Multi-view Self-Constructing Graph Convolutional Networks (MSCG-Net) for semantic segmentation. Building on the recently proposed Self-Constructing Graph (SCG) module, which makes use of learnable latent variables to self-construct the underlying graphs directly from the input features without relying on manually built prior knowledge graphs, we leverage multiple views in order to explicitly exploit the rotational invariance in airborne images. We further develop an adaptive class weighting loss to address the class imbalance. We demonstrate the effectiveness and flexibility of the proposed method on the Agriculture-Vision challenge dataset [1] *and our model achieves very competitive results (0.547 mIoU)* [2] *with much fewer parameters and at a lower computational cost compared to related pure-CNN based work.*

1. Introduction

Currently, the end-to-end semantic segmentation models are mostly inspired by the idea of fully convolutional networks (FCNs) [14] that generally consist of an encoder-decoder architecture. To achieve higher performance, CNN-based end-to-end methods normally rely on deep and wide multi-scale CNN architectures to create a large receptive field in order to obtain strong local patterns, but also capture long range dependencies between objects of the scene. However, this approach for modeling global context relationships is highly inefficient and typically requires a large number of trainable parameters, considerable computational resources, and large labeled training datasets.

Recently, graph neural networks (GNNs) [1] and Graph Convolutional Networks (GCNs) [8] have received increasing attention and have been applied to, among others, image classification [10], few-shot and zero-shot classification [4], point clouds classification [16] and semantic segmentation [11]. However, these approaches are 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 [11]. In order to address this problem and learn a latent graph structure directly from 2D feature maps for semantic segmentation, the Self-Constructing Graph module (SCG) [13] was recently proposed and has obtained promising results.

In this work, we extend the SCG to explicitly exploit the rotation invariance in airborne images, by extending it to consider multiple views. More specifically, we augment the input features to obtain multiple rotated views and fuses the multi-view global contextual information before projecting the features back onto the 2-D spatial domain. We further propose a novel adaptive class weighting loss that addresses the issue of class imbalance commonly found in semantic segmentation datasets. Our experiments demonstrate that

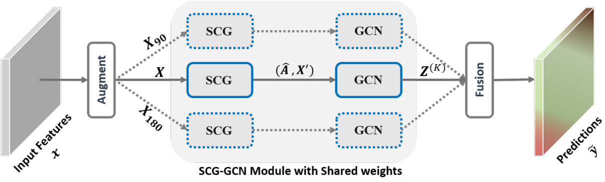


Figure 1. Overview of the MSCG-Net. The Self-Constructing Graph module (SCG) learns to transform a 2D feature map into a latent graph structure and assign pixels (X') to the vertices of the graph ($\hat{A}$). The Graph Convolutional Networks (GCN) is then exploited to update the node features ($Z^{(K)}$, K here denotes the number of layer of GCN) along the edges of graph. The combined module of SCG and GCN (SCG-GCN) can takes augmented multi-view input features (X, X_{90} and X_{180}, where the index indicates degree rotation) and finally the updated multi-view representations are fused and projected back onto 2D maps.

[1] https://www.agriculture-vision.com/dataset

[2] The leaderboard at: https://competitions.codalab.org/competitions/23732?secret_key=dba10d3a-a676-4c44-9acf-b45dc92c5fcf#results

the MSCG-Net achieves very robust and competitive results on the Agriculture-Vision challenge dataset, which is a subset of the Agriculture-Vision dataset [2].

The rest of the paper is organized as follows. In the method Section 2, we present the methodology in details. Experimental procedure and evaluation of the proposed method is performed in Section 3. Finally in Section 4, we draw conclusions.

2. Methods

In this section, we briefly present graph convolutional networks and the self-constructing graph (SCG) approach that are the foundation of our proposed model, before presenting our end-to-end trainable Multi-view SCG-Net (MSCG) for semantic labeling tasks with the proposed adaptive class weighting loss.

2.1. Graph Convolutional Networks

Graph Convolutional Networks (GCNs) [8] are neural networks designed to operate on and extract information from graphs and were originally proposed for the task of semi-supervised node classification. $G = (A, X)$ denotes an undirected graph with n nodes, where $A \in \mathbb{R}^{n\times n}$ is the adjacency matrix and $X \in \mathbb{R}^{n\times d}$ is the feature matrix. At each layer, the GCN aggregates information in one-hop neighborhoods, more specifically, the representation at layer $l+1$ is computed as

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

where $\theta^{(l)} \in \mathbb{R}^{d\times f}$ are the weights of the GCN, $Z^{(0)} = X$, and $\hat{A}$ is the symmetric normalization of A including self-loops [13]:

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

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

Note, in the remainder of the paper, we use $Z^{(K)} = \text{GCN}(A, X)$ to denote the activations after a K-layer GCN. However, in practice the GCN could be replaced by alternative graph neural network modules that perform K steps of message passing based on some adjacency matrix A and input node features X.

2.2. Self-Constructing Graph

The Self-Constructing Graph (SCG) module [13] allows the construction of undirected graphs, capturing relations across the image, directly from feature maps, instead of relying on prior knowledge graphs. It has achieved promising performance on semantic segmentation tasks in remote sensing and is efficient with respect to the number of trainable parameters, outperforming much larger models. It is inspired by variational graph auto-encoders [9]. A feature map $X \in \mathbb{R}^{h\times w\times d}$ consisting of high-level features, commonly produced by a CNN, is converted to a graph $G = (\hat{A}, X')$. $X' \in \mathbb{R}^{n\times d}$ are the node features, where $n = h' \times w'$ denotes the number of nodes and where $(h' \times w') \leq (h \times w)$. Parameter-free pooling operations, in our case adaptive average pooling, are employed to reduce the spatial dimensions of X to h' and w', followed by a reshape operation to obtain X'. $\hat{A} \in \mathbb{R}^{n\times n}$ is the learned weighted adjacency matrix.

The SCG module learns a mean matrix $\boldsymbol{\mu} \in \mathbb{R}^{n\times c}$ and a standard deviation matrix $\boldsymbol{\sigma} \in \mathbb{R}^{n\times c}$ of a Gaussian using two single-layer convolutional networks. Note, following convention with variational autoencoders [7], the output of the model for the standard deviation is $\log(\sigma)$ to ensure stable training and positive values for σ. With help of reparameterization, the latent embedding Z is $Z \leftarrow \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\varepsilon}$ where $\varepsilon \in \mathbb{R}^{N'\times C}$ is an auxiliary noise variable and initialized from a standard normal distribution ($\boldsymbol{\varepsilon} \sim \text{N}(0, I)$). A centered isotropic multivariate Gaussian prior distribution is used to regularize the latent variables, by minimizing a Kullback-Leibler divergence loss

$$\mathcal{L}_{kl} = -\frac{1}{2n}\sum_{i=1}^{n}\left(1 + \log\left(\sigma_i\right)^2 - \mu_i^2 - \sigma_i^2\right) . \qquad (3)$$

Based on the learned embeddings, A' is computed as $A' = \text{ReLU}(ZZ^T)$, where $A'_{ij} > 0$ indicates the presence of an edge between node i and j.

Liu et al. [13] further introduce a diagonal regularization term

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

where γ is defined as

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

and a diagonal enhancement approach

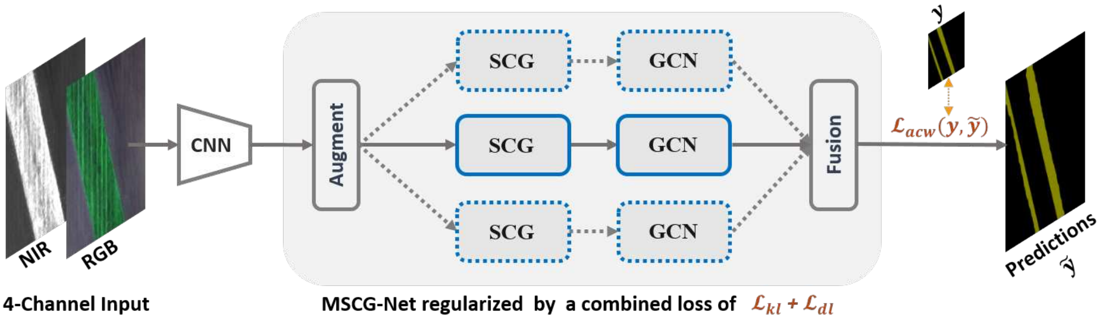
$$A^{\star} = A' + \gamma \cdot \text{diag}(A') \qquad (5)$$

to stabilize training and preserve local information.

The symmetric normalized $\hat{A}$ that SCG produces and that will be the input to later graph operations is computed as

$$\hat{A} = D^{-\frac{1}{2}}\left(A^{\star} + I\right)D^{\frac{1}{2}} . \qquad (6)$$

The SCG further produces an adaptive residual prediction $\hat{\boldsymbol{y}} = \gamma \cdot \boldsymbol{\mu} \cdot (1 - \log \boldsymbol{\sigma})$, which is used to refine the final prediction of the network after information has been propagated along the graph.



NIR
RGB
4-Channel Input
CNN
Augment
SCG
GCN
SCG
GCN
SCG
GCN
Fusion
y
$\mathcal{L}_{acw}(y, \tilde{y})$
Predictions
$\tilde{y}$
MSCG-Net regularized by a combined loss of $\mathcal{L}_{kl} + \mathcal{L}_{dl}$

Figure 2. Model architecture of MSCG-Net for semantic labeling includes the CNN-based feature extractor (e.g. customized Se_ResNext50_32x4d taking 4-channel input and output 1024-channel in this work), SCG module taking 3-view (augment the original input by 90°and180°) inputs and K-layer GCNs (K=2 in this work), the Fusion block merging 3-view outputs together, the fused output is projected and upsampled back to 2D maps for final prediction.

2.3. The MSCG-Net

We propose a so-called Multi-view SCG-Net (MSCG-Net) to extend the vanilla SCG and GCN modules by considering multiple rotated views in order to obtain and fuse more robust global contextual information in airborne images in this work. Fig. 2 shows an illustration of the end-to-end MSCG-Net model for semantic labeling tasks. The model architecture details are shown in Table 2.3. We first augment the features (X) learned by a backbone CNN network to multiple views (X_{90} and X_{180}) by rotating the features. The employed SCG-GCN module then outputs multiple predictions: $(\hat{\boldsymbol{y}}, Z^{(2)}), (\hat{\boldsymbol{y}}_{90}, Z^{(2)}_{90}), (\hat{\boldsymbol{y}}_{180}, Z^{(2)}_{180})$ with different rotation degrees (the index indicates the degree of rotation). The fusion layer merges all the predictions together by reversed rotations and element-wise additions as shown in Table 2.3. Finally, the fused outputs are projected and up-sampled back to the original 2-D spatial domain.

We utilize the first three bottleneck layers of a pretrained Se_ResNext50_32x4d or Se_ResNext101_32x4d [3] as the backbone CNN to learn the high-level representations. The output size of the CNN is $\frac{h}{16} \times \frac{w}{16} \times 1024$. Note that, we duplicate the weights corresponding to the Red channel of the pretrained input convolution layer in order to take NIR-RGB 4 channels in the backbone CNN, and GCNs (Equation 1) are used in our model. We utilize ReLU activation and batch normalization only for the first layer GCN. Note, we set $n = 32^2$ and $d = 128$ in this work, and c here is equal to the number of classes, such that $c = 7$ for the experiments performed in this paper.

2.4. Adaptive Class Weighting Loss

The distribution of the classes is highly imbalanced in the dataset (e.g. most pixels in the images belongs to the background class and only few belong to classes such as planter skip and standing water). To address this problem, most existing methods make use of weighted loss functions with pre-computed class weights based on the pixel frequency of the entire training data [5] to scale the loss for each class-pixel according to the fixed weight before computing gradients. In this work, we introduce a novel class weighting method based on iterative batch-wise class rectification, instead of pre-computing the fixed weights over the whole dataset.

The proposed adaptive class weighting method is derived from median frequency balancing weights [5]. We first compute the pixel-frequency of class j over all the past training steps as follows

$$f^t_j = \frac{\hat{f}^t_j + (t-1) * f^{t-1}_j}{t} \, . \tag{7}$$

where, $t \in \{1, 2, ..., \infty\}$ is the current training iteration number, $\hat{f}^t_j$ denotes the pixel-frequency of class j at the current t-th training step that can be computed as $\frac{\mathrm{SUM}(y_j)}{\sum_{j \in C} \mathrm{SUM}(y_j)}$, and $f^0_j = 0$.

The iterative median frequency class weights can thus be computed as

$$w^t_j = \frac{\mathrm{median}(\{f^t_j | j \in C\})}{f^t_j + \epsilon} \, . \tag{8}$$

here, C denotes the number of labels (7 in this paper), and $\epsilon = 10^{-5}$.

Then we normalize the iterative weights with adaptive broadcasting to pixel-wise level such that

$$\tilde{w}_{ij} = \frac{w^t_j}{\sum_{j \in C}(w^t_j)} * (1 + y_{ij} + \tilde{y}_{ij}) \, , \tag{9}$$

where $\tilde{y}_{ij} \in (0, 1)$ and $y_{ij} \in \{0, 1\}$ denote the ij-th prediction and the ground-truth of class j separately in the current training samples.

In addition, instead of using traditional cross-entropy function which focuses on positive samples, we introduce a

Layers	**Outputs**	**Sizes**
CNN	X	$\frac{h}{16} \times \frac{w}{16} \times 1024$
Augment	(X, X_{90}, X_{180})	$3 \times (\frac{h}{16} \times \frac{w}{16} \times 1024)$
SCG	$(\hat{A}, X', \hat{\boldsymbol{y}}), (\hat{A}_{90}, X'_{90}, \hat{\boldsymbol{y}}_{90}), (\hat{A}_{180}, X'_{180}, \hat{\boldsymbol{y}}_{180})$	$3 \times [(n \times n), (n \times 1024), (n \times c)]$
GCN^1	$(Z^{(1)}, Z^{(1)}_{90}, Z^{(1)}_{180})$	$3 \times (n \times d)$
GCN^2	$(Z^{(2)}, Z^{(2)}_{90}, Z^{(2)}_{180})$	$3 \times (n \times c)$
Fusion	$(\hat{\boldsymbol{y}} + Z^{(2)}) \oplus (\hat{\boldsymbol{y}}_{90} + Z^{(2)}_{90})_{r90} \oplus (\hat{\boldsymbol{y}}_{180} + Z^{(2)}_{180})_{r180}$	$(n \times c) \longrightarrow (\frac{h}{16} \times \frac{w}{16} \times c)$
Projection	$\tilde{\boldsymbol{y}}$	$h \times w \times c$

Table 1. MSCG-Net Model Details with one sample of input image size of $h \times w \times 4$. Note: $\oplus$ denotes an element-wise addition, the index (i.e. 90, 180) indicates the rotated degree, while $r90$ and $r180$ denote the reversed rotation degrees.

positive and negative class balanced function (PNC) which is defined as

$$\boldsymbol{p} = \boldsymbol{e} - \log\left(\frac{1-\boldsymbol{e}}{1+\boldsymbol{e}}\right) , \tag{10}$$

where $\boldsymbol{e} = (\boldsymbol{y} - \tilde{\boldsymbol{y}})^2$.

