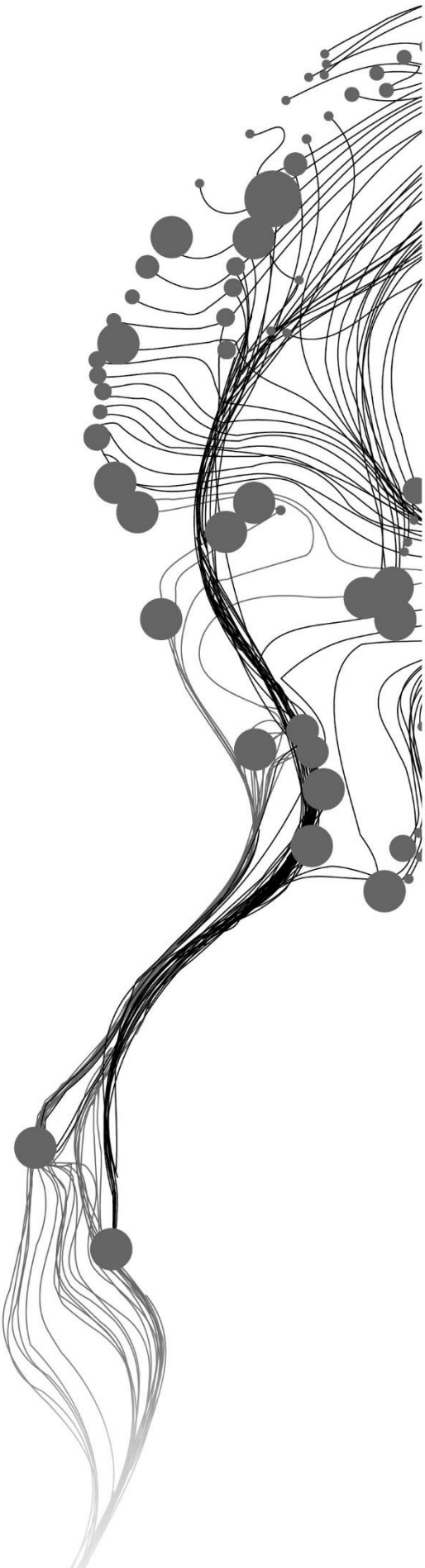
SEMANTIC SEGMENTATION OF URBAN AIRBORNE OBLIQUE IMAGES

LI LIU
February, 2019

SUPERVISORS:
Dr. M. Y. Yang
Dr. ir. S.J. Oude Elberink



SEMANTIC SEGMENTATION OF URBAN AIRBORNE OBLIQUE IMAGES

LI LIU
Enschede, The Netherlands, February, 2019

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: Geoinformatics

SUPERVISORS:
Dr. M. Y. Yang
Dr. ir. S.J. Oude Elberink

THESIS ASSESSMENT BOARD:
Prof.dr.ir. M.G. Vosselman (Chair)
Dr. R.C. Lindenbergh, Delft University of Technology, Optical and Laser Remote Sensing
etc

ABSTRACT

Computer recognition and classification of remote sensing digital images is an important research direction in remote sensing image processing. A lot of classification methods has been widely used for urban scene understanding, the large availability of airborne images necessitates an image classification approach that automatically learns relevant features from the raw image and performs semantic segmentation in an end-to-end framework. For this, deep learning methodologies were preferred, specifically in the form of convolutional neural network (CNN) which is widely used in computer vision. In this research, airborne oblique images are used as data source for semantic segmentation in urban areas, we investigate the ability of Deep Convolutional Neural Network (DCNN) structures that is designed based on U-net architecture and adapt it to our dataset. Different from its original architecture based on VGG11, U-net based on VGG16 architecture with deeper layers is applied in this study. However, the training time of deep neural network is too long because of enormous parameters and present some challenges for model training. Here, depthwise separable convolution is coupled with convolution block in our architecture to reduce the model parameters and improve model efficiency. Outputs of deep neural network are sometimes noisy and the boundary of objects are not very smooth because continuous pooling layers reduce the ability of localization. As a result, the mean field approximation to the fully connected conditional random field (CRF) inference which models label and spectral compatibility in the pixel-neighbourhood is applied to refine the output of other classifiers. Finally, we achieved an end-to-end model which connect the scores of the U-net classification with a fully connected Conditional Random Field (CRF) architecture and make the inference as Recurrent Neural Network (RNN), to consider the local class label dependencies in an end-to-end structure and make full use of DCNNs and probability random field model. We compare the fully connected CRF optimized segmentation results with those obtained by applying the trained U-net classifier. Segmentation results indicate that our classification U-net model with deeper layers have performed favourably, the modified network not only ensures the accuracy but also greatly shortens the time of model training in the large-scale image classification problem, furthermore, this end-to-end model which combines fully connected CRF has effectively improved the segments of U−net.

Keywords
Semantic segmentation, convolutional neural network, conditional random field, deep learning

ACKNOWLEDGEMENTS

I would like to thank my supervisors Dr. M. Y. Yang and Dr. ir. S.J. Oude Elberink for their guidance and warm encourage during my thesis, I cannot complete thesis without your support. I also want to express my sincere thanks to all GFM teachers and colleagues, your care and help let me experience a challenging and wonderful academic life here.

I want to particularly thank my friends zhengchao zhang, yaping lin and keke song to give me an insight of the deep learning and constructive suggestion.

Additionally, I would also thank to all staffs in ITC and ITC hotel for providing me fun ambiance and good service, let me experience a period of Dutch life.

Finally, I am very grateful to get accompany and encourage from my parents for helping me get through the confusion period and resist stress.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Motivation and problem statement

Metropolitan areas will be an important spatial form in modern cities in the new era, but they will also bring the new challenge to traditional urban plan, disaster management, and urban research. Detailed 3D city modelling can be widely used in all aspects, with the rapid development of techniques and Geographic Information Systems (GIS), it brings tangible and considerable benefits. In terms with an urban plan, for example, analysing the traffic flow, pedestrian patterns and land use can provide more useful, sustainable and healthy plan for the future development of a city (R. Chen, 2011). When facing the disaster, 3D city model shows its powerful function to quickly assess the damage, to guide helpers and to reconstruct the damaged sites because it delineates the shape and configuration of a city. For urban research, the amount of sunlight the building is exposed to is often used to assess the suitability of installing solar panel on the roof, the light energy received by the building can be estimated by the material of the roof, its orientation and the tilt, while the 3D city model can quickly obtain these information (Biljecki et al., 2015). This proves that the detailed 3D city model is highly demanded in urban development.