Building on the dice coefficient [15] with our adaptive class weighting PNC function, we develop an adaptive multi-class weighting (ACW) loss function for multi-class segmentation tasks

$$\mathcal{L}_{acw} = \frac{1}{|Y|}\sum_{i \in Y}\sum_{j \in C} \tilde{w}_{ij} * p_{ij} - \log\left(\text{MEAN}\{d_j | j \in C\}\right) , \tag{11}$$

where Y contains all the labeled pixels and d_j is the dice coefficient given as

$$d_j = \frac{2\sum_{i \in Y} y_{ij}\tilde{y}_{ij}}{\sum_{i \in Y} y_{ij} + \sum_{i \in Y} \tilde{y}_{ij}} . \tag{12}$$

The overall cost function of our model, with a combination of two regularization terms $\mathcal{L}_{kl}$ and $\mathcal{L}_{dl}$ as defined in the equations 3 and 4, is therefore defined as

$$\mathcal{L} \leftarrow \mathcal{L}_{acw} + \mathcal{L}_{kl} + \mathcal{L}_{dl} . \tag{13}$$

3. Experiments and results

We first present the training details and report the results. We then conduct an ablation study to verify the effectiveness of our proposed methods.

3.1. Dataset and Evaluation

We train and evaluate our proposed method on the Agriculture-Vision challenge dataset, which is a subset of the Agriculture-vision dataset [2]. The challenge dataset consists of $21,061$ aerial farmland images captured throughout 2019 across the US. Each image contains four 512x512 color channels, which are RGB and Near Infra-red (NIR). Each image has a boundary map that indicates the region of the farmland, and a mask that indicates valid pixels in the image. Seven types of annotations are included: Background, Cloud shadow, Double plant, Planter skip, Standing Water, Waterway and Weed cluster. Models are evaluated on the validation set with $4,431$ NIR-RGB images segmentation pairs, while the final scores are reported on the test set with $3,729$ images. The mean Intersection-over-Union (mIoU) is used as the main quantitative evaluation metric. Due to the fact that some annotations may overlap in the dataset, for pixels with multiple labels, a prediction of either label will be counted as a correct pixel classification for that label.

3.2. Training details

We use backbone models pretrained on ImageNet in this work. We randomly sample patches of size 512×512 as input and train it using mini batches of size 10 for the MSCG-Net-50 model and size 7 for the MSCG-Net-101 model. The training data (containing 12901 images) is sampled uniformly and randomly flipped (with probability 0.5) for data augmentation and shuffled for each epoch.

According to our best practices, we first train the model using Adam [6] combined with Lookahead [17] as the optimizer for the first 10k iterations and then change the optimizer to SGD in the remaining iterations with weight decay 2×10^{-5} applied to all learnable parameters except biases and batch-norm parameters. We also set $2 \times LR$ to all bias parameters compared to weight parameters. Based on our training observations and empirical evaluations, we use initial LRs of $\frac{1.5 \times 10^{-4}}{\sqrt{3}}$ and $\frac{2.18 \times 10^{-4}}{\sqrt{3}}$ for MSCG-Net-50 and MSCG-Net-101 separately, and also apply cosine annealing scheduler that reduces the LR over epochs. All models are trained on a single NVIDIA GeForce GTX 1080Ti. It took roughly 10 hours to train our model for 25 epochs with batch size 10 over $12,901$ NIR-RGB training images.

3.3. Results

We evaluated and tested our trained models on the validation sets and the hold out test sets with just single feed-forward inference without any test time augmentation

Models	mIoU	Background	Cloud shadow	Double plant	Planter skip	Standing water	Waterway	Weed cluster	mIoU*
MSCG-Net-50	0.547	0.780	**0.507**	0.466	**0.343**	**0.688**	0.513	**0.530**	0.508
MSCG-Net-101	**0.550**	0.798	0.448	**0.550**	0.305	0.654	**0.592**	0.506	**0.509**
Ensemble-TTA	0.599	0.801	0.503	0.576	0.520	0.696	0.560	0.538	0.566

Table 2. mIoUs and class IoUs of our models on Agriculture-Vision test set. Note: mIoU is the mean IoU over all 7 classes while mIoU* is over 6-class without the background, and Ensemble-TTA denotes the two models ensemble (MSCG-Net-50 with MSCG-Net-101) combined with TTA methods [12].

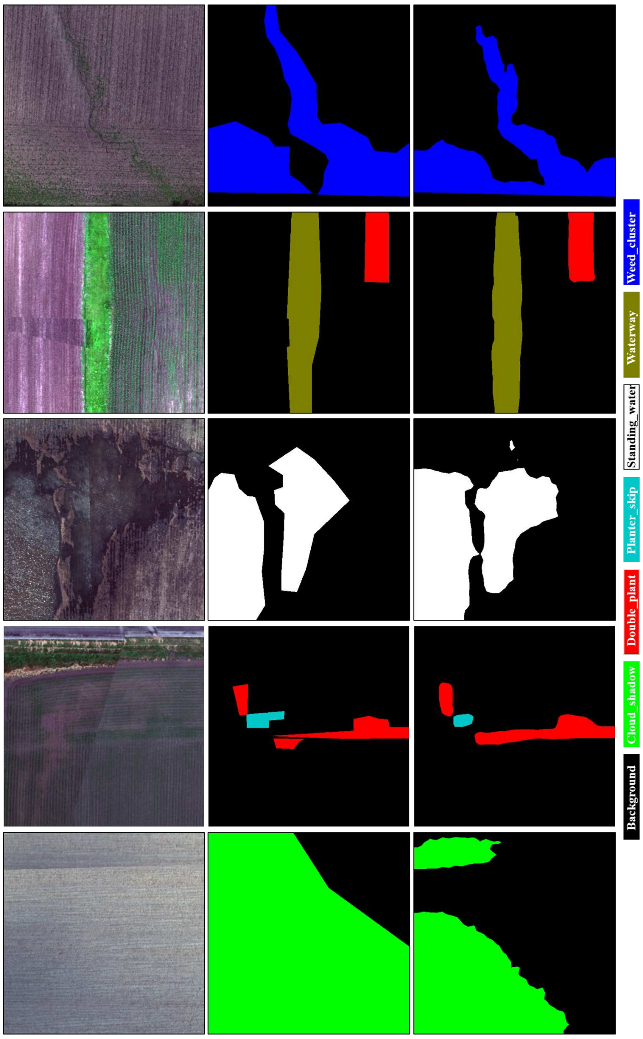


Figure 3. Segmentation results on validation data. From the left to right, the input images, the ground truths and the predictions of our trained models.

(TTA) or models ensemble. However, we do include results for a simple two-model ensemble (MSCG-Net-50 together with MSCG-Net-101) with TTA for completeness. The test results are shown in Table 2. Our MSCG-Net-50 model obtained very competitive performance with 0.547 mIoU with very small training parameters (9.59 million) and has low computational cost (18.21 Giga FLOPs with input size of $4 \times 512 \times 512$), resulting in fast training and inference performance on both CPU and GPU as shown in Table 3.3. A qualitative comparisons of the segmentation results from our trained models and the ground truths on the validation data are shown in Fig. 3.

3.4. Ablation studies

Effect of the Multi-view. To investigate how the multiple views help, we report the results of the single-view models and the multi-view models trained with both Dice loss and ACW loss in Table 4. Note that, for simplicity, we fixed the backbone encoder as Se_ResNext50 and other training parameters (e.g. learning rate, decay police, and so on.). Also, the mIoUs are computed on the validation set without considering multiple labels. The results suggest that multiple views could improve the overall performance from 0.456 to 0.516 (+6%) mIoU when using Dice loss, and from 0.472 to 0.527 (+5.5%) with the proposed ACW loss.

Effect of the ACW loss. As shown in Table 4, we note that for the single-view models, the overall performance can be improved from 0.456 to 0.472 (+1.6%) mIoU. For the multi-view models, the performance improved +1.1%, increasing from 0.516 to 0.527. Compared to the single-view model SCG-Net with Dice loss, which was proposed in [13] and achieved state-of-the-art performance on a commonly used segmentation benchmark dataset, our Multi-view MSCG-Net model with ACW loss achieved roughly +7.1% higher mIoU accuracy. We show some qualitative results in Fig. 4 that illustrate the proposed multi-view model and the adaptive class weighting method and show that they help to produce more accurate segmentation results for both larger and smaller classes.

4. Conclusions

In this paper, we presented a multi-view self-constructing graph convolutional network (MSCG-Net) to extend the SCG module which makes use of learnable latent variables to self-construct the underlying graphs, and to explicitly capture multi-view global context representations with rotation invariance in airborne images. We further developed a novel adaptive class weighting loss that alleviates the issue of class imbalance commonly found in semantic segmentation datasets. On the Agriculture-Vision

Models	Backbones	Parameters	FLOPs	Inference time
		(Million)	(Giga)	(ms - CPU/GPU)
MSCG-Net-50	Se_ResNext50	9.59	18.21	522 / 26
MSCG-Net-101	Se_ResNext101	30.99	37.86	752 / 45

Table 3. Quantitative Comparison of parameters size, FLOPs (measured on input image size of $4 \times 512 \times 512$), Inference time on CPU and GPU separately.

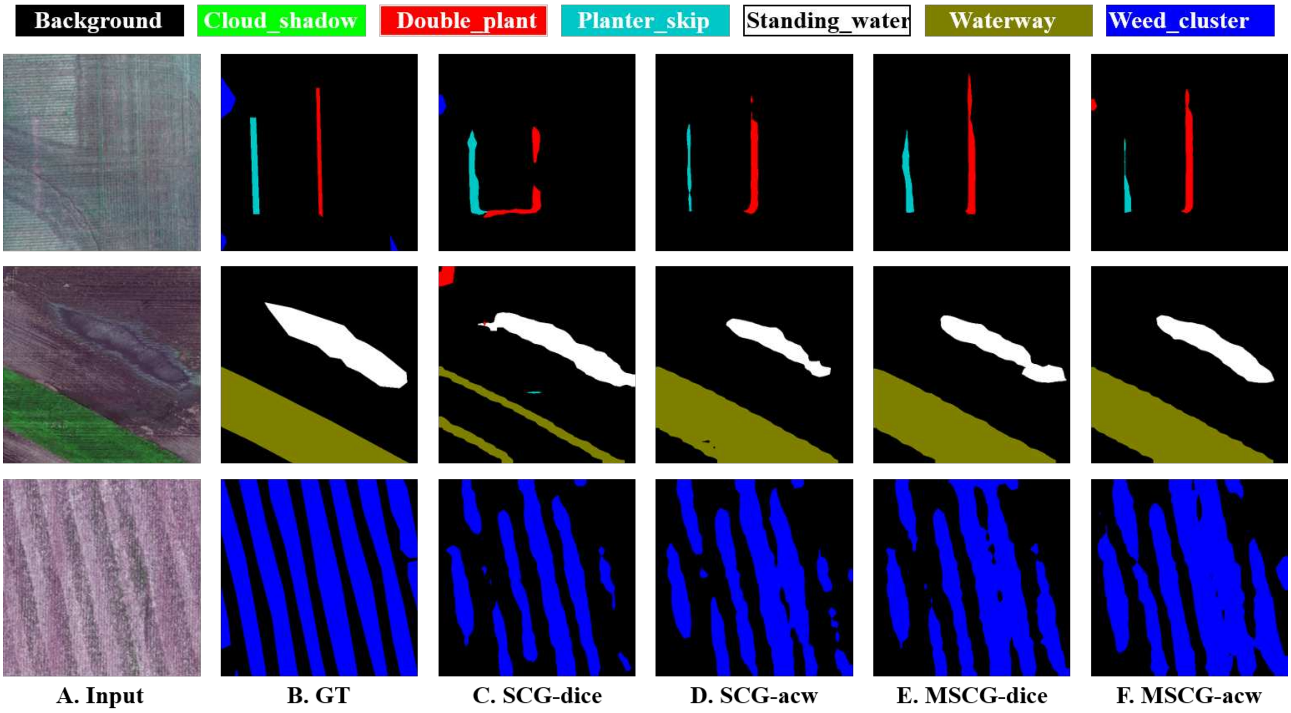


Figure 4. Segmentation results using different models. From the left to right, the input images, the ground truths and SCG-Net with dice loss, SCG-Net with ACW loss, MSCG-Net with dice loss, and MSCG-Net with ACW loss.

Models	Configurations			mIoU
	Multi-view	Dice loss	ACW loss	
SCG-dice		✓		0.456
SCG-acw			✓	0.472
MSCG-dice	✓	✓		0.516
MSCG-acw	✓		✓	0.527

Table 4. Ablation study of our proposed network. Note that, for simplicity, we fixed the learning high-parameters and the backbone encoder, and mIoU is evaluated on validation set without considering overlapped annotations.

challenge dataset, our MSCG-Net model achieves very robust and competitive results, while making use of fewer parameters and being computationally more efficient.

Acknowledgments

This work is supported by the foundation of the Research Council of Norway under Grant 220832.

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*, 34(4):18–42, 2017. 1

[2] Mang Tik Chiu, Xingqian Xu, Yunchao Wei, Zilong Huang, Alexander Schwing, Robert Brunner, Hrant Khachatrian, Hovnatan Karapetyan, Ivan Dozier, Greg Rose, David Wilson, Adrian Tudor, Naira Hovakimyan, Thomas S. Huang, and Honghui Shi. Agriculture-vision: A large aerial image database for agricultural pattern analysis, 2020. 2, 4

[3] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 3

[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*, pages 11487–11496, 2019. 1

[5] Michael Kampffmeyer, Arnt-Borre Salberg, and Robert Jenssen. Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2016. 3

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[7] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[8] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 1, 2

[9] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016. 2

[10] Boris Knyazev, Xiao Lin, Mohamed R Amer, and Graham W Taylor. Image classification with hierarchical multigraph networks. *arXiv preprint arXiv:1907.09000*, 2019. 1

[11] Xiaodan Liang, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P Xing. Symbolic graph reasoning meets convolutions. In *Advances in Neural Information Processing Systems*, pages 1853–1863, 2018. 1

[12] Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Borre Salberg. Dense dilated convolutions' merging network for land cover classification. *IEEE Transactions on Geoscience and Remote Sensing*, page 1–12, 2020. 5

[13] Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. Self-constructing graph convolutional networks for semantic labeling. *arXiv preprint arXiv:2003.06932*, 2020. 1, 2, 5

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 1

[15] 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)*, pages 565–571. IEEE, 2016. 4

[16] 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)*, 38(5):146, 2019. 1

[17] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9593–9604, 2019. 4

15

Paper IV

Multi-modal land cover mapping of remote sensing images using pyramid attention and gated fusion networks

Qinghui Liu, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg

Submitted to International Journal of Remote Sensing, September, 2021.

Multi-modal land cover mapping of remote sensing images using pyramid attention and gated fusion networks

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]UiT Machine Learning Group, Department of Physics and Technology, UiT the Arctic University of Norway, Tromsø, Norway

ARTICLE HISTORY
Compiled October 8, 2021

ABSTRACT

Multi-modality data is becoming readily available in remote sensing (RS) and can provide complementary information about the Earth's surface. Effective fusion of multi-modal information is thus important for various applications in RS, but also very challenging due to large domain differences, noise, and redundancies. There is a lack of effective and scalable fusion techniques for bridging multiple modality encoders and fully exploiting complementary information. To this end, we propose a new multi-modality network (MultiModNet) for land cover mapping of multi-modal remote sensing data based on a novel pyramid attention fusion (PAF) module and a gated fusion unit (GFU). The PAF module is designed to efficiently obtain rich fine-grained contextual representations from each modality with a built-in cross-level and cross-view attention fusion mechanism, and the GFU module utilizes a novel gating mechanism for early merging of features, thereby diminishing hidden redundancies and noise. This enables supplementary modalities to effectively extract the most valuable and complementary information for late feature fusion. Extensive experiments on two representative RS benchmark datasets demonstrate the effectiveness, robustness, and superiority of the MultiModNet for multi-modal land cover classification.

KEYWORDS
Multiple modalities, Pyramid attention, Gated fusion, Multi-modal segmentation, Remote sensing

1. Introduction

Automatic mapping of land cover using remote sensing (RS) data is of great importance for a wide range of earth observation applications since it provides a fast and cost-effective solution for analyzing large areas (Salberg 2011; Audebert, Le Saux, and Lefèvre 2016). This includes applications like urban planning (Noor, Abdullah, and Hashim 2018), precision agriculture (Chiu et al. 2020a; Liu et al. 2020b), and disaster management (Salberg, Rudjord, and Solberg 2014; Bello and Aina 2014; Fan et al. 2021), to name a few. In the past few years, the emergence of deep learning and con-

CONTACT Qinghui Liu. Email: samleoqh@gmail.com

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.

volutional neural networks (CNNs) has led to significant improvements for land cover mapping in RS (Maggiori et al. 2017; Audebert, Le Saux, and Lefèvre 2018; Pashaei et al. 2020; Liu et al. 2020). Many existing deep learning approaches, however, only use unimodal remote sensing images, e.g., the standard three-channel data such as RGB or IRRG (NIR-Red-Green) images. Multi-modality data is becoming readily available and increasingly essential in remote sensing. This raises open challenges such as "what," "how," and "where" to effectively fuse multi-modal data (Hong et al. 2020) in order to develop joint representations of multiple modalities for enhancing land cover mapping performance.

Remote sensing imagery is often characterized by complex data properties in the form of heterogeneity and class imbalance, as well as overlapping class-conditional distributions that bring severe challenges for generating land cover maps or detecting and localizing objects, producing a high degree of uncertainty in obtained results. As shown in Fig. 1, mismapped or mislabeled results appear in the unimodal case for objects with similar color and texture, e.g., the roof of buildings vs. surfaces, and, the trees vs. low vegetation in Vaihingen dataset. On the other hand, our proposed multi-modal learning-based method alleviates these problems.

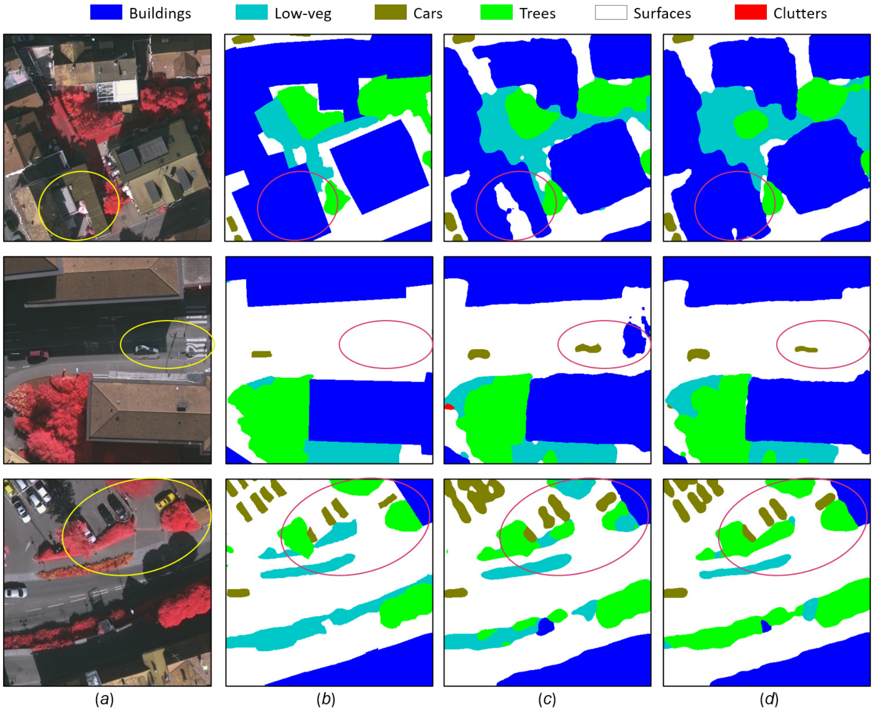


Figure 1. Mismapped or mislabeled examples in the Vaihingen dataset. (a) the IRRG images, (b) the labels, (c) the mapping results from a unimodal (only IRRG) model, and (d) the mapping results from our multi-modal (IRRG + DSM) model.

In order to improve the performance of semantic mapping that can be obtained from a single modality (e.g., RGB or IRRG), additional modalities, either from the same sensor (e.g. multi-spectral or hyperspectral images) or from a different one (e.g., LiDAR point cloud data or SAR) are increasingly used for land cover mapping (Hazirbas et al. 2016; Xu et al. 2017; Audebert, Le Saux, and Lefèvre 2018). Examples include Synthetic Aperture Radar (SAR) images (Hong et al. 2020; Li et al. 2020), hyperspectral imagery (HSI) (Xu et al. 2017; Audebert, Le Saux, and Lefèvre 2019) and Digital

Surface Models (DSM) (Hazirbas et al. 2016; Audebert, Le Saux, and Lefèvre 2018). Multi-modal data has been proven to provide rich complementary information to deal with complex scenes as different imaging technologies in RS are capable of capturing a variety of properties from the earth's surface, such as height information, spectral radiance, and reflectance (Gómez-Chova et al. 2015).

One of the main challenges in the utilization of multi-modal data is how to effectively extract and fuse multi-modal features. Although deep learning-based methods can automatically learn representative features, multi-modal inputs and features often provide unequal, redundant, or even contradictory information. Current multi-modal models tend to extract features independently using two separate encoders, combining feature maps indiscriminately at early and/or late layers via concatenation or summation (Couprie et al. 2013; Audebert, Le Saux, and Lefèvre 2018). We argue that this design leads to both inaccurate and computationally inefficient models. In particular, it brings high sensitivity to missing or noisy data (Audebert, Le Saux, and Lefèvre 2018), which has a significant negative influence on overall model performance when dealing with missing or noisy modality scenarios (Kampffmeyer, Salberg, and Jenssen 2018). Another challenge of pixel-wise classification of multi-modal images is the increased model size and computational burden (Marmanis et al. 2016; Audebert, Le Saux, and Lefèvre 2018) that also limit the application in most scenarios with real-time requirements. Hence, the effective and efficient fusion of multi-modal information is still an open research direction and also needs to be further optimized for scalability and real-time consideration for real-world applications.

Recently, the usage of attention mechanisms and graph-based approaches has led to promising performance and computational efficiency gains on a range of different computer vision tasks (Mou and Zhu 2019; Fu et al. 2019; Liu et al. 2020b). These works, typically use these mechanisms to emphasize salient features and suppress irrelevant signals in unimodal settings. Further, they tend to ignore multi-scale information by only leveraging same-dimensional representations of the same scale (e.g., typically low-spatial-resolution feature spaces) in order to alleviate the computational cost. To facilitate an efficient multi-scale (pyramid) attention feature extraction from each modality, we propose a pyramid attention fusion (PAF) module for extracting multiple hierarchical-scale representations. By using a novel gated fusion unit (GFU) to blend complementary features between multi-modal encoders, we introduce a lightweight multi-modal segmentation network (MultiModNet). For more details, please refer to Section 3.

Our experiments demonstrate that the network achieves robust and accurate results on the representative ISPRS Semantic Labeling Contest Vaihingen dataset (Rottensteiner et al. 2012) and the Agriculture-Vision challenge dataset (Chiu et al. 2020b). The main contributions of this paper are as follows:

(1) We present a novel pyramid attention and gated fusion mechanism for multi-modality data that builds on our proposed gated fusion unit (GFU) and our pyramid attention fusion (PAF) module. It facilitates interactions between the encoders of each modality to effectively combine the extracted features from multiple modalities and weaken the influence of noise and redundancies among the multi-modal data.

(2) The proposed PAF module is a lightweight network with a built-in cross-hierarchical-scale and cross-view attention fusion mechanism that can obtain rich and robust contextual representations. It can be used as a stand-alone decoder for a unimodal model to improve segmentation performance, or as a vital

fusion mechanism to merge several modalities when combined with our gated fusion unit.

(3) Built upon the PAF and GFU modules, our end-to-end multi-modal segmentation model (MultiModNet) achieves state-of-the-art performance and outperforms the baselines on two representative remote sensing datasets with considerably fewer parameters and at a lower computational cost. We also validate the effectiveness and flexibility of our framework through extensive ablation studies.

The paper is structured as follows. Section 2 provides an overview of the related work. In Section 3, we present the methodology in detail. Experimental procedure and evaluation of the proposed method is performed in Section 4. Section 5 further discusses and evaluates our method via ablation studies. Finally, we draw conclusions and outline future research directions in Section 6.

2. Related work

The state-of-the-art deep learning-based segmentation models are mostly inspired by the idea of fully convolutional networks (FCNs) (Long, Shelhamer, and Darrell 2015). FCN models generally consist of an encoder-decoder architecture in which all layers are based on convolutions (and upsampling/downsampling operations). However, vanilla FCNs tend to cause a loss of spatial information due to the presence of pooling layers that reduce the resolution of feature maps by sacrificing the positional information of objects. The UNet (Ronneberger, Fischer, and Brox 2015) extends the FCN by introducing symmetric skip connections (i.e., concatenations) between the encoder and decoder modules to maintain spatial information. The precise spatial information can be gradually recovered in the decoder module by combining multiple skipped connections with upsampling or de-convolution layers. Since then, the encoder-decoder architecture has been widely extended in recent works including, among others, pyramid scene parsing network (PSPNet) (Zhao et al. 2017), SegNet (Badrinarayanan, Kendall, and Cipolla 2017), DeepLabV3+ (Chen et al. 2018), Dual attention network (Fu et al. 2019), and HRNet-OCR (Yuan, Chen, and Wang 2020).

The FCN-based or UNet-based encoder-decoder architectures have also been widely adopted and applied to the ISPRS Semantic Labeling Contest (Paisitkriangkrai et al. 2015; Kampffmeyer, Salberg, and Jenssen 2016; 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. 2017; Kampffmeyer, Salberg, and Jenssen 2018; Liu et al. 2020), and the Agriculture-Vision benchmark dataset for automatic mapping of land pattern types (Chiu et al. 2020b; Liu et al. 2020b; Chiu et al. 2020a). In general, these architectures differ from each other in how they capture rich and global contextual information at multiple scales. For instance, the stacked UNet architecture is proposed by Ghosh et al. (2018) for land cover segmentation in remote sensing imagery, which merges high-resolution details and long range contextual information captured at low-resolution to generate segmentation maps. Further, Liu et al. (2020) introduced a dense dilated convolutions merging (DDCM) network that sequentially stacked the output of each layer with its input features before feeding it to the next layer to capture global and multi-scale contextual features.

Despite the aforementioned impressive progress on unimodal deep learning, deep learning has also been exploited for multi-modal data processing to obtain finer representations of different modalities. From the perspective of multi-modal fusion in remote sensing, a multi-modal deep learning model normally involves concatenation of extracted features from unimodal networks (e.g., a backbone network) and then learning a joint representation for classification or segmentation. A representative work proposed by Audebert, Le Saux, and Lefèvre (2018) investigated two fusion strategies, namely early and late fusion methods based on the FuseNet framework, using SegNet or ResNet to classify multi-modal remote sensing data (such as LiDAR and multispectral images). Specifically, one CNN-based encoder (e.g, VGG or ResNet) is used to extract the features from RGB or IRRG images while another encoder is exploited to extract the features from LiDAR data and other bands (e.g NDVI). Note that the LiDAR data has been rasterized in the image domain as a digital surface model (DSM) with normalization (nDSM). Early fusion concatenates the features after each convolutional block from both encoders, while later fusion merges the last feature maps from the two deep networks. The results show that late fusion improves the overall accuracy at the cost of less balanced predictions, while early fusion achieves better performance for all classes but inducing higher sensitivity to missing or noisy data. Indeed, such fusion techniques do require all modalities to be available to the classification during both training and testing. Kampffmeyer, Salberg, and Jenssen (2018) therefore presented a novel CNN architecture based on so-called hallucination networks for urban land cover classification that can replace missing data modalities in the test phase. This enables fusion capabilities even when data modalities are missing in testing. Lately, Feng et al. (2019) presented an adaptive approach to fuse HSI and LiDAR data, in which a two-stream CNN is used to extract LiDAR and HSI features separately. Then an adaptive method based on squeeze-and-excitation networks (Hu, Shen, and Sun 2018) is designed to combine the features with adaptive weights instead of simply concatenation. Xu, Du, and Zhang (2018) and Xu et al. (2019) further proposed a Fusion-FCN framework for the classification of multi-source remote sensing data using fused FCNs where three different types of data (LiDAR data, hyperspectral images, and very high-resolution RGB images) are utilized in one model.

Recently, the usage of attention mechanisms in deep learning models has been increasingly explored in various visual inference tasks and has shown very promising performance gain (Fu et al. 2019; Mou and Zhu 2019; Mohla et al. 2020). Generally, the attention modules highlight the prominent features while suppressing the irrelevant features through a self-attention learning method (Vaswani et al. 2017). Recent work by Liu et al. (2020b) proposed a multi-view graph-based attention paradigm (MSCG) that demonstrated significant performance gain in contrast to a single-view attention module (SCG) (Liu et al. 2020a) for land cover mapping of multi-spectral aerial images. However, in most of these works, the attention modules are carried out only on single-level features with coarse resolution from a single modality to alleviate the computational cost. This brings challenges when attempting to accurately classify relatively small objects in very high-resolution remote sensing data. To alleviate these problems, our focus in this paper is mainly on land cover mapping (pixel-wise classification) tasks of multi-modal remote sensing images, facilitated through our proposed multi-scale and cross-view attention fusing mechanism.

3. Method

The proposed multi-modality network (MultiModNet) consists of four key modules: a backbone encoder (ENC), a pyramid attention fusion (PAF) module, a gated fusion unit (GFU), and a decoder (DEC) that produces the final output. Given a primary

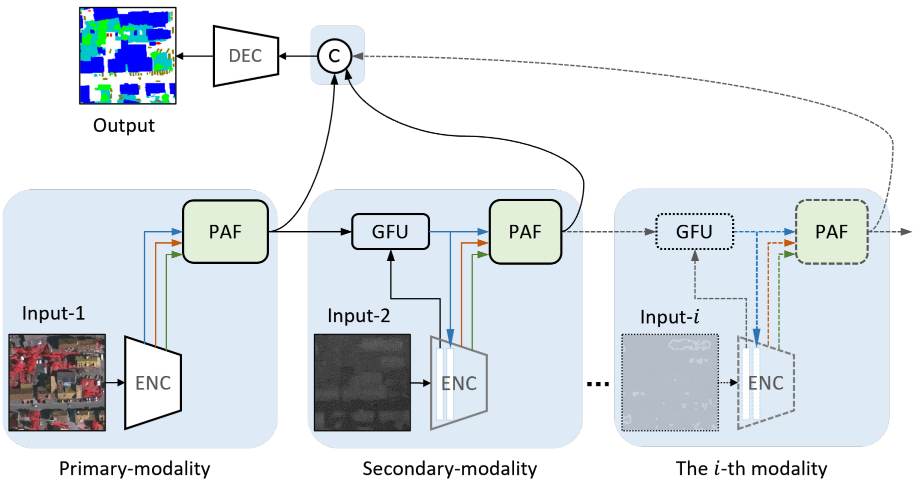


Figure 2. General concept structure of our multi-modality network (MultiModNet) based on proposed pyramid attention and gated fusion methods. Here ENC denotes the feature encoder, GFU accounts for a gated fusion unit, PAF is our proposed pyramid attention fusion module, ⓒ denotes concatenation, Input-1, Input-2 and Input-i are the primary, the secondary and the i-th modalities respectively, and, DEC is the decoder layer to output the final prediction. Note that our PAF module normally takes three different scale (i.e. 3-level) features of each modality as the input shown with blue, orange and green line.

and a secondary modality[1] Input-1 and Input-2, respectively, where Input-1, e.g. IRRG or RGB images, contains more valuable information than Input-2 , e.g. DSM or NIR images, off-the-shelf encoders (ENC), such as multi-layer CNN based backbones (e.g. ResNet), are used to extract multi-level feature maps for each modality. Then we utilize PAF modules (Section 3.1) to generate fine-grained cross-level features and GFUs (Section 3.2) to merge complementary features from the primary modality into the secondary modality. Finally, we concatenate all PAF generated features from each modality and feed them into a simple decoder (DEC) module, which in this work is composed of only a single convolution layer and a bi-linear interpolation function, to output the pixel-wise classification maps. As shown in Fig. 2, our MultiModNet framework has a scalable structure that allows it to easily extend to more than two modalities. The parts that follow will go through our PAF and GFU modules in detail.