At present, there are lots of researches focusing on image-based 3D city modelling technology as it has the merit of easy data assessing, a lighter workload and lower cost. Large areas of airborne imagery have more object details rather than general remote sensing imagery such as satellite image, it encourages the application of model and the perception of location to understand the object-level scenes. Semantic segmentation of images with high resolution such as airborne image is one of the sub-tasks on the computer vision problem to build 3D city model. However, the interpretation of airborne images on urban scale became a barrier. Mainly because for the semantic segmentation in urban areas, objects of the urban scene are complex and small, many man-made objects are composed of different materials and interact with non-man-made objects though occlusion, shadows and inter-reflections. This results in high variability of image intensity within classes and low difference between classes, additionally, manually geometric modelling for large urban areas is time and cost inefficient (Döllner et al., 2015).

Semantic segmentation problem is more often regarded as a kind of supervised learning. A classifier learns to predict the conditional probabilities from the features of the image through some labelled training data. Images from different data sources can extract different features, the image pixel intensity, image texture and various filtering responses are most commonly used as input features [Leung & Malik, 2001, Schmid, 2001, Shotton et al., 2009]. For airborne urban images, neighbouring pixel values are high-degree spatial correlated, it is hard to determine its true class label. Usually before semantic segmentation, a large and redundant set of features are needed to do feature learning and let the classifier select optimal dataset, to hope to reduce the loss of relevant information by feature encoding in this way. However, this classifier is often guided by user-experience and controlled by a set of user-defined parameters which is suboptimal and non-exhaustive (Volpi & Tuia, 2017). The limitation of both parametric and non-parametric supervised image classification approaches promotes a new way to automatically extract and learn features for specific images like airborne images.

Recent advances in machine learning and deep learning are about to overcome this limitation and providing a high degree of automatic and semiautomatic processing of airborne images which has opened a door for

interpretation on a large city scale, both of it aim at training classifier to assign a label to each pixel based on features. The conventional machine learning algorithms are that as complex as they may seem, they are still machine-like. They need a lot of domain expertise to extract features, human intervention only capable of what they are designed for. Deep learning is a sub-field of machine learning which uses multi-level nonlinear information processing and abstraction for supervised or unsupervised feature learning, representation, classification and pattern recognition (Claesson & Hansson, 2014). And the most famous one in deep learning is the Convolutional Neural Networks (CNNs), current CNNs are generally composed of convolution layer, activation layer, pooling layer and fully connected layer (Figure 1). There are at least three structural characteristics of CNNs: locally connected, weight sharing and subsampling, these properties make the CNN invariant in translation, scaling and rotation to a certain extent. Here, deep conventional neural networks (DCNNs) which stacks deeper layers follow the same order, capturing neighbouring information by convolutional filters are trained in an end-to-end, pixel-to-pixel manner and delivering strikingly better results than systems that rely on hand-crafted features, it has exceeded the state-of-the-art in semantic segmentation and pushed the performance of computer vision system to soaring heights on a range of high-level performance.

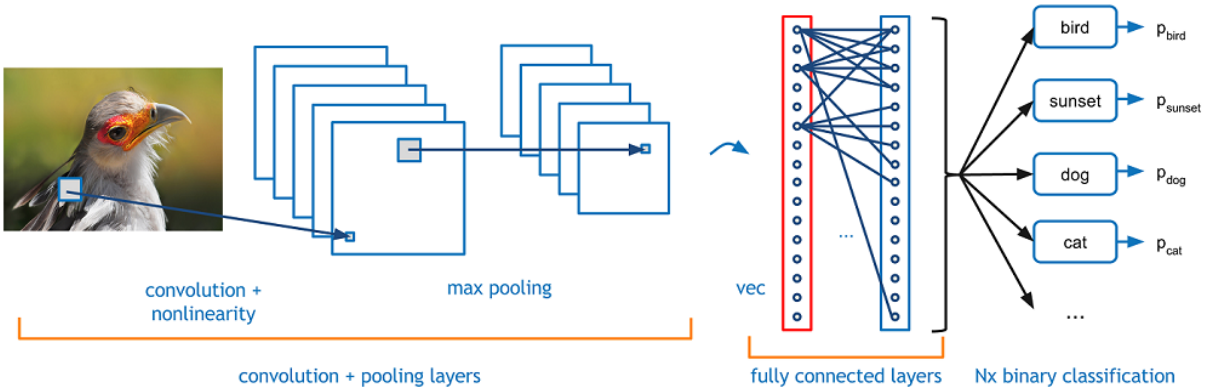


Figure 1. Simple CNN architecture for image classification[1].

Classical CNN structures are suitable for image-level classification and regression tasks because they all end up expecting a numerical description (probability) of the entire input image, then the input image is labelled as class with highest probability. Different from the previous application, semantic segmentation is a dense classification (pixel-level) task, in the past, for the CNN used for semantic segmentation, taking an image block for each pixel and then constantly sliding the window, each sliding window classifies CNN. Each pixel was marked by the object or area category surrounding it, however, this method has great defects in both speed and accuracy (Long, Shelhamer, & Darrell, 2015). Long et al., (2015) proposed Fully Convolutional Neural Network (FCN) which replace fully connected layers with convolutional layers, restoring the class of each pixel from the abstract features so that lift classification problem from image-level to pixel-level classification.

Another way to solve the fully connected structure and the aggregated background discards some location information problem is using encoder-decoder architecture. In encoding process, the continuous pooling operation gradually reduce the spatial dimension of the input data, while the detail information and the corresponding spatial dimension will be recovered during the decoding process. Directly link and combine the information of encoder and decoder contributes to recover the target details. U-net is a typical network

[1] http://code.flickr.net/

of this kind of way, U-net (Ronneberger, Fischer, & Brox, 2015) was used for biomedical image segmentation at the earliest. Different from point-by-point addition of FCN, U-net adopts to concatenate features together in channel dimension to form "thicker" features, in this way, more features can be acquired without increasing the number of training sets, it is more suitable for small data sets.