3.1. *Pyramid Attention Fusion*

We develop the lightweight PAF module with a built-in cross-hierarchical-scale and cross-view attention fusion mechanism that can obtain rich and robust representations. The features produced from the PAF module at each previous modality will be integrated into the encoder layer of its successor modality through a GFU module. The proposed PAF module thus plays a vital role in fusing a range of modalities in a compact yet effective manner, while it can still be used as a stand-alone decoding layer for unimodal models to improve segmentation performance.

As illustrated in Fig. 3, our PAF module contains three key sub-blocks: the pyramid cross-view encoder, the attention construction and updating block, and the feature fusion block:

[1] There will be a third or even more supplementation modalities, we thus describe them using the i-th modality as illustrated in Figure 2, and assume they are ordered depending on informational richness and significance, i.e., Input-1 $\geq$ Input-2 $\geq \cdots \geq$ Input-i. In other words, each preceding modality can be seen as a primary modality with respect to the following (succeeding, if any) ones.

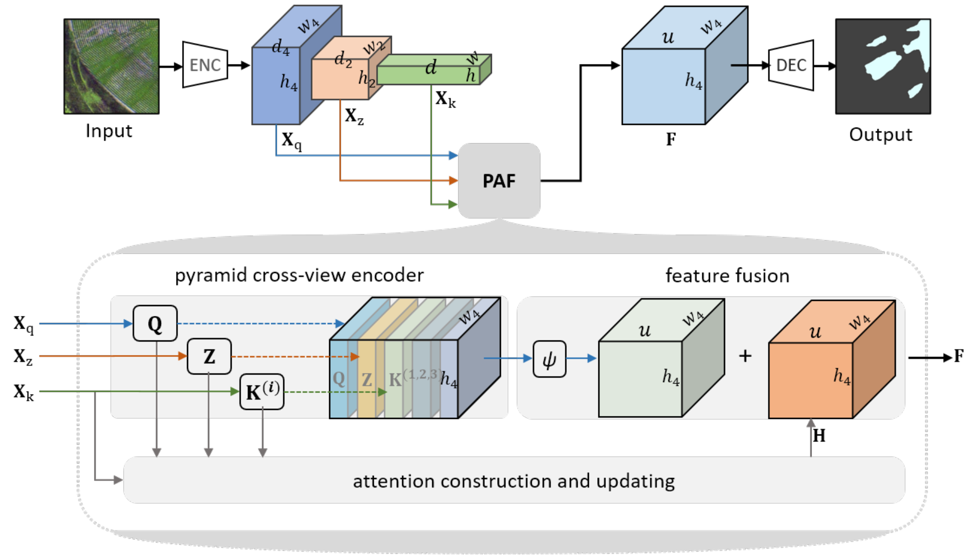


Figure 3. The illustration diagram of the pyramid attention fusion (PAF) module. Overall PAF is composed of three key blocks, i.e., the *pyramid coss-view encoder* that transforms the input pyramid features (i.e., $\mathbf{X}_\mathrm{q}$, $\mathbf{X}_\mathrm{z}$ and $\mathbf{X}_\mathrm{k}$, obtained by the ENC module) to corresponding multi-scale latent spaces (i.e., $\mathbf{Q}$, $\mathbf{Z}$ and $\mathbf{K}^{(i)}$), the *attention construction and updating* block that constructs the cross-view attention matrix and then transforms the high-level features onto high-resolution 2D attention features ($\mathbf{H}$) by a message-passing function (see Eq. 3), and the *feature fusion* unit, which combines the latent multi-scale features with a CNN network (ψ) and sums the learned attention features ($\mathbf{H}$) to eventually generate the fused feature map ($\mathbf{F}$), which can then be fed into the DEC module to produce the final output.

- The *pyramid cross-view encoder* transforms the selected three-different-size feature maps (e.g., $\mathbf{X}_\mathrm{q}$, $\mathbf{X}_\mathrm{z}$, $\mathbf{X}_\mathrm{k}$) to corresponding cross-level and cross-view latent representations (i.e., $\mathbf{Q}$, $\mathbf{Z}$ and $\mathbf{K}^{(i)}$), in order to decrease computational burden while extracting salient latent features for late cross-view attention map generation.
- The *attention construction and updating* block constructs the cross-view and cross-level attention matrix (see Eq. 2) and then transforms the high-level features onto high-resolution 2D attention representations $\mathbf{H}$ by a message-passing function (see Eq. 3), in order to obtain robust non-local and high-resolution contextual features.
- The *feature fusion* module combines the latent cross-level and cross-view features with a CNN network (denoted by ψ) and sums the learned high-resolution attention representation $\mathbf{H}$, in order to eventually produce the fine-grained contextual features $\mathbf{F}$, which can then be fed into the DEC module to produce the final output.

We describe each component of the framework in detail as follows.

3.1.1. Pyramid cross-view encoder

To reduce computational cost while obtaining robust latent feature representations for late constructing attention maps, we utilize a multi-view augmenting method (Liu et al. 2020b) in the pyramid cross-view encoder to explicitly exploit the rotation invariance in the deep features. We first define a view generation function $\mathbf{X}_\mathrm{k}^{(i)} = \tau(\mathbf{X}_\mathrm{k}, i)$, and a view reversion function $\mathbf{X}_\mathrm{k} = \tau^{-1}(\mathbf{X}_\mathrm{k}^{(i)}, i)$ for three different views ($i \in \{1, 2, 3\}$).

We let $\mathbf{X}_\mathrm{k}^{(1)} = \mathbf{X}_\mathrm{k}$, and generate $\mathbf{X}_\mathrm{k}^{(2)}$ and $\mathbf{X}_\mathrm{k}^{(3)}$ by transposing and vertically flipping, respectively. Then the module learns pyramid-level and cross-view latent representations, i.e, a low-level feature matrix $\mathbf{Q} \in \mathbb{R}^{h_4 \times w_4 \times c}$, a middle-level latent matrix $\mathbf{Z} \in \mathbb{R}^{h_2 \times w_2 \times c}$ and the high-level 3-view matrix $\mathbf{K}^{(i)} \in \mathbb{R}^{h \times w \times c}$ from the multi-scale features $\mathbf{X}_\mathrm{q} \in \mathbb{R}^{h_4 \times w_4 \times d_4}$, $\mathbf{X}_\mathrm{z} \in \mathbb{R}^{h_2 \times w_2 \times d_2}$, and $\mathbf{X}_\mathrm{k} \in \mathbb{R}^{h \times w \times d}$, respectively, using CNNs, i.e.,

$$\mathbf{Q} = \varphi\left(\mathbf{X}_\mathrm{q}; \boldsymbol{\theta}_\mathrm{q}\right), \quad \mathbf{Z} = \varphi\left(\mathbf{X}_\mathrm{z}; \boldsymbol{\theta}_\mathrm{z}\right), \quad \text{and} \quad \mathbf{K}^{(i)} = \tau^{-1}\left(\varphi\left(\mathbf{X}_\mathrm{k}^{(i)}; \boldsymbol{\theta}_\mathrm{k}\right), i\right), \tag{1}$$

where φ denotes the convolution layers with parameter kernels of $\boldsymbol{\theta}_\mathrm{q} \in \mathbb{R}^{d_4 \times 3 \times 3 \times c}$, $\boldsymbol{\theta}_\mathrm{z} \in \mathbb{R}^{d_2 \times 3 \times 3 \times c}$, and $\boldsymbol{\theta}_\mathrm{k} \in \mathbb{R}^{d \times 3 \times 3 \times c}$ respectively. Note that d_4, d_2, and d represent the input feature dimensions of $\mathbf{X}_\mathrm{q}$, $\mathbf{X}_\mathrm{z}$ and $\mathbf{X}_\mathrm{k}$ respectively, c is the output feature dimension, and typically $c < d_4 < d_2 < d$. Here, $h_4 \times w_4$, $h_2 \times w_2$ and $h \times w$ denote the spatial sizes of both the input and the output feature maps, and commonly $h_4 = 2h_2 = 4h$, $w_4 = 2w_2 = 4w$. We also use zero-padding methods in CNN layers of the module to keep the output spatial resolution the same as the input.

3.1.2. Attention construction and updating

To obtain a robust non-local and high-resolution contextual feature space based on these learned three-level and three-view latent representations (i.e., $\mathbf{Q}$, $\mathbf{Z}$, $\mathbf{K}^{(i)} : i = 1, 2, 3.$), we propose a novel attention construction and updating module that can efficiently model long-range and cross-level pixel-wise dependencies and effectively produce rich non-local and high-resolution contextual representations via an upsampling-based attention-passing mechanism. This module is formed of two key components: attention construction and attention-passing. They are described in detail as follows.

Attention construction. Inspired by the success of self-attention (Vaswani et al. 2017) to encode the structural information of a sequence of data, we present a long-range cross-level attention method that uses latent feature similarity to model the interactions between every pair of pixels in cross-level feature maps. Furthermore, we introduce a multi-view fusion strategy in the attention module, which allows us to encode cross-level as well as cross-view pixel-wise dependencies to improve its robustness. Specifically, we first reshape the low-level high-resolution latent matrix $\mathbf{Q}$ to $\hat{\mathbf{Q}} \in \mathbb{R}^{(h_4 w_4) \times c}$, the middle-level latent matrix $\mathbf{Z}$ to $\hat{\mathbf{Z}} \in \mathbb{R}^{(h_2 w_2) \times c}$ and the high-level view matrices $\mathbf{K}^{(i)}$ to $\hat{\mathbf{K}}^{(i)} \in \mathbb{R}^{(hw) \times c}$. Then our cross-view and cross-level attention construction function is defined as

$$\mathbf{A} = \mathrm{norm}\left(\overbrace{\sum_{i=1}^{3}}^{\substack{\text{cross-view}\\ \text{fusion}}} w_i \ \mathrm{ReLU}\left(\overbrace{\hat{\mathbf{Q}}\left(\underbrace{\tanh\left(\hat{\mathbf{Z}}^\top \hat{\mathbf{Z}}\right) + \mathbf{I}_\alpha}_{\text{channel-wise attention}}\right)\hat{\mathbf{K}}^{\mathrm{T}(i)}}^{\text{long-range cross-level attention}}\right)\right) \in \mathbb{R}^{(h_4 w_4) \times (hw)} \tag{2}$$

where w_i is a learnable parameter initialized as 1 for our attention construction function, $\tanh(\cdot)$ and $\mathrm{ReLU}(\cdot)$ denote the tanh and ReLU non-linear functions respectively, and $\mathbf{I}_\alpha \in \mathbb{R}^{c \times c}$ is a learnable bias kernel initialized as $\mathbf{I}$. Note that the attention matrix $\mathbf{A}$ is constructed from features from different scales, resulting in long-range cross-level attention. The matrix is therefore a tall matrix, i.e. it has more rows (e.g., $16hw$) than columns (e.g., hw), and it is further normalized, i.e. norm($\cdot$), along rows by dividing by the sum of each row, so that the elements of each row vector in the matrix add up to 1. Figure 4 illustrates the attention matrix constructing process.

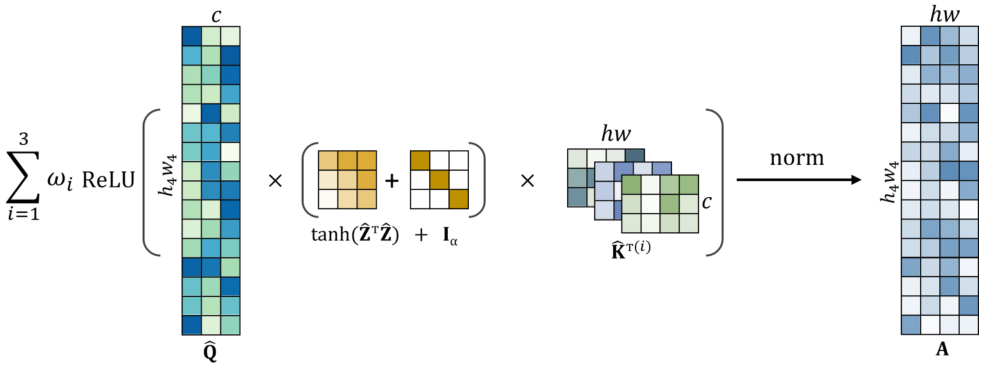

$\sum_{i=1}^{3} \omega_i$ ReLU
c
h_4w_4
$\hat{\mathbf{Q}}$
×
$\tanh(\hat{\mathbf{Z}}^{\top}\hat{\mathbf{Z}})$ + $\mathbf{I}_{\alpha}$
×
hw
c
$\hat{\mathbf{K}}^{\top(i)}$
norm
hw
h_4w_4
$\mathbf{A}$

Figure 4. The illustration of attention construction

Note that our attention construction process differs from the self-attention scheme in three major ways, i.e.,

- *Cross-level attention:* Our cross-level attention scheme utilizes three distinct level feature maps as the sources of multi-scale latent representations to efficiently generate a tall non-local interaction matrix instead of a square self-attention matrix. This allows our attention module to learn high-resolution features from low-resolution but high-abstract feature space. Based on our observations, using our cross-level attention to capture contextual information leads to faster training and better performance on remote sensing data than using self-attention methods based on one-scale low-resolution features or image patches (Dosovitskiy et al. 2021).
- *Channel-wise attention:* We also integrate a channel-wise attention method, i.e., $\tanh\left(\hat{\mathbf{Z}}^{\top}\hat{\mathbf{Z}}\right) + \mathbf{I}_{\alpha}$, into our long-range cross-level attention scheme (see Eq. 2) to improve feature discriminability by blending channel-wise weights learned from the middle-level feature ($\mathbf{Z}$) space. We observe that this results in better training stability and less sensitivity to latent feature dimensionalities (i.e., c) when compared to not using the channel-wise attention mechanism. We think that the channel-wise attention, like the dual attention network (Fu et al. 2019), could enhance our long-range attention mechanism by merging both channel and spatial attention attributes to capture robust cross-level information.
- *Cross-view fusion:* Furthermore, we introduce a cross-view fusion strategy into our attention module, inspired by our previous work (Liu et al. 2020b), to explicitly encode the rotation invariance in the high-abstract and deep-level latent features (i.e., $\mathbf{K}$). We fuse (add up) three-view long-range attention maps using learnable weights (i.e., w_i in Eq. 2) to further improve the model's robustness. Base on our experiments, using cross-view attentions can further speed up the model's learning process and result in better performance than using single-view attention maps.

Attention-passing. To produce a non-local but high-resolution feature representation (i.e., $\mathbf{H}$: typically 4 times the size of $\mathbf{X}_{\mathrm{k}}$) from the high-level but low-resolution features $\mathbf{X}_{\mathrm{k}}$, we develop an upsampling-based attention-passing function $f(\cdot)$ (Eq. 3). It is parameterized by the normalized attention matrix $\mathbf{A}$ and $\mathbf{X}_\mathrm{k}$ with trainable parameters $\mathbf{W} \in \mathbb{R}^{d\times u}$ where u denotes the output feature dimension. Our attention-passing mechanism, i.e., $\mathbf{A}\hat{\mathbf{X}}_\mathrm{k}\mathbf{W}$, is similar to the one-hop neighborhood message-passing function of graph convolutional networks (Kipf and Welling 2016) when viewing our learned tall attention matrix as a special type of adjacency matrix. Note that $\hat{\mathbf{X}}_\mathrm{k} \in \mathbb{R}^{(hw)\times d}$ is obtained by reshaping of $\mathbf{X}_\mathrm{k}$.

$$\mathbf{H} = f\left(\mathbf{X}_\mathrm{k} \in \mathbb{R}^{h\times w\times d}; \mathbf{A}, \mathbf{W}\right) = \delta\left(\mathbf{A}\hat{\mathbf{X}}_\mathrm{k}\mathbf{W}\right) \in \mathbb{R}^{h_4\times w_4\times u} . \tag{3}$$

With a combination operator, denoted by $\delta(\cdot)$ in Eq. 3, of non-linear activation function (e,g. ReLU) with bath normalization and reshaping, we eventually obtain a high-resolution attention representation $\mathbf{H} \in \mathbb{R}^{h_4\times w_4\times u}$ as illustrated in Figure 5.

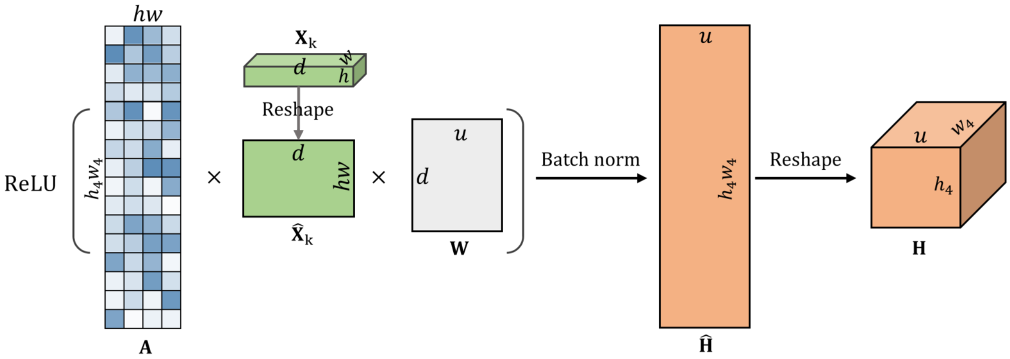


Figure 5. The illustration of attention-passing pipeline.

3.1.3. Feature fusion

Finally, we fuse the high-resolution attention features with cross-level and cross-view latent features in order to produce fine-grained high-resolution representations with robust non-local contextual and spatial information as the output using

$$\mathbf{F} = \psi\left(\mathbf{Q} \parallel \tilde{\mathbf{Z}} \parallel \tilde{\mathbf{K}}^{(1)} \parallel \tilde{\mathbf{K}}^{(2)} \parallel \tilde{\mathbf{K}}^{(3)}; \boldsymbol{\theta}_\psi\right) + \mathbf{H} , \tag{4}$$

where ψ denotes the convolution layer with parameter kernels of $\boldsymbol{\theta}_\psi \in \mathbb{R}^{5c\times 3\times 3\times u}$, batch normalization and non-linearity, and $\parallel$ denotes concatenation. Please note that middle-level latent and high-level view feature matrices ($\mathbf{Z}$ and $\mathbf{K}^{(i)}$) are up-sampled using bi-linear interpolation to $\tilde{\mathbf{Z}}$ and $\tilde{\mathbf{K}}^{(i)}$ in order to match the dimension of the high-resolution (i.e. $h_4 \times w_4$) feature matrix $\mathbf{Q}$ for concatenating. This is similar to the multi-level feature fusion method of pyramid feature networks (FPNs) (Lin et al. 2017), which fuse multi-level features from the top-down path by upsampling and summing, but instead of summation, we use concatenation and convolution operations to merge multi-level feature maps for remote sensing data.

3.2. *Gated Fusion Unit*

The GFU module is designed to serve as a fusion gateway between the main and secondary modalities. It utilizes a novel gating mechanism to allow the primary modality to aid its secondary modality in extracting the supplementary information via a gating network, thereby minimizing the influence of hidden noise and redundancies. Specifically, the GFU module is composed of two CNN layers with two gating operations (element-wise multiplications) as shown in Fig. 6. The first gate operation helps to weaken redundancies and capture salient useful features from the secondary modality, while the second gate operation aims to obtain complementary features from the primary modality and merge them into the secondary modality. The operation of the GFU module can be summarized by the following mathematical equations:

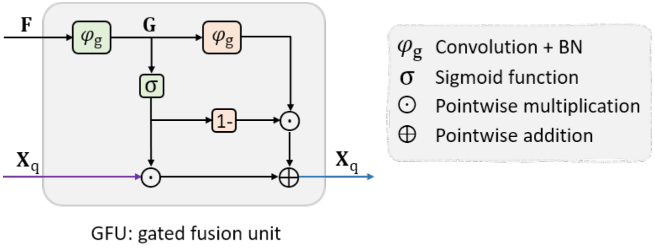


Figure 6. The gated fusion unit (GFU) consists of 2 CNN layers (φ_g) with batch normalization, and an activation function (σ, i.e, Sigmoid), where '1-' denotes one minus the input activation maps.

$$\mathbf{G} = \varphi_{\mathrm{g}}\left(\mathbf{F}; \boldsymbol{\theta}_{\mathrm{s}}\right), \qquad \mathbf{X}_{\mathrm{q}} = \sigma\left(\mathbf{G}\right) \odot \mathbf{X}_{\mathrm{q}} + \left(1 - \sigma\left(\mathbf{G}\right)\right) \odot \varphi_{\mathrm{g}}\left(\mathbf{G}; \boldsymbol{\theta}_{\mathrm{r}}\right)\ , \tag{5}$$

where $\mathbf{F}$ represents the fused representations by the PAF module of the primary modality and $\mathbf{X}_{\mathrm{q}}$ denotes the low-level features extracted by the encoder of the secondary modality. Here φ_{g} represents the convolution layers with 1×1 filters $\boldsymbol{\theta}_{\mathrm{s}}$ and $\boldsymbol{\theta}_{\mathrm{r}}$ respectively, and combine a batch normalization operator. σ is a sigmoid activation function. Note that the updated $\mathbf{X}_{\mathrm{q}}$ (the output of GFU) will feed the remaining layers of the encoder and also serve as one of the three input feature maps to the PAF module.

4. Data, experiments and results

4.1. *Benchmark datasets*

In this paper, we focus on two different representative databases, namely the ISPRS Vaihingen 2D dataset (Rottensteiner et al. 2012) and the Agriculture-Vision[2] challenge dataset (Chiu et al. 2020b). The ISPRS Vaihingen 2D dataset[3] is comprised of aerial remote sensing images over the city Vaihingen in Germany. The Agriculture-Vision dataset consists of large-scale high-quality aerial images from 3, 432 farmlands across the US and has been annotated with nine types of field anomaly patterns that are most important to farmers. Each dataset provides online leaderboards and reports test metrics measured on hold-out test images.

[2] https://www.agriculture-vision.com/agriculture-vision-2021/dataset-2021

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

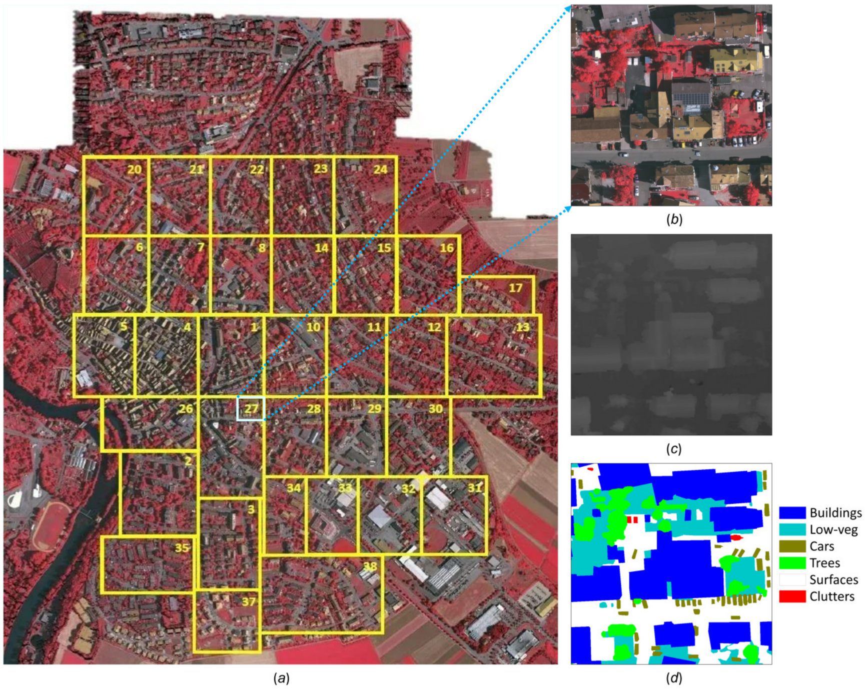


Figure 7. Overview of the ISPRS Vaihingen 2D semantic labeling benchmark dataset that contains 33 tiles: (a) overview of the entire dataset (the ID number labeled in the upper right corner of each area), (b) the IRRG image patch, (c) the DSM, (d) the ground truth.

4.1.1. Vaihingen dataset

The Vaihingen dataset is composed of 33 orthorectified image tiles acquired by a near-infrared (NIR) - red (R) - green (G) aerial camera and has been labeled with six common land cover categories: impervious surfaces (i.e., roads and concrete surfaces), buildings, low vegetation, trees, cars and clutter (representing uncategorizable land covers). 16 out of the 33 tiles are fully annotated at pixel level as the training set, and 17 tiles (i.e., areas: 2, 4, 6, 8, 10, 12, 14, 16, 20, 22, 24, 27, 29, 31, 33, 35 and 38) are used as hold-out test images as shown in Fig. 7. The average size of the tiles is approximately 2500 × 2000 pixels with a ground resolution of 9cm.

Images are accompanied by a digital surface model (DSM) that is derived from dense image matching techniques and represents the absolute height of pixels. Normalized DSM (nDSM) data are also included, which represent the pixels heights relative to the elevation of the nearest ground surface. We use both IRRG and nDSM data for training and test. Fig. 7 shows some examples of the dataset.

4.1.2. Agriculture-Vision dataset

The Agriculture-Vision dataset consists of 94, 986 aerial farmland images, of which 19, 708 images are used as the hold out test set, and 18, 334 are used as the local validation set. Each image consists of 512 × 512 RGB and NIR channels with resolution as high as 10 cm per pixel. Nine types of the most important field patterns are annotated: double plant, drydown, endrow, nutrient deficiency, planter skip, storm damage, water, waterway, and weed cluster. In addition, each image has a boundary map that indicates the region of the farmland, and a mask that indicates valid pixels in the image. Fig. 8 shows some examples of the dataset (note that the black regions

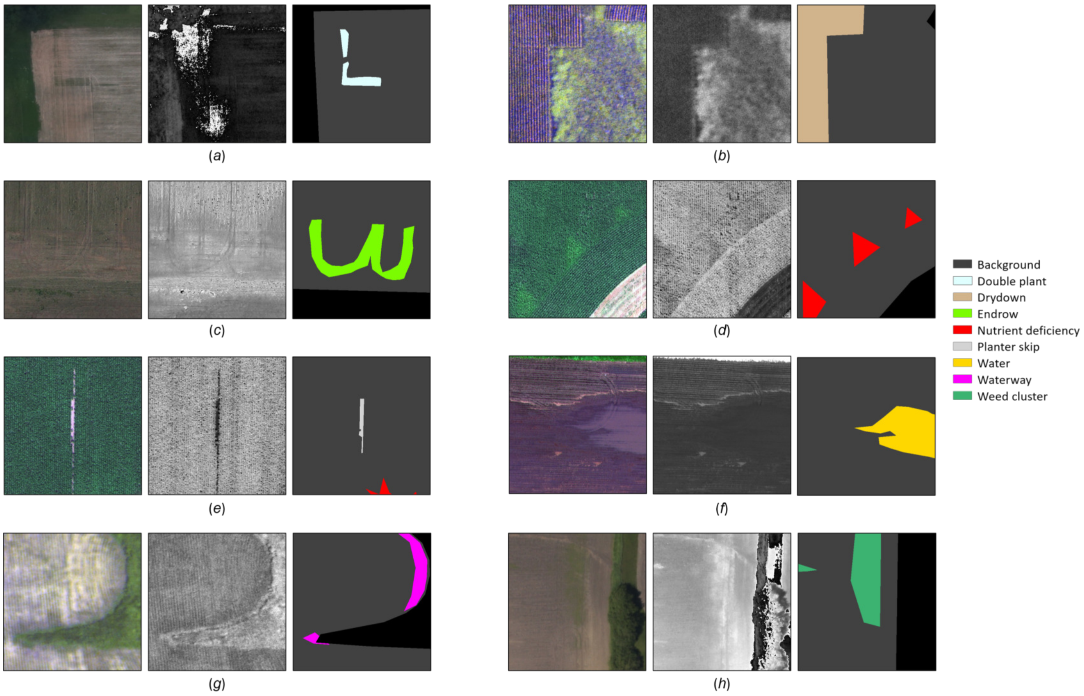

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 8. Some image examples of the agriculture-vision dataset: (a) the double plant patches (left: RGB image, middle: NIR image, right: the ground truth), (b) the drybown patches, (c) the endrow patches, (d) the nutrient deficiency patches, (e) the planter skip patches, (f) the water patches, (g) the waterway patches, (h) the weed cluster patches.

in the ground truth denote invalid areas).

Due to the fact that some annotations may overlap in the dataset, for pixels with multiple labels, a prediction of either label will be counted as a correct pixel classification for that label. Therefore, the conventional mean Intersection-over-Union (mIoU) metric is modified accordingly by categorizing predictions of any label in a pixel as a correct prediction. This customized mIoU is used as the main quantitative evaluation metric of the contest dataset (Chiu et al. 2020a).

4.2. *Variants of MultiModNet*

Built upon PAF, GFU, and the incorporation of different backbone encoders, we develop various MultiModNet models for land cover mapping tasks on different remote sensing data as shown in Table 1. Specifically, we use different backbone encoders (all are pretrained on ImageNet in this work) for Vaihingen and Agriculture-vision

Table 1. Detailed configurations of variants of MultiModNet with quantitative comparison of parameters size, FLOPs (measured on input image size of 4 × 512 × 512), Inference time on CPU and GPU separately.