However, the continuously developed combination of max-pooling and down-sampling has a toll on localization accuracy so that to coarsen the outputs and object boundaries (L.-C. Chen et al., 2018), and the relationship between pixels is not fully considered, spatial regularization used in the usual segmentation method based on pixel classification is ignored, lacking of spatial consistency. The probabilistic graphical model was proposed to combine with DCNNs to qualitatively and quantitively improve localization performance. Conditional random field (CRF) is regarded as one of the representative probabilistic graphical models on improving the classification results and breaking through the limitation of DCNNs by incorporating contextual information (such as the presence of edges, homogeneous image regions and texture) as a linear combination of pairwise energy potentials (Moser, Serpico, & Benediktsson, 2013), joining probability of the entire sequence of labels given the observation sequence (Lafferty, 2001). CRF is usually used as a kind of post-processing step which transforms the semantic segmentation problem into a Bayesian maximum posteriori (MAP) inference, to refine coarse pixel-level classification results of DCNN (Krähenbühl & Koltun, 2012a). There are several kinds of CRF, local CRFs like 4-connected and 8-connected CRFs only capture neighbouring information in confined areas. Fully connected CRF that considers both short-range and long-range interactions between neighbouring pixels can mitigate the blurry object boundaries (Figure 2). The response of the last DCNN layer is combined with a fully connected CRF to locate segment boundaries L.-C. Chen et al., (2014). Here, rough segmentation and fine segmentation are completely separated and are not an end-to-end training model. Zheng et al., (2015) regarded the iterative process of solving the reasoning of CRF as the correlation operation of Recurrent Neural Network (RNN) and embedded in the DCNN model to really achieve the fusion between algorithms.

In this study, various remote sensing image classification and RGB-oriented image segmentation methods are modified and integrated to the DCNN structure methodological framework to generate airborne urban image segmentation. In addition, this study also investigates the potential of using Recurrent Neural Network (RNN) to incorporate contextual neighbouring information by joining a kind of DCNN architecture into an end-to-end trainable network.

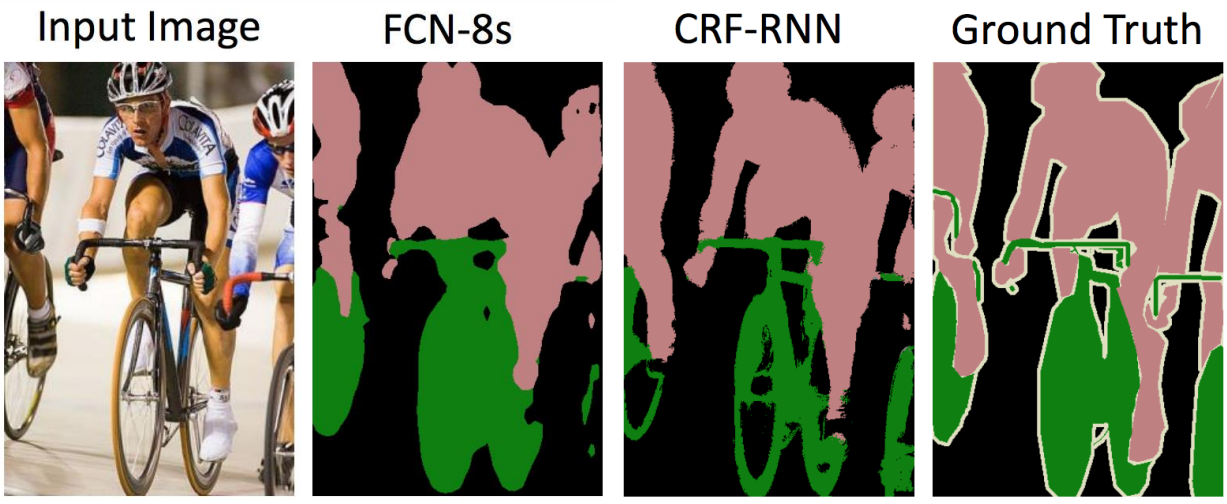


Figure 2. Example image and resulting segmentation from FCN-8s, and CRF-RNN coupled with FCN-8s (Zheng et al., 2015).

1.2. Research objectives

This study mainly focuses on semantic segmentation field, the applicability of DCNN to urban airborne images, and exploring the optimization of deep neural network with fully connected CRF model. The novelty of this study lies in treating the inference of the fully connected CRF as an RNN, joining with an arbitrary DCNN architecture to build an end-to-end classification model. The main objective can be divided into the several following sub-objectives:

1. Choose and train a DCNN classifier to segment airborne urban images.

2. Integrate fully connected CRF and DCNN, regarding the learning and inference process of CRF as RNN and then embedded in the U-net model, training an end-to-end model to further improve the segmentation accuracy.

3. Apply and assess the ability of fully connected CRF inference for refining the semantic segmentation of airborne oblique urban images (Krähenb et al, 2011).

1.2.1. Research questions

Sub-objective 1:

- How to make this DCNN adapt to the airborne oblique urban images in this study and what are optimal parameters in this classifier?
- What is the accuracy matrix of this classifier?
- How can the image resolution influence the classification results?

Sub-objective 2:

- How to apply the CRF-RNN model to the segmentation problem?
- How to construct this fully connected CRF? What are the parameters required for specifying the fully connected CRF model?

Sub-objective 3:

- What is the difference in segmentation accuracy between the DCNN after CRF-RNN fused and single DCNN?
- Which class can gain the most benefits from fully connected CRF and which class gain the least benefits?

1.2.2. Innovation aimed at

In recent years, structure prediction shows its importance in solving mismatched relationship errors, and most of these errors are partially or fully related to context and global information. This study experiments on a new airborne oblique urban dataset, modifying and applying a DCNN to the new dataset. We also furtherly implement a fully connected CRF to incorporate with DCNN and making the inference as RNN to obtain an end-to-end model.

1.3. Thesis structure

This thesis consists of seven chapters. Chapter 1 introduces the motivation of this thesis and problems that will be proposed in this research. Chapter 2 briefly reviews the current mainstream methods of scene interpretation. Chapter 3 focus on the methodology tried in this research. Chapter 4 describes the parameters optimization, experiment implement and showing how training dataset can influence the semantic segmentation results. Chapter 5 shows the semantic segmentation results comparing to ground truth. Chapter 6 discusses the advantages of method in this research and points out some failures and limitations. Chapter 7 gives a short conclusion on this research and potential ways can be explored in the future.

2. LITERATURE REVIEW

This chapter briefly review the mainstream and existing semantic segmentation approaches that are related to this research. Section 2.1 mainly reviews development and application of CNNs for image segmentation. Section 2.2 briefly introduce the theoretical foundations and learning of CNNs. Section 2.3 explains the application of CRF to take contextual information in semantic segmentation.