Models	Inputs	ENC-1	ENC-2	PAF-1	PAF-2	GFU	Parameters (Million)	FLOPs (Giga)	Inference time (ms - CPU/GPU)
PAFNet[a]	DSM-IRRG (unimodal 4-band)	Se_ResNext50 (output: 256/512/1024)	✗	latent:6 output:24	✗	✗	9.52	18.12	513 / 24
PAGNet[a]	IRRG, DSM (Two-modal)	Se_ResNext50 (output: 256/512/1024)	MobileNetV3 (output: 40/112/960)	latent:6 output:24	latent:6 output:24	output:40	12.62	21.34	564 / 36
PAFNet[b]	NIR-RGB (unimodal 4-band)	MobileNetV3 (output: 40/112/960)	✗	latent:8 output:32	✗	✗	4.86	5.23	68 / 6
PAGNet[b]	RGB, NIR (Two-modal)	MobileNetV3 (output: 40/112/960)	MobileNetV3 (output: 40/112/960)	latent:8 output:32	latent:8 output:32	output:40	6.14	7.86	127 / 11

datasets. Our models, i.e., PAFNet^a and multi-modal PAGNet^a for the Vaihingen dataset, use the Se_ResNext50 (Hu, Shen, and Sun 2018) as the ENC-1 for IRRG images and the MobileNetV3 (Howard et al. 2019) as ENC-2 for DSM data respectively, while for the Agriculture-Vison dataset, the models i.e. PAFNet^b and PAGNet^b, we use two identical MobileNetV3 models as the encoders for both RGB and NIR data. We were not able to use Se_ResNext50 for the Agriculture-Vision dataset due to the memory limitation (11Gb) of our GPU, since Se_ResNext50 requires much more memory compared to MobileNetV3 when taking larger input size and batch size required for training models on the Agriculture-vision dataset.

4.3. *Training details*

According to best practices, we train all our models using Adam (Kingma and Ba 2014) as the optimizer for the first 10k iterations and then change the optimizer to SGD in the remaining iterations with weight decay 2×10^{-5} applied to all learnable parameters except biases and batch-norm parameters. We use a polynomial learning rate (lr) decay $(1 - \frac{cur\_iter}{max\_iter})^{0.9}$ with the maximum iterations set to 10^8. We also set $2 \times$ lr to all bias parameters in contrast to weights parameters. Based on our training observations to achieve fast and stable convergence, we apply the adaptive multi-class weighting loss ($\mathcal{L}_{acw}$) function (Liu et al. 2020b) for all our experiments.

Guided by our empirical results and our previous work (Liu et al. 2020; Liu et al. 2020b), we train and validate the networks for the Vaihingen dataset with 5000 randomly sampled patches of size 448×448 as input and a batch size of five. For the experiments on the Agriculture-Vison dataset, we randomly sample images of size 512×512 as input and train it using mini-batches of size 12. We conduct all experiments using PyTorch on a computer with a single GeForce GTX 2080Ti. For the Vaihingen dataset, we set the initial learning rate to 1.8×10^{-4} and utilized a stepwise learning-rate schedule method that reduces the learning rate by a factor of 0.75 every 5 epochs based on our training observations and empirical evaluation, while for Agriculture-vison models, we use initial learning rates of 2.8×10^{-4} and apply a cosine annealing scheduler that reduces the learning rates over epochs (for a maximum epoch of 40).

4.4. *Augmentation and evaluation methods*

During training, all data is sampled uniformly and augmented with random flip (horizontal and vertical), rotation (90 degree), Gaussian noise, and brightness contrast (all probabilities are 0.5) for each epoch. 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.

During test and evaluation, we apply test time augmentation (TTA) in terms of flipping and mirroring. For Vaihingen data, 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. For the agriculture-vision data, we first apply TTA on the full size test image (512×512) and average the predictions to get the final output. The performance is measured by the F1-score for Vaihingen dataset, and the modified Intersection over Union (IoU) (Chiu et al. 2020a) for the agriculture-vision dataset. Please note that the mIoU metric was computed by averaging over the nine classes (including the 'Background' class) in the Agriculture-

Vision benchmark dataset.

4.5. *Test results*

We tested our trained models on the hold out test sets of the Vaihingen and Agriculture-Vision datasets. The test results are shown in Table 2 and Table 3, respectively. It is clearly visible for all the cases that our method outperforms all the state-of-the-art methods with a significant margin. For the Vaihingen dataset, it can be seen that the accuracy for the 'Car' class (90.8% F1-score) is notably improved (+2.5%) using our method in comparison to other methods. In case of the Agriculture-Vision dataset, many difficult classes also show significant increases in terms of IoU accuracies, e.g., double plant (+9.4%), drydown (+5.8%), endrow (+8.2%), and planter skip (+5.3%), etc.

A qualitative comparison of the segmentation results from our trained models and the ground truths on the validation data are shown in Fig. 9 and Fig. 10. It can be visually verified that the classification maps obtained from our PAG-Net models tend to be less noisy and have smooth and fine-gained boundary recovery without any post-processing. In addition, our multi-modal PAGNet^b model obtained the best performance on the Agriculture-Vision dataset with 48.2% mIoU (+4.2%) with fewer training parameters (6.14M) and 2× faster training and inference speed on both CPU (127ms) and GPU (11ms) in comparison to MSCG-Net50 as shown in Table 1. It is worth noting that our two PAF-base unimodal models (PAFNet^a and PAFNet^b) also obtain the best performance compared to other unimodal methods on both the Vaihingen and Agriculture-Vision datasets.

Table 2. Comparisons between our method with other published methods on the hold-out test images of Vaihingen Dataset.

Models	OA	Surface	Building	Low-veg	Tree	Car	mF1
UOA (Lin et al. 2016)	0.876	0.898	0.921	0.804	0.882	0.820	0.865
DNN_HCRF (Liu et al. 2019)	0.878	0.901	0.932	0.814	0.872	0.720	0.848
ADL_3 (Paisitkriangkrai et al. 2015)	0.880	0.895	0.932	0.823	0.882	0.633	0.833
DST_2 (Sherrah 2016)	0.891	0.905	0.937	0.834	0.892	0.726	0.859
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. 2017)	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. 2020)	0.904	0.927	0.953	0.833	0.894	0.883	0.898
SCG-GCN (Liu et al. 2020b)	0.904	0.924	0.948	0.839	0.897	0.880	0.898
FuseNet(IRRG+DSM/NDVI)	0.908	0.913	0.943	**0.848**	0.899	0.859	0.901
PAFNet^a(DSM-IRRG)	0.906	0.929	0.949	0.826	0.894	0.905	0.900
PAGNet^a(IRRG+DSM)	**0.913**	**0.930**	**0.952**	0.843	**0.900**	**0.908**	**0.907**

Table 3. Comparisons between our method with other published methods in terms of mIoUs and class IoUs on the hold-out Agriculture-Vision test set.

Models	mIoU	Background	Double plant	Drydown	Endrow	Nutrient deficiency	Planter skip	Water	Waterway	Weed cluster
DeepLabv3(os=8)	0.322	0.704	0.215	0.510	0.126	0.394	0.204	0.157	0.337	0.250
DeepLabv3+(os=8)	0.391	0.710	0.197	0.509	0.195	0.413	0.244	0.623	0.341	0.280
DeepLabv3(os=16)	0.422	0.727	0.252	0.536	0.210	0.440	0.246	0.704	0.386	0.299
DeepLabv3+(os=16)	0.424	0.725	0.260	0.536	0.241	0.442	0.244	0.703	0.379	0.288
FPN-ResNet (Chiu et al. 2020b)	0.437	0.726	0.279	0.523	0.243	0.438	0.310	**0.713**	**0.388**	0.309
MSCG-Net50 (Liu et al. 2020b)	0.441	0.716	0.289	0.513	0.270	0.442	0.331	0.692	0.366	0.349
PAFNet^b(IR-RGB)	0.442	0.687	0.343	0.562	0.281	0.420	0.305	0.680	0.378	0.324
PAGNet^b(RGB+IR)	**0.482**	**0.740**	**0.383**	**0.581**	**0.352**	**0.460**	**0.384**	0.686	0.373	**0.379**

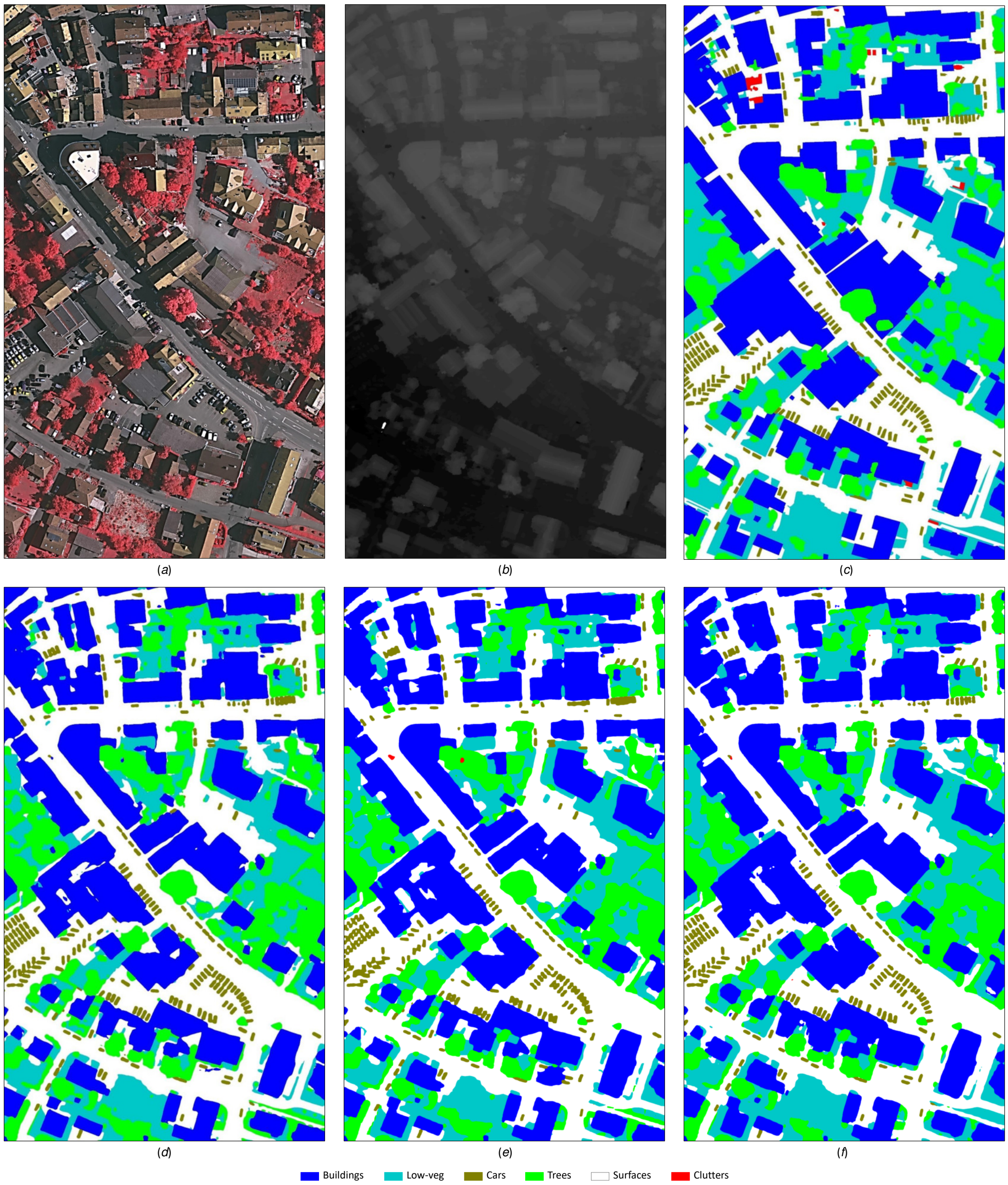


Figure 9. Segmentation results for the test image of Vaihingen tile-27: (a) the test IRRG image, (b) the DSM image, (c) the ground truth, (d) DDCM-R50, (e) SCG-GCN, (f) PAGNet^a

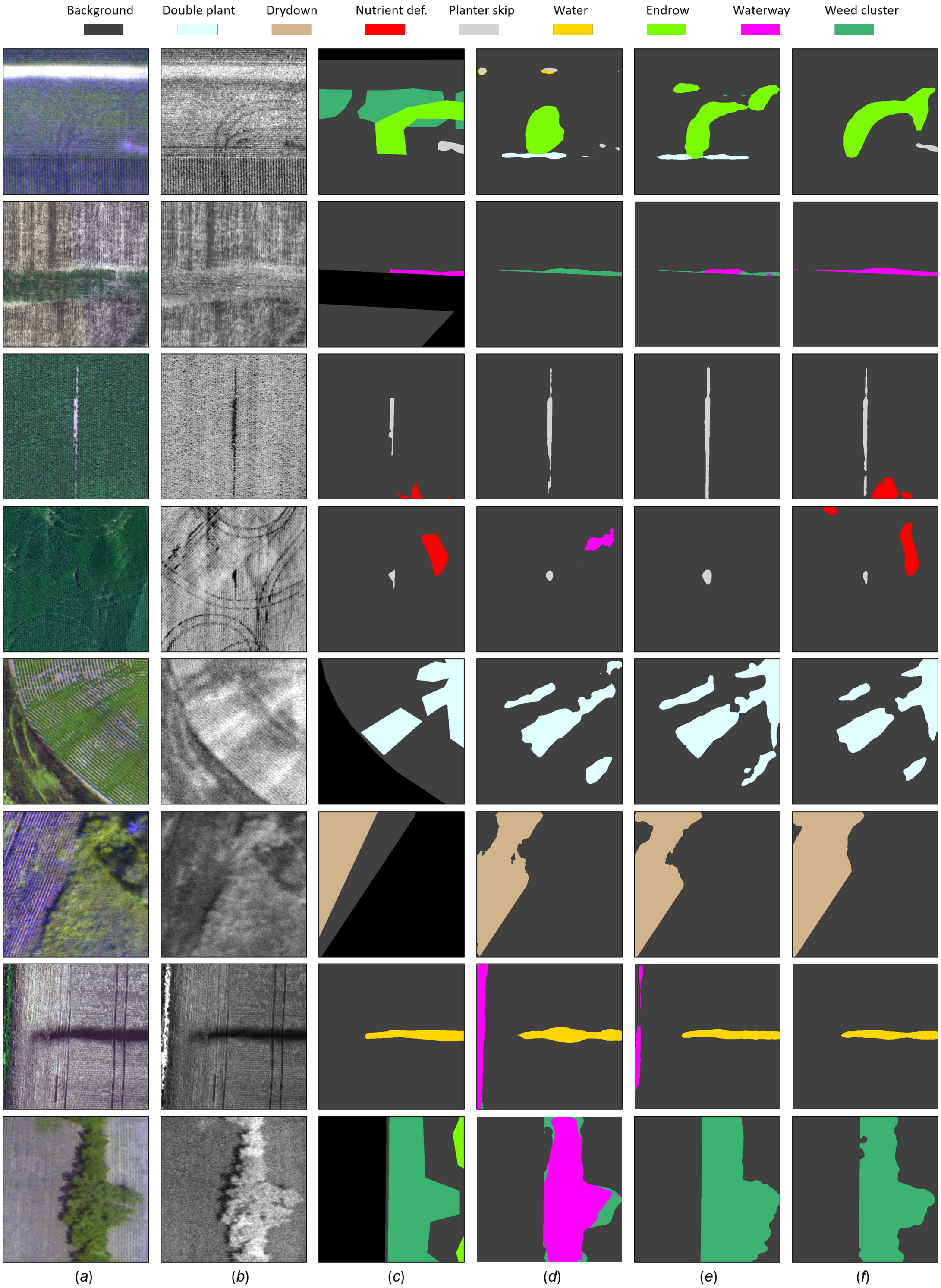


Figure 10. Segmentation results on the validation images of the agriculture-vision dataset: (a) the RGB images, (b) the NIR images, (c) the ground truth, (d) MSCG-Net50, (e) PAFNet^b, (f) PAGNet^b

5. Discussion

In our network, pyramid attention fusion (PAF) modules are employed to capture multi-level and cross-view robust representations, and gated fusion units (GFU) are designed to bridge and interact among different modalities to better combine multi-modality information. To validate the effectiveness of these modules, we perform ablation experiments on the Vaihingen dataset.

5.1. *Effect of the pyramid attention fusion module*

In Table 4, we evaluate our model's performance by removing the key components of the PAF module, i.e., pyramid cross-view encoder (PCE), attention construction and updating (ACU) module as shown in Fig. 3. Since PCE and ACU are interdependent, we are not able to just use ACU without the PCE unit. We, therefore, evaluated the following two variations: 1) V-1: replacing both PCE and ACU with using multi-level concatenation fusion networks (FPN-style) with the same number of hidden features of the PAF module. 2) V-2: only removing ACU module. It is evidently shown that in absence of our pyramid attention fusion module, the V-1 model tends to underperform a lot on small objects (e.g., 'Car' -4.8%), while only applying our pyramid cross-view encoder block, our V-2 model improves the performance a bit overall (mF1: $+0.9\%$ in contrast to V-1 model) but still underperforms on the small class 'Car' (mF1: -2.4% in contrast to PAGNet).

Table 4. The effect of the two key units (PCE, ACU) of our PAF module on the hold-out test images of Vaihingen Dataset.

Model	PCE	ACU	OA	Δ%	Building	Δ%	Car	Δ%	mF1	Δ%	Steps (K)
V-1	✗	✗	0.898	-1.5	0.942	-1.0	0.860	-4.8	0.890	-1.7	31
V-2	✓	✗	0.906	-0.7	0.945	-0.7	0.884	-2.4	0.899	-0.8	25
PAGNet	✓	✓	**0.913**	-	**0.952**	-	**0.908**	-	**0.907**	-	29

We also investigated the effect of the latent features and cross-view settings on the performance. Note that, we assume that the number of latent features (c) should be close to the number for classes (i.e., 6 for Vaihingen dataset). The latent features are thus set to be in the range of $\{4, 6, 8, 12\}$, and the number of views (v) are in the range of $\{1, 2, 3\}$. Table 5 presents the details of the evaluation results where five models are trained on various latent features and cross-view settings. Our model with latent features of 6 and view number of 3 achieves the best results. Note that the latent feature number does not show a significant impact on overall performance (mF1: ±0.5%), while the number of views seems to be more sensitive on both overall results (mF1: ±1.0%) and the performance on small objects (mF1: ±2.9%).

Table 5. The effect of different the number of latent features and views of our PAF module on the hold-out Vaihingen test set.

Model	c	v	OA	Δ%	Building	Δ%	Car	Δ%	mF1	Δ%	Steps (K)
PAG-v1	4	3	0.907	-0.6	0.950	-0.2	0.904	-0.4	0.903	-0.4	35
PAG-v2	8	3	0.910	-0.3	**0.957**	+0.3	0.905	-0.3	0.904	-0.3	21
PAG-v3	12	3	0.908	-0.5	0.952	0	0.891	-1.7	0.902	-0.5	17
PAG-v4	6	1	0.902	-1.1	0.947	-0.5	0.879	-2.9	0.897	-1.0	19
PAG-v5	6	2	0.909	-0.4	0.949	-0.3	0.898	-1.0	0.904	-0.3	23
PAGNet	6	3	**0.913**	-	0.952	-	**0.908**	-	**0.907**	-	29

$^*c$ is the number of latent features and v denotes the number of views. Here Steps K=1000 denote training iterations.

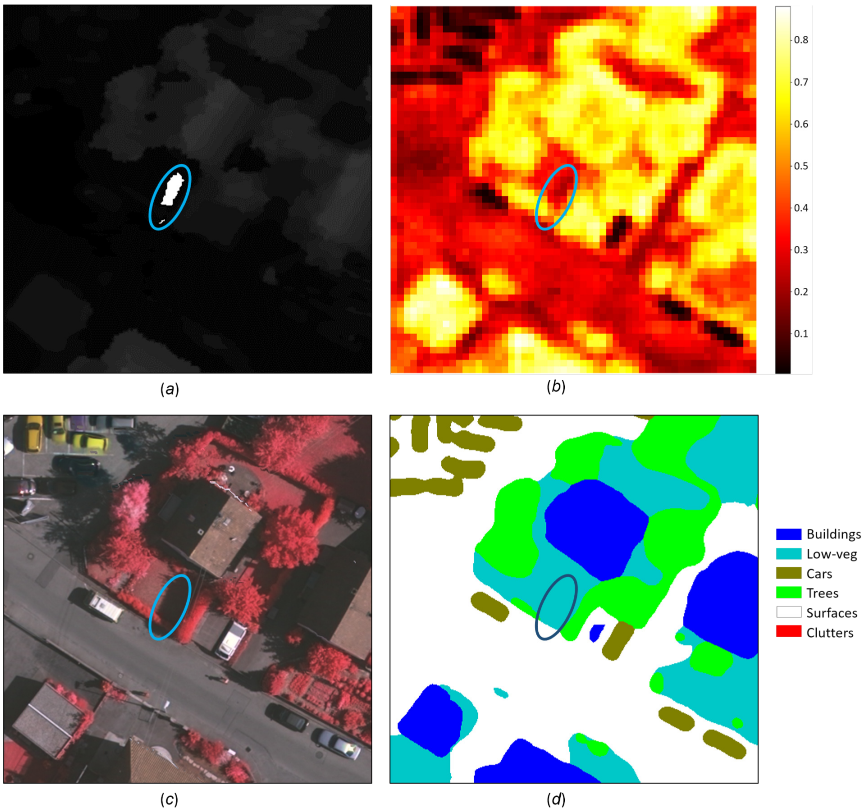
5.2. *Effect of the gated fusion unit*

The GFU module plays a very important role for the effectiveness and efficiency of our multi-modal PAGNet. We, therefore, evaluate GFU by comparing it with the other two commonly used fusion methods (i.e., element-wise summing, and concatenation). Table 6 displays the performance of these three methods. It is clearly shown that simply concatenating or summing multi-modality features will cause a degradation in performance to unimodal models. Our GFU approach, instead, shows notable performance gains (mF1: +1.1 $\sim$ 1.4%) in general and significantly boosts the results on small objects (mF1: +2.2 $\sim$ 3.3%) and improves the training converges speed (+2x faster). We visualized the GFU module learned attention gate map as shown in Figure 11. It illustrates that GFU module is able to capture a significant or complementary part of the information contained in the DSM data and diminish the influence of noisy data as well.

Table 6. Test performance of different fusion settings on Vaihingen test set.

⊕	ⓒ	ⓖ	OA	Δ%	Building	Δ%	Car	Δ%	mF1	Δ%	Steps (K)	Δ
✓			0.901	-1.2	0.950	-0.2	0.875	-3.3	0.893	-1.4	67	2.3x
	✓		0.906	-0.7	0.946	-0.6	0.886	-2.2	0.896	-1.1	92	3.1x
		✓	**0.913**	-	**0.952**	-	**0.908**	-	**0.907**	-	**29**	-

*⊕ denote point-wise summing fusion, ⓒ is concatenation fusion, and ⓖ is our gated fusion method.



(a) (b)

(c) (d)

Figure 11. Heatmap of the GFU learned attention gate : (a) the DSM image (containing some noisy patch), (b) the attention gate heatmap, (c) the IRRG image, (d) the prediction.

5.3. *Effect of missing and noisy data*

We also evaluate the performance of our model to handle situations where DSM data are missing, noisy and completely interfered during testing. Specifically, we assume that the IRRG data modality of the Vaihingen dataset is available, while the DSM modality is missing (letting DSM data to 0), random noisy signal (using white noise data sampled from 0 to 255), or completely interfered (setting data value to 255). Table 7 illustrates the results. It clearly indicates that our model is capable of dealing with missing or noisy data. In other words, our model, which was trained with all data modalities (e.g., IRRG+DSM), generalizes well to circumstances in which extra modality data (e.g, DSM) is absent or entirely noisy during the testing phase. Our model's weakness for missing and noisy data modalities is that it does not handle missing primary modalities adequately (e.g., IRRG or RGB). In many situations, this is not an issue because it is usual to merely evaluate a small number of extra data modalities in remote sensing.

Table 7. Evaluation results with missing, noisy and interfered DSM data on the hold-out test images of Vaihingen Dataset.

Modalities	OA	Surface	Building	Low-veg	Tree	Car	mF1
Baseline (IRRG+DSM)	**0.913**	**0.930**	**0.952**	**0.843**	**0.900**	**0.908**	**0.907**
(IRRG+missing-DSM)	0.908	0.926	0.949	0.839	0.897	0.903	0.903
(IRRG+random-noisy-data)	0.904	0.923	0.943	0.837	0.898	0.900	0.900
(IRRG+interferred-data)	0.899	0.905	0.943	0.836	0.892	0.902	0.896

6. Conclusions

We presented a novel pyramid attention and gated fusion method for multi-modality land cover and land use mapping in remote sensing. Our proposed pyramid attention fusion (PAF) module can effectively capture multi-level and cross-view attention maps to obtain rich and robust representations, that can further be flexibly harnessed as a key fusion bridge between multiple modalities using our developed gated fusion (GFU) algorithms. The GFU module can tune the noisy modalities and extract complementary features to improve the performance of our multimodal models. Built upon the PAF and GFU modules, our MultiModNet framework provides an end-to-end and lightweight multi-modal segmentation solution, which achieves the state-of-the-art performance and outperforms the strong baselines on two different representative remote sensing datasets. In addition, our methods easily generalize to more than two modalities for addressing more complicated problems in remote sensing.

Acknowledgements

The benchmark datasets: the Vaihingen dataset was provided by the International Society for Photogrammetry and Remote Sensing (ISPRS); the Agriculture-Vision dataset was provided by UIUC, IntelinAir and University of Oregon. 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 Norges Forskningsråd under Grant 272399 and Grant 309439.

References

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.

Audebert, Nicolas, Bertrand Le Saux, and Sébastien Lefèvre. 2018. "Beyond RGB: Very high resolution urban remote sensing with multimodal deep networks." *ISPRS J. Photogramm. Remote Sensing* 140: 20–32.

Audebert, Nicolas, Bertrand Le Saux, and Sébastien Lefèvre. 2019. "Deep learning for classification of hyperspectral data: A comparative review." *IEEE Geoscience and Remote Sensing Magazine* 7 (2): 159–173.

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. 2017. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (12): 2481–2495.

Bello, Olalekan Mumin, and Yusuf Adedoyin Aina. 2014. "Satellite remote sensing as a tool in disaster management and sustainable development: towards a synergistic approach." *Procedia-Social and Behavioral Sciences* 120: 365–373.

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.

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.

Chiu, Mang Tik, Xingqian Xu, Kai Wang, Jennifer Hobbs, Naira Hovakimyan, Thomas S. Huang, Honghui Shi, et al. 2020a. "The 1st Agriculture-Vision Challenge: Methods and Results." In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 212–218.

Chiu, Mang Tik, Xingqian Xu, Yunchao Wei, Zilong Huang, Alexander G. Schwing, Robert Brunner, Hrant Khachatrian, et al. 2020b. "Agriculture-Vision: A Large Aerial Image Database for Agricultural Pattern Analysis." In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June.

Couprie, Camille, Clément Farabet, Laurent Najman, and Yann LeCun. 2013. "Indoor semantic segmentation using depth information." *arXiv preprint arXiv:1301.3572* .

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. 2021. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." *ICLR* .

Fan, Chao, Cheng Zhang, Alex Yahja, and Ali Mostafavi. 2021. "Disaster City Digital Twin:

A vision for integrating artificial and human intelligence for disaster management." *International Journal of Information Management* 56: 102049.

Feng, Quanlong, Dehai Zhu, Jianyu Yang, and Baoguo Li. 2019. "Multisource hyperspectral and lidar data fusion for urban land-use mapping based on a modified two-branch convolutional neural network." *ISPRS International Journal of Geo-Information* 8 (1): 28.

Fu, J., J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu. 2019. "Dual Attention Network for Scene Segmentation." In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3141–3149.

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

Ghosh, Arthita, Max Ehrlich, Sohil Shah, Larry Davis, and Rama Chellappa. 2018. "Stacked U-Nets for Ground Material Segmentation in Remote Sensing Imagery." In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 252–2524.

Gómez-Chova, Luis, Devis Tuia, Gabriele Moser, and Gustau Camps-Valls. 2015. "Multimodal classification of remote sensing images: A review and future directions." *Proceedings of the IEEE* 103 (9): 1560–1584.

Hazirbas, Caner, Lingni Ma, Csaba Domokos, and Daniel Cremers. 2016. "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture." In *Asian Conference on Computer Vision*, 213–228. Springer.

Hong, Danfeng, Lianru Gao, Naoto Yokoya, Jing Yao, Jocelyn Chanussot, Qian Du, and Bing Zhang. 2020. "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification." *IEEE Transactions on Geoscience and Remote Sensing* .

Howard, Andrew, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, et al. 2019. "Searching for mobilenetv3." In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1314–1324.

Hu, Jie, Li Shen, and Gang Sun. 2018. "Squeeze-and-excitation networks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7132–7141.

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, Arnt-Borre Salberg, and Robert Jenssen. 2016. "Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 1–9.

Kampffmeyer, Michael, 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.

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

Li, Xiao, Lin Lei, Yuli Sun, Ming Li, and Gangyao Kuang. 2020. "Multimodal bilinear fusion network with second-order attention-based channel selection for land cover classification." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13: 1011–1026.

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.

Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. "Feature pyramid networks for object detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2117–2125.

Liu, Q., M. Kampffmeyer, R. Jenssen, and A. B. Salberg. 2020. "Dense Dilated Convolutions'

Merging Network for Land Cover Classification." *IEEE Transactions on Geoscience and Remote Sensing* 58 (9): 6309–6320.

Liu, Qinghui, Michael Kampffmeyer, Robert Jenssen, and Arnt-Børre Salberg. 2020a. "Self-Constructing Graph Convolutional Networks for Semantic Labeling." In *Proceedings of IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, .

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 *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June.

Liu, Yansong, Sankaranarayanan Piramanayagam, Sildomar T Monteiro, and Eli Saber. 2019. "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.

Maggiori, Emmanuel, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. 2017. "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark." In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 3226–3229. IEEE.

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.

Mohla, Satyam, Shivam Pande, Biplab Banerjee, and Subhasis Chaudhuri. 2020. "FusAtNet: Dual Attention based SpectroSpatial Multimodal Fusion Network for Hyperspectral and LiDAR Classification." In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 416–425.

Mou, Lichao, and Xiao Xiang Zhu. 2019. "Learning to pay attention on spectral domain: A spectral attention module-based convolutional network for hyperspectral image classification." *IEEE Transactions on Geoscience and Remote Sensing* 58 (1): 110–122.