2.1. Convolutional Neural Network on image segmentation

At the early stage, the commonly concerned question of improving the semantic segmentation accuracy is efficient feature extraction, most of the early approaches focused on exploring more hand-engineered features. The scale-invariant feature transform (SIFT) is an image feature generation method which makes use of a staged filtering to transform an image into a large collection of local feature vectors (Lowe, 1999). A generally used classifier random forest often uses SIFT features combined with colour features to do scene interpretation. In addition, some methods suggest that region-based models are more likely to extract robust features by forcing the pixels of a region to have the same label and reduce the computational complexity, at the same time, it considers more context (Yu et al., 2018). What makes automatically interpret urban scene particularly challenging is the high variability of urban scene image intensity in class and the low difference between classes are not conducive to distinguish objects.

With the development of deep learning, learning-feature method received more and more attention. CNN is a well-known branch of deep learning method whose design is inspired by the natural visual perception mechanism of organisms. At the earliest, Lecun et al, (1998) developed a multi-layer neural network named LeNet-5 to classify handwriting numbers, LeNet-5 has multi-layers and can be trained using the back-propagation algorithm. However, it does not work well on large-scale image and more complex tasks such as video classification because of lacking extensive training data and computing power. In 2012, with the 8 layer convolutional neural network AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) won the "ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) of 2012" by a big margin, and the recognition error rate was about 10% lower than the second place. It proves the effectiveness of CNN under the complex model, then the GPU implementation makes the training get results within an acceptable time range. AlexNet used Rectified Linear Unit (ReLU) to be activation function to solve the gradient dispersion problem of sigmoid in deep networks, it performs much faster in gradient training time. Additionally, AlexNet replaced average pooling with max pooling to avoid blurring effect of average pooling (Krizhevsky et al., 2012). For the overfitting problem, AlexNet used dropout for the first two fully connected layers, the output of each hidden layer neuron is set to 0 with a certain probability, that is, some neurons are randomly ignored. Soon after, the appearance of GoogLeNet (Szegedy et al., 2014) and VGG (Simonyan & Zisserman, 2014) proved that with the increase of the depth, the network can better approximate the nonlinear increasing objective function and obtain better characteristic representation (Gu et al., 2015). As the network depth increases, network accuracy might to be saturated or even decrease, He et al., (2015) developed Residual Neural Network (ResNet) to solve degradation problem. For a stacked structure, when we have input x, the learned feature can be marked as $H(x)$, now we expect to learn the residual $F(x) = H(x) - x$, so that the original learning feature is $F(x) + x$. This is because residual learning is easier than direct learning of the original features. The proposal of deep residual network became a milestone event in the history of CNN image.

CNNs train end-to-end, pixels-to-pixels themselves and exceed the state-of-the-art meanwhile. Typical CNNs require fixed-size images as input and output a single prediction for whole image classification. The powerful function of CNN is that its multi-layer structure can automatically learn both local features and abstract features, it helps to improve classification accuracy. Usually, the size of the pixel block is much smaller than that of the whole image, so caused limitation on perception field and only some local features can be extracted, which leads to the limitation of classification performance. Then, Long and Shelhamer., (2015) formulated a fully convolutional network (FCN) that takes the input of arbitrary images and produced the same size of output with efficient inference and learning. To achieve a significant improvement on semantic segmentation, they defined a skip architecture which combines the deep and coarse semantic information together with shallow and fine appearance information in multiple layers, this kind of multi-layer structure fusion constitutes a nonlinear semantic pyramid from local to global. For remote sensing images of this kind of super-large image, a FCN divides the whole image into patches of the same size and trained on all overlapping patches to produce the best results at that time (Long et al., 2015).

Using semantic segmentation on urban scene interpretation is a pretty recent field of investigation. At first, the dataset used for urban image segmentation is usually of very high resolution (<10cm). For these high resolution images, they have inherently lower spectral resolution but small objects and small-scale surface texture become visible which tends to bring more noise (Marmanis et al., 2016), therefore, different from low resolution images such as the often-used Landsat and SPOT satellite data which has unmatched greater geometric details, in contrast, the spectral resolution of high resolution images like airborne images is much lower, and the spectral characteristics are very few, basically only RGB characteristics and near-infrared feature is occasionally added (Dimitris Marmanis et al., 2016). Nguyen et al. (2017) employed random forest (RF) and a fully connected conditional random field (CRF) to do semantic segmentation from urban airborne imagery, the base-case accuracy reached 82.9% on dataset released by ISPRS WG III/4 for the Urban classification test project [1], the GSD of both, the TOP and the DSM, is 9cm. "Semantic segmentation of aerial images with an ensemble of CNNs" proposed by Marmanis et al. (2016) achieved accuracy of 88.5% on Vaihingen data set of the ISPRS 2D semantic labelling contest which has GSD of 9cm. Sherrah (2016) proposed an FCN without down-sampling which preserves the full input image resolution at every layer on high-resolution remote sensing data, for high resolution detail images, the classification accuracy is significantly improved. Later, Zhao et al., (2017) proposed Pyramid Scene Parsing Network (PSPNet) to avoid context information loss between different sub-regions, the hierarchical global prior contains information with different scales different sub-regions, experiment on cityscapes dataset outputted good result.

Overall, currently DCNN is becoming mainstream method on urban scene interpretation, most of dataset are city street view images and aerial images with very high resolution, urban airborne oblique images are seldom used for urban scene understanding. This study explores the potentials of urban airborne oblique images on semantic segmentation problem with CRF model and DCNN.

2.2. A brief overview of Convolutional Neural Network

Neural Network: A neural network consists of a large number of neurons connected to each other. After each neuron receives the input of linear combination, it is just a simple linear weighting at the beginning, and then a nonlinear activation function is added to each neuron, so as to carry out the output after the nonlinear transformation. The weight value which links each neuron can influence the output of a neural network, the different combination of weights and activation functions will lead different neural network outputs. The input of the neural network consists of a group of input neurons which is activated by the

input image pixel and nonlinear transformation. After the neurons are activated, they are converted to other neurons, repeating the whole process until the last output neuron is activated. A single neuron of the neural network is as Figure 3:

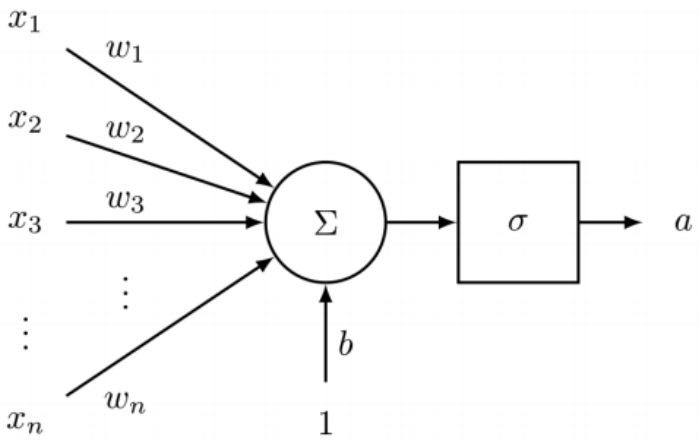


Figure 3. Schematic representation of a neuron in neural network[2].

Given an input vector $X = \{x_1, x_2, ..., x_n\}$ and performs a dot-product operation with a vector of weights $W = \{w_1, w_2, ..., w_n\}$, a bias term b is added to the product results and then the output passes through an activation function, the process can be mathematically defined as:

$$a = \sigma(W \cdot X + b) \qquad \text{Equation 1}$$

Where σ is the activation function, the weight vector W and the bias term b together constitute the parameters of the control output value of a.

When individual neurons are organized together, a neural network is formed. At the same time, each layer may be composed of one or more neurons, and the output of each layer will serve as the input data of the next layer. For the hidden layer in the middle, the neurons $a_1, a_2, a_3, ..., a_n$ receive input from multiple different weights (because the inputs $x_1, x_2, x_3, ..., x_n$, the neurons will accept the weight respectively, that is, the number of inputs equals to the number of weights). Then, $a_1, a_2, a_3, ..., a_n$ become the input of the output layer under the influence of their respective weights, and finally output the final result from the output layer, the process can be defined as:

$$a = \sigma(n) = \sigma\left(\sum_{i=1}^{n} x_i w_i\right) = \sigma\left((w_1, w_2, ..., w_n)\begin{pmatrix} x_1 \\ x_2 \\ \bullet \\ \bullet \\ \bullet \\ x_n \end{pmatrix} + b\right) = \sigma\left(W^T \cdot X + b\right) \qquad \text{Equation 2}$$

[2] https://wugh.github.io/

A neural network is formed by the regular combination of neurons, Figure 4 shows a fully connected neural network, the regulation can be briefly summarized as the following points by observing the diagram:

- Neurons are laid out in layers. The first layer is called the input layer and is used to recieve input data; the last layer is called the output layer, from which we can obtain the output data of the neural network.
- All the layers between the input and output layers are invisible to the outside world, so these layers are called hidden layers.
- In the same layer, there are no connections between each neuron.
- Each neuron in each layer is connected to all neurons in the previous or next layer so called "fully connected", the output of each layer of neurons is the input of the next layer of neurons.
- Each connection of different layers has a weight.

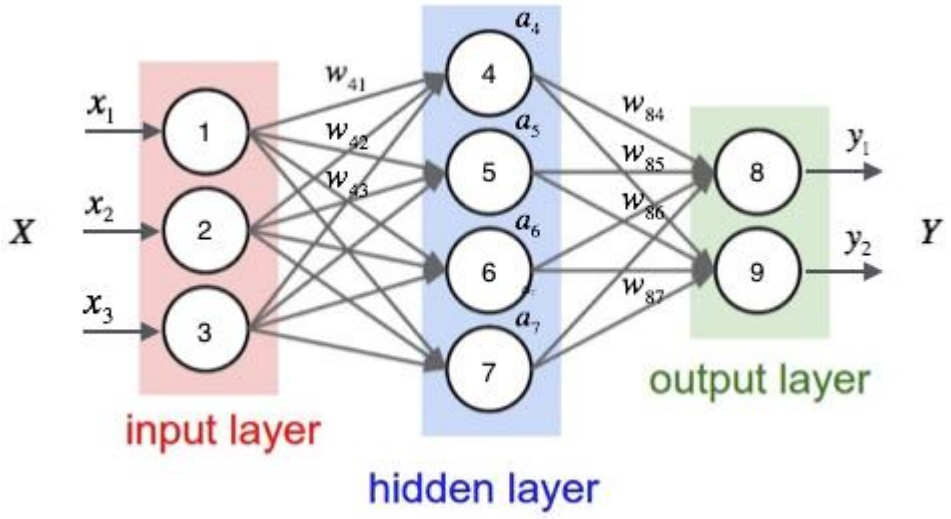


Figure 4. Simple structure of single layer neural network (Tang & Yiu, 2018).

Inside the structure of a neural network, each neuron is called a node. For instance, in Figure 4, in order to calculate the output value of node 4, the output value of all its upstream nodes (i.e., node 1, 2 and 3) must be first obtained. Nodes 1, 2 and 3 are nodes in the input layer, so the output values are the input vectors $\boldsymbol{X}$ themselves. The dimension of the input vector is the same as the number of neurons in the input layer, it is free to say which input node one of the input vector goes to.

For the application of neural network in computer vision, the key point is that to know the weight values on each connection of a neural network. It can be said that the neural network is a model, so these weights are the parameters of the model, which is what the model needs to learn. Nevertheless, the connection mode of a neural network, the number of layers in the network and the number of nodes in each layer are not learned, but, setting artificially in advance. For these manual set parameters, we call them hyper-parameters. The learning of neural network is to study how to make these parameters "fit" with the training set. In fact, it is to obtain the values of these parameters. The most commonly used neural network learning method is backpropagation algorithm (BP algorithm).

In the forward propagation process, the error between the output value suffered from neural network operation and the actual value can be defined by a cost function J, the cost function is the average of the sum of all the sample errors. The error of a single sample can be expressed by a loss function L, so for

classification problem, error is the difference between the predicted category and the actual category. The above mentioned "fitting" is to minimize the errors between the output and the actual value, that is minimizing the cost function J . The cost function can be quantified as follows:

$$L(x,c)=\sum_{i=1}^{N}L(x_i,c_i) \quad \text{Equation 3}$$

Where x is the image data which has a 4 dimensional structure: $H\times W\times F\times N$, H , W , F and N respectively represent the height, width of image and the number of feature channels of the input image, for example, a RGB image has three spectral features so the value of F is 3, while N represents the number of such 3 dimensional image that together make up the contents of a single input batch. x_i represents a vector of class-probability scores for all pixels of the input image batch, c_i represents a reference label vectors of ground truth (true label).

The forward propagation phase is the phase in which data is propagated from low level to high level. When the results obtained from the current forward propagation are not consistent with the expected results, the error is carried out in the stage of propagation training from the high level to the low level, that is, the stage of backpropagation. The point of backpropagation is to learn weights, it is the application of the chain rule. Weight is learned by calculating the cumulative loss with respect to the partial derivative of each parameter weight, this weight is updated throughout the backpropagation process and it can be represented as:

$$w_i=w_i+\lambda\frac{\partial L}{\partial w_i} \quad \text{Equation 4}$$

Where λ is the learning rate of weights.

So far, we have derived the backpropagation algorithm. It should be noted that the training rules are based on the activation function, fully connected network and stochastic gradient descent optimization algorithm. CNN is the most widely used deep learning method in image classification problem, a more detailed overview of CNN is explained below, passing the image to a series of convolution, non-linearity, convergence (down-sampling) and fully connected layers and get the output.

Convolution neural network: The name for the convolutional neural network comes from the operation convolution. In convolutional neural networks, the main purpose of convolution is to extract features from input images. By using small squares in input data to learn image features, convolution preserves the spatial relationship between pixels. Each image can be viewed as a pixel value matrix, consider $m\times m$ image with pixel values matrix, considering another $n\times n$ matrix with pixel value of only 0 and 1, when applying convolution operation of this $n\times n$ matrix to image pixel value matrix, another pixel value matrix will be derived. This $n\times n$ matrix is called a "convolution kernel" or "filter", the matrix obtained by moving the filter over the image and computing the dot product is called the "convolution feature" or "activation map" or "feature map", the filter is a feature detector for the original input image. For the same input image, different filter matrices will produce different feature maps. Simply change the value of the filter matrix before the convolution operation to perform different operations such as edge detection, sharpening and blurring, which means that the various of filters can be used to extract different features of the image, such as edges and curves.

In fact, different from traditional machine learning method, the convolutional neural network learns the values of these filters by itself, but the number of filters, the size and the network framework must be

artificially specified. In general, the more filters used in network, the more features can be extracted, and the better the network is at recognizing new images. The size of the feature map (the convolution feature) is controlled by three parameters, which need to be determined before performing the convolution step: depth, stride and zero padding. The number of filters in the convolution operation is the depth, the stride is the number of pixels that move the filter matrix once over the input matrix, the larger the stride, the smaller the feature mapping. The function of zero padding is to control the size of feature map, so that we can apply filters to the boundary elements of the input image matrix.

Non-Linear Activations: In a multi-layered and complex network, the output of each layer is a linear function of the input of the upper layer, so the output of the neural network is a linear function no matter how many layers there are, this linearity is not conducive to play the advantages of neural networks. So the main purpose of introducing nonlinear activation function is to increase the nonlinearity of neural network.

Sigmoid is a representative non-linear activation function, it is also known as the logistic function which has value range from 0 to 1. The activation function has a large amount of calculation, and when the error gradient is obtained by back propagation, the derivation involves division. In the case of back propagation, the gradient will easily disappear, thus failing to complete the training of deep network, that is "gradient vanishing" (Volpi & Tuia, 2017). Rectified Linear Units (ReLU) is an operation on an element (applied per pixel) and all the negative value of pixel in feature map will be replaced with 0. The ReLU function has good linearity in parts greater than or less than zero, which is convenient for derivation. On the other hand, the ReLU function has a simple structure and is conducive to the forward inference. Therefore, both the training process and the test process have been greatly reduced. Since the positive interval of the ReLU function is unsaturated, the gradient loss problem is slowed down (the positive samples are considered more). Therefore, ReLU is currently and widely used as a non-linear activation function.

Maximum or Average Pooling: Spatial pooling is used to reduce the dimensions of each feature map and retains the most important information, reduces the number of parameters and operations in the network, it makes the network more robust to the tiny transformation, distortion and translation of the input image. There are several different ways to pool space: maximum, average, sum, etc. In the case of maximum pooling, a spatial neighbourhood can be defined. The maximum pooling operation takes the largest element of the modified feature map in the window, the average of all the elements in the window (average pooling) or the sum of all the elements also can be taken. In practice, the maximum pooling performs better.

Dropout: In the deep learning model, if there are too many parameters of the model and too few training samples, the model will have a small loss function on the training data and a high prediction accuracy, however the loss function is large in test data and the testing accuracy is low, this phenomenon is "overfitting". Therefore, Hinton et al., (2012) proposed Dropout operation to avoid overfitting. During the forward propagation process, the activation value of a neuron is made to stop working with a certain probability p, which can make the model more generalized, because it is not too dependent on some local features (Krizhevsky, Sutskever, & Hinton, 2012).

Fully connected layer: The fully connected layer uses the softmax activation in the output layer, it is essentially a multi-layer perception. In this layer, each neuron in the previous layer is connected to each neuron in the next layer. The output of the convolutional layer and the pooling layer represent the advanced features of the input image. The purpose of the fully connected layer is to divide the input images into different classes by using the features obtained based on the training data set. The fully connected layer maps the learned "distributed feature representation" into the sample label space. In practice, the fully connected layer can be implemented by convolution operation. Because softmax activation is used in the

output layer of the full connection layer, softmax takes any real vector as input and compresses it to a vector whose value is between 0 and 1 and whose sum is 1, the sum of the output probabilities of the fully connected layer is 1.

2.3. Conditional random field

Although deep learning achieved a significant improvement on semantic segmentation, however, there still exists limitations. Separately classifying pixels based on local features could easily cause inconsistency and noise in the results because there is usually some correlation between pixels. Hence, lots of strategies pay much attention to inconsistency problems. A more notable way to overcome the spatial consistency is to explicitly model the relations between pixels or regions, contextual model: Markov random field (MRF) and CRF are the representatives. In the MRF framework, it refines the classification results by modelling the joint distribution between observations and label. Such a generative model suffers from some defects when applying to image processing. Pixels are generally considered to be strictly independent and equally distributed in MRF, moreover, even the class posterior is simple, model inference can be quite complex (Yu et al., 2018). As a result, MRF only achieves to capture the spatial dependency in label space, the observation from image still needs to explore.

CRF is a discriminative model which modelling probability of labels depends on given observed data. It is usually difficult to describe natural image pixels and its label distribution with a simple model, so CRF shows its high computation efficiency because there is no need to consider the observation variable and label variable distribution compared to MRF. In most commonly used CRF model, the pairwise potential is often used to build penalizing function to estimate the difference between pixels. However, the context in which this pairwise potential is utilized is very limited because adjacent pixels tend to have the same label. Some works concentrate on encoding contextual information to construct the unary potential. Shotton et al., (2006) developed a discriminative model which exploits novel features based on textons, jointly model shape, appearance, and context. Using boosting to train an efficient classifier achieves unary classification and feature selection. Vezhnevets et al., (2012) formulated semantic segmentation as a pairwise CRF and consider a parametric family of CRF models to give different mixing weights to different visual similarity metrics between super-pixels. Nevertheless, this model is limited in some objects that may be present in a range at the same time. This context is very limited, so the dependencies between classes are underutilized and global contexts are not taken into account, this leads to aggravating the oversmoothed boundaries of objects. For capturing the long-range dependencies to overcome the oversmoothed problem, Kohli et al., (2009) proposed a robust P^n Potts model as high order potentials on the image segments. He et al., (2004) used multiscale CRF which fused the information from local and global scales and combined it in a probabilistic manner, their result outperformed traditional hidden Markov model labelling of text feature sequences. Hierarchical CRF (Ladicky et al., 2009) is also utilized in exploiting multi-level contexts to refine the object boundaries. It is easy to incorporate various constraints based on higher-order potentials, but it is also possible to produce misleading boundaries.

Here, fully connected CRF avoid this problem by considering interactions between long-range pixels not limited to adjacent pixels. A lot of methods have incorporated CRF into DCNNs for improving the classification accuracy on boundaries. L.-C. Chen et al., (2018) trained a DCNN as a unary term of fully connected CRF to obtain sharp boundaries. In this model, all nodes are connected in pairs, the efficient interference that represents pairwise terms by the linear combination of Gaussian kernels makes the fully connected CRF tractable (Krähenbühl & Koltun, 2012b). This fully connected CRF can incorporate with DCNNs to solve the coarse labelling at the pixel level.

So far, CRF is widely used as post-processing of other classifiers, output from the other classifier is used as input of CRF, this mode of operation is a one-way propagation. Errors in the previous classifier are optimized and the classification results are passed into the post-processing operation, but the feedback of CRF cannot be back propagated to the previous classifier, this makes the model which combines the two architecture does not play out its most efficiency and advantage. Zheng et al., (2015) formulated "mean-field approximate inference for the dense CRF with Gaussian pairwise potentials as a Recurrent Neural Network (RNN)", to achieve an end-to-end deep learning solution which combines the strengths of both CNN and CRF. Their experiment outperformed a structure that CRF is often applied as a post-processing method to refine the output of other classifiers.

3. METHOD

In this chapter, we mainly introduce key methods. In various deep convolutional neural network frameworks, we adopt U-net to be this classifier to do semantic segmentation and provide unary potential for CRF. Section 3.1 explains how the U-net is used to do semantic segmentation to get unary term for CRF. Section 3.2 introduces the structure of CRF and how it refines the output of U-net. Section 3.3 explains the quality assessment. Figure 5 demonstrates the experiment workflow in this study.

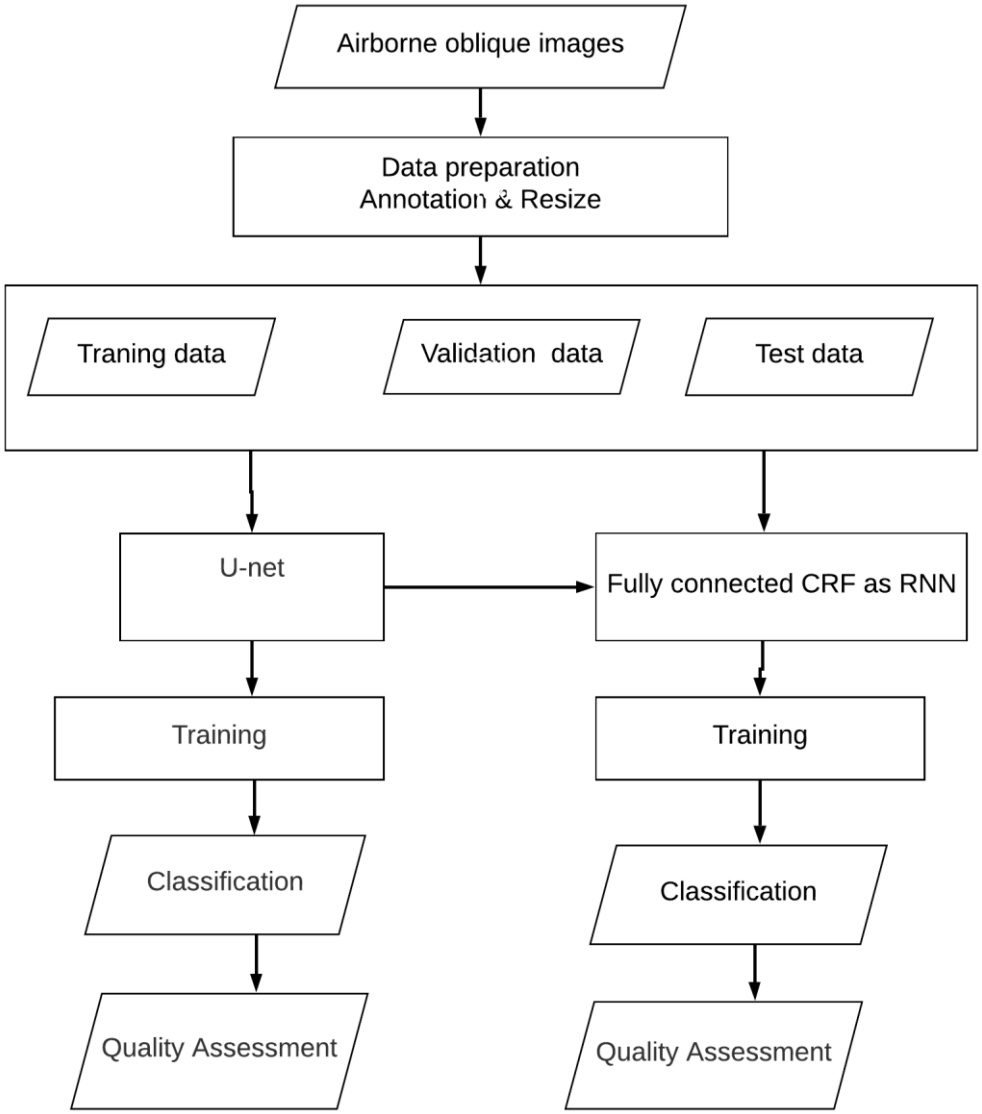


Figure 5. Workflow.

3.1. Deep convolutional neural network

As we stated in chapter 1, certain DCNNs have made significant contributions to semantic segmentation field. Such as AlexNet which is presented by Krizhevsky et al. (2012), won the ILSVRC-2012 with a TOP-5 test accuracy of 84.6% compared its closest competitor which made use of traditional techniques instead of deep networks, achieved a 73.8% accuracy in the same challenge. Later, on the ImageNet Large Scale

Visual Recognition Challenge (ILSVRC)-2013, University of Oxford proposed a famous CNN model named Visual Geometry Group 16 (VGG-16) which achieved 92.7% TOP-5 test accuracy due to its 16 weight layers configuration. It makes use of the convolution layer with a small receptive field on the first layer, rather than the convolution layer with a large receptive field, so that the intermediate parameters are fewer and the non-linearity is stronger, the decision function, it makes the decision function to be more discriminative and the model easier to train (Garcia-Garcia et al., 2018). In addition to this, GoogLeNet (Szegedy et al., 2014) is composed by 22 layers and proved that CNN layers could be stacked in more ways to form different network frameworks, this complex architecture won the ILSVRC-2014 challenge with a TOP-5 test accuracy of 93.3%. Another remarkable network ResNet-152 (K. He et al., 2015) won the ILSVRC-2016 with 96.4% accuracy by introducing identity skip connections in its 152 layers so that layers can copy their inputs to the next layer. These classical deep neural network structures have been currently widely used in the construction of many segmentation architectures.

3.1.1. Architecture of the U-net

While convolutional networks have rapidly developed, the network training relies on a large number of training data, the size of the existing training set limits its development (Shotton et al., 2009), mainly because of labelling the ground truth of airborne images is very time-consuming, now deeper and deeper neural networks have up to million-level parameters to train, that requires a fairly huge dataset, otherwise it is easy to cause overfitting problem. On the other hand, the continuously developed combination of max-pooling and down-sampling in DCNNs has a toll on localization accuracy so that to coarsen the outputs and object boundaries (Chen et al., 2016). Therefore, semantic segmentation using deep convolutional neural network becomes more difficult. At the same time, in many visual tasks, the desired output of biomedical images should include localization information, which is similar to the semantic segmentation task of general images. Moreover, thousands of training images are usually beyond reach in biomedical tasks. Inspired by this, Ronneberger et al., (2015) proposed U-net which modified and extend fully convolutional network (FCN) such that it works with very few training images and yield precise segmentations as well. We decided to use the existing convolutional neural network model for image segmentation instead of developing a model from scratch. In other words, we adopt U-net which was originally developed for biomedical image segmentation task. The network can predict pixel-level binary classification with good accuracy. U-net is basically based on the Fully connected Convolutional Network (FCN), it extracts features of different levels through convolution sequence, Rectified Linear Unit (ReLU) activation function and max pooling operation so as to capture the context of each pixel.

The characteristic of U-net is that the contraction network and the expansion network are mutually mapped, it is a typical encoding and decoding network structure. In the process of decoding, the lost boundary information can be completed by merging the contraction layer features of the mapping to improve the accuracy of the predicted edge information. U-net uses skip-connections between the encoding process and the decoding process to precisely localize and capture context. Thus, the encoding process consists of a series of deconvolutions which concatenates features with the corresponding features from encoding process, and then followed by ReLU activation. By doing so, the thickness of the feature channel is doubled. Moreover, the flexible architecture of U-net makes it easy to freely deepen the network structure based on experiment data, for example, when dealing with objects with large receptive field. A figure of the U-net taken from Ronneberger et al., (2015) is presented below:

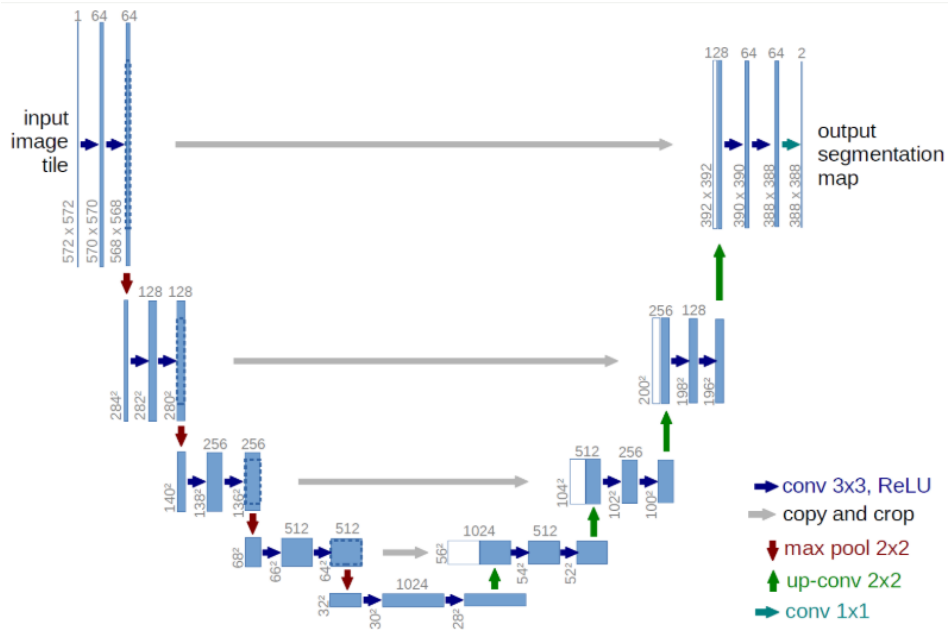


Figure 6. Original U-net architecture for biomedical image (Ronneberger et al., 2015)

3.1.2. Modification

For this urban airborne oblique image segmentation problematic, we choose to use a slightly modified version of the U-Net which is a version implemented by *George Seif*. We end up not using the pre-trained weights to initialize the network since these weights were trained on synthetic data. So we train the model from scratch and make some changes to the original architecture. Firstly