Noor, Norzailawati Mohd, Alias Abdullah, and Mazlan Hashim. 2018. "Remote sensing UAV/drones and its applications for urban areas: a review." In *IOP conference series: Earth and environmental science*, Vol. 169, 012003. IOP Publishing.

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.

Pashaei, Mohammad, Hamid Kamangir, Michael J Starek, and Philippe Tissot. 2020. "Review and evaluation of deep learning architectures for efficient land cover mapping with UAS hyper-spatial imagery: A case study over a wetland." *Remote Sensing* 12 (6): 959.

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.

Salberg, Arnt-Børre. 2011. "Land Cover Classification of Cloud-Contaminated Multitemporal High-Resolution Images." *IEEE Transactions on Geoscience and Remote Sensing* 49 (1): 377–387.

Salberg, Arnt-Børre, Øystein Rudjord, and Anne H. Schistad Solberg. 2014. "Oil Spill Detection in Hybrid-Polarimetric SAR Images." *IEEE Transactions on Geoscience and Remote Sensing* 52 (10): 6521–6533.

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.

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

Xu, Xiaodong, Wei Li, Qiong Ran, Qian Du, Lianru Gao, and Bing Zhang. 2017. “Multisource remote sensing data classification based on convolutional neural network.” *IEEE Transactions on Geoscience and Remote Sensing* 56 (2): 937–949.

Xu, Yonghao, Bo Du, and Liangpei Zhang. 2018. “Multi-Source Remote Sensing Data Classification via Fully Convolutional Networks and Post-Classification Processing.” In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 3852–3855.

Xu, Yonghao, Bo Du, Liangpei Zhang, Daniele Cerra, Miguel Pato, Emiliano Carmona, Saurabh Prasad, Naoto Yokoya, Ronny Hänsch, and Bertrand Le Saux. 2019. “Advanced Multi-Sensor Optical Remote Sensing for Urban Land Use and Land Cover Classification: Outcome of the 2018 IEEE GRSS Data Fusion Contest.” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12 (6): 1709–1724.

Yuan, Yuhui, Xilin Chen, and Jingdong Wang. 2020. “Object-Contextual Representations for Semantic Segmentation.” In *Proceedings of the European Conference on Computer Vision (ECCV)*, 173–190.

Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 2017. “Pyramid Scene Parsing Network.” In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6230–6239.

Bibliography

[1] Luis Gómez-Chova, Devis Tuia, Gabriele Moser, and Gustau Camps-Valls. Multimodal classification of remote sensing images: A review and future directions. *Proceedings of the IEEE*, 103(9):1560–1584, 2015.

[2] Jun Li, Yanqiu Pei, Shaohua Zhao, Rulin Xiao, Xiao Sang, and Chengye Zhang. A review of remote sensing for environmental monitoring in china. *Remote Sensing*, 12(7):1130, 2020.

[3] Qiangqiang Yuan, Huanfeng Shen, Tongwen Li, Zhiwei Li, Shuwen Li, Yun Jiang, Hongzhang Xu, Weiwei Tan, Qianqian Yang, Jiwen Wang, et al. Deep learning in environmental remote sensing: Achievements and challenges. *Remote Sensing of Environment*, 241:111716, 2020.

[4] Mang Tik Chiu, Xingqian Xu, Kai Wang, Jennifer Hobbs, Naira Hovakimyan, Thomas S. Huang, Honghui Shi, Yunchao Wei, Zilong Huang, Alexander Schwing, Robert Brunner, Ivan Dozier, Wyatt Dozier, Karen Ghandilyan, David Wilson, Hyunseong Park, Junhee Kim, Sungho Kim, Qinghui Liu, Michael C. Kampffmeyer, Robert Jenssen, Arnt B. Salberg, Alexandre Barbosa, Rodrigo Trevisan, Bingchen Zhao, Shaozuo Yu, Siwei Yang, Yin Wang, Hao Sheng, Xiao Chen, Jingyi Su, Ram Rajagopal, Andrew Ng, Van Thong Huynh, Soo-Hyung Kim, In-Seop Na, Ujjwal Baid, Shubham Innani, Prasad Dutande, Bhakti Baheti, Sanjay Talbar, and Jianyu Tang. The 1st agriculture-vision challenge: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 212–218, 2020.

[5] Arnt-Børre Salberg, Øivind Due Trier, and Michael Kampffmeyer. Large-scale mapping of small roads in lidar images using deep convolutional neural networks. In *Scandinavian Conference on Image Analysis*, pages 193–204. Springer, 2017.

[6] Norzailawati Mohd Noor, Alias Abdullah, and Mazlan Hashim. Remote sensing uav/drones and its applications for urban areas: a review. In *IOP Conference Series: Earth and Environmental Science*, volume 169, page 012003. IOP Publishing, 2018.

[7] Arnt-Børre Salberg, Øystein Rudjord, and Anne H. Schistad Solberg. Oil spill detection in hybrid-polarimetric SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, 52(10):6521–6533, 2014.

[8] Olalekan Mumin Bello and Yusuf Adedoyin Aina. Satellite remote sensing as a tool in disaster management and sustainable development: towards a synergistic approach. *Procedia-Social and Behavioral Sciences*, 120:365–373, 2014.

[9] Chao Fan, Cheng Zhang, Alex Yahja, and Ali Mostafavi. Disaster city digital twin: A vision for integrating artificial and human intelligence for disaster management. *International Journal of Information Management*, 56:102049, 2021.

[10] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. DeepGlobe 2018: A challenge to parse the earth through satellite images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2018.

[11] Mang Tik Chiu, Xingqian Xu, Yunchao Wei, Zilong Huang, Alexander G Schwing, Robert Brunner, Hrant Khachatrian, Hovnatan Karapetyan, Ivan Dozier, Greg Rose, et al. Agriculture-vision: A large aerial image database for agricultural pattern analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2828–2838, 2020.

[12] Franz Rottensteiner, Gunho Sohn, Jaewook Jung, Markus Gerke, Caroline Baillard, Sebastien Benitez, and Uwe Breitkopf. 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, 2012.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[15] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017.

[16] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre. Joint learning from earth observation and OpenStreetMap data to get faster better semantic maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 67–75, 2017.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[18] Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. Speech recognition using deep neural networks: A systematic

review. *IEEE access*, 7:19143–19165, 2019.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[20] NASA. What is remote sensing?, Aug 2021. `https://earthdata.nasa.gov/learn/backgrounders/remote-sensing`.

[21] Cecie Starr, Christine Evers, and Lisa Starr. *Biology: concepts and applications*. Cengage Learning, 2014.

[22] Joao Fonseca, Georgios Douzas, and Fernando Bacao. Improving imbalanced land cover classification with K-Means SMOTE: Detecting and oversampling distinctive minority spectral signatures. *Information*, 12(7):266, 2021.

[23] José A Sáez, Bartosz Krawczyk, and Michał Woźniak. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, 57:164–178, 2016.

[24] Michael Alonzo, Bodo Bookhagen, and Dar A Roberts. Urban tree species mapping using hyperspectral and LiDAR data fusion. *Remote Sensing of Environment*, 148:70–83, 2014.

[25] Zhiying Cao, Wenhui Diao, Xian Sun, Xiaode Lyu, Menglong Yan, and Kun Fu. C3Net: Cross-modal feature recalibrated, cross-scale semantic aggregated and compact network for semantic segmentation of multi-modal high-resolution aerial images. *Remote Sensing*, 13(3):528, 2021.

[26] Danfeng Hong, Lianru Gao, Naoto Yokoya, Jing Yao, Jocelyn Chanussot, Qian Du, and Bing Zhang. More diverse means better: Multimodal deep learning meets remote sensing imagery classification. *IEEE Transactions on Geoscience and Remote Sensing*, 59(5):4340–4354, 2020.

[27] Xiao Xiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4):8–36, 2017.

[28] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[29] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

[30] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis

Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[31] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[32] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[33] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[34] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and Regression Trees*. Routledge, 2017.

[35] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[36] Suresh Balakrishnama and Aravind Ganapathiraju. Linear discriminant analysis - a brief tutorial. *Institute for Signal and Information Processing*, 18(1998):1–8, 1998.

[37] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

[38] Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pages 5628–5637. PMLR, 2019.

[39] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607. PMLR, 2020.

[40] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[42] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[43] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, volume 30, page 3. Citeseer, 2013.

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into

rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[45] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

[46] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

[47] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[48] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[49] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The Handbook of Brain Theory and Neural Networks*, 3361(10):1995, 1995.

[50] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1389–1397, 2017.

[51] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1492–1500, 2017.

[52] Daniël M Pelt and James A Sethian. A mixed-scale dense convolutional neural network for image analysis. *Proceedings of the National Academy of Sciences*, 115(2):254–259, 2018.

[53] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[54] 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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.

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

[56] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1251–1258, 2017.

[57] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[58] Ru Zhang, Feng Zhu, Jianyi Liu, and Gongshen Liu. Depth-wise separable convolutions and multi-level pooling for an efficient spatial CNN-based steganalysis. *IEEE Transactions on Information Forensics and Security*, 15:1138–1150, 2019.

[59] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

[60] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016.

[61] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017.

[62] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 801–818, 2018.

[63] Yuhong Li, Xiaofan Zhang, and Deming Chen. CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1091–1100, 2018.

[64] Yunchao Wei, Huaxin Xiao, Honghui Shi, Zequn Jie, Jiashi Feng, and Thomas S Huang. Revisiting dilated convolution: A simple approach for weakly and semi-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7268–7277, 2018.

[65] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

[66] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for MobileNetV3. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019.

[67] Guilin Liu, Kevin J Shih, Ting-Chun Wang, Fitsum A Reda, Karan Sapra,

Zhiding Yu, Andrew Tao, and Bryan Catanzaro. Partial convolution based padding. *arXiv preprint arXiv:1811.11718*, 2018.

[68] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12), 2010.

[69] Marc Ranzato, Christopher Poultney, Sumit Chopra, Yann LeCun, et al. Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems*, 19:1137, 2007.

[70] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *International Conference on Machine Learning (ICML)*, 2011.

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

[72] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691*, 2019.

[73] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning (International Conference on Machine Learning (ICML))*, pages 1462–1471. PMLR, 2015.

[74] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain Monte Carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226. PMLR, 2015.

[75] Tejas D Kulkarni, Will Whitney, Pushmeet Kohli, and Joshua B Tenenbaum. Deep convolutional inverse graphics network. *arXiv preprint arXiv:1503.03167*, 2015.

[76] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in Neural Information Processing Systems*, 28:3483–3491, 2015.

[77] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks. *arXiv preprint arXiv:1905.07953*, 2019.

[78] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In *Advances in Neural Information Processing Sstems*, pages 4558–4567, 2018.

[79] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[80] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[81] Boris Knyazev, Xiao Lin, Mohamed R Amer, and Graham W Taylor. Image classification with hierarchical multigraph networks. *arXiv preprint arXiv:1907.09000*, 2019.

[82] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5177–5186, 2019.

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

[84] 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)*, 38(5):146, 2019.

[85] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4558–4567, 2018.

[86] Xiaojuan Qi, Renjie Liao, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. 3D graph neural networks for RGBD semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5199–5208, 2017.

[87] Song Ouyang and Yansheng Li. Combining deep semantic segmentation network and graph convolutional neural network for semantic segmentation of remote sensing imagery. *Remote Sensing*, 13(1), 2021.

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

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

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

[91] Michael C. 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/CVF Conference on Computer Vision*

and Pattern Recognition (CVPR), pages 11487–11496, 2019.

[92] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

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

[94] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations, CBLS, April 2014*, 2014.

[95] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

[96] 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.

[97] Fan Chung, Linyuan Lu, and Van Vu. Spectra of random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 100(11):6313–6318, 2003.

[98] Michael C. Kampffmeyer. *Advancing Segmentation and Unsupervised Learning Within the Field of Deep Learning*. PhD dissertation, UiT The Arctic University of Norway, 2018.

[99] G Kuntimad and Heggere S Ranganath. Perfect image segmentation using pulse coupled neural networks. *IEEE Transactions on Neural networks*, 10(3):591–598, 1999.

[100] Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Class segmentation and object localization with superpixel neighborhoods. In *IEEE International Conference on Computer Vision (ICCV)*, pages 670–677. IEEE, 2009.

[101] Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic texton forests for image categorization and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[102] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[103] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.

[104] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

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

[106] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1520–1528, 2015.

[107] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[108] Zhenli Zhang, Xiangyu Zhang, Chao Peng, Xiangyang Xue, and Jian Sun. Exfuse: Enhancing feature fusion for semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 269–284, 2018.

[109] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3146–3154, 2019.

[110] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 173–190, 2020.

[111] International Society for Photogrammetry and Remote Sensing (ISPRS). 2D Semantic Labeling Contest. online, 2018. `http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html`.

[112] Michael C. Kampffmeyer, Arnt-Børre Salberg, and Robert Jenssen. Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1–9, 2016.

[113] Jamie Sherrah. Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. *CoRR*, abs/1606.02585, 2016.

[114] Guosheng Lin, Chunhua Shen, Anton Van Den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3194–3203, 2016.

[115] 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*, abs/1612.01337, 2016.

[116] 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*, pages 180–196. Springer, 2016.

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

[118] Lichao Mou and Xiao Xiang Zhu. RiFCN: Recurrent network in fully convolutional network for semantic segmentation of high resolution remote sensing images. *CoRR*, abs/1805.02091, 2018.

[119] Michael C. Kampffmeyer, Arnt-Børre Salberg, and Robert Jenssen. 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, 2018.

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

[121] Mohamed Samy, Karim Amer, Kareem Eissa, Mahmoud Shaker, and Mohamed ElHelw. NU-Net: Deep residual wide field of view convolutional neural network for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 267–2674. IEEE, 2018.

[122] Alex Davydow, OU Neuromation, and Sergey Nikolenko. Land cover classification with superpixels and jaccard index post-optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 280–2804. IEEE, 2018.

[123] Guillem Pascual, Santi Seguí, and Jordi Vitria. Uncertainty gated network for land cover segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[124] Selim Seferbekov, Vladimir Iglovikov, Alexander Buslaev, and Alexey Shvets. Feature pyramid network for multi-class land segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[125] Arthita Ghosh, Max Ehrlich, Sohil Shah, Larry Davis, and Rama Chellappa. Stacked U-Nets for Ground Material Segmentation in Remote Sensing Im-

agery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 252–2524, 2018.

[126] Tzu-Sheng Kuo, Keng-Sen Tseng, Jia-Wei Yan, Yen-Cheng Liu, and Yu-Chiang Frank Wang. Deep aggregation net for land cover classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[127] Chao Tian, Cong Li, and Jianping Shi. Dense fusion classmate network for land cover classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 262–2624. IEEE, 2018.

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

[129] A Helen Victoria and G Maragatham. Automatic tuning of hyperparameters using bayesian optimization. *Evolving Systems*, 12:217–223, 2021.

[130] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[131] Javiera Castillo-Navarro, Bertrand Le Saux, Alexandre Boulch, Nicolas Audebert, and Sébastien Lefèvre. Semi-supervised semantic segmentation in Earth Observation: the MiniFrance suite, dataset analysis and multi-task network study. *Machine Learning*, 2021.

[132] Jingjing Li, Erpeng Chen, Zhengming Ding, Lei Zhu, Ke Lu, and Heng Tao Shen. Maximum density divergence for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3918–3930, 2021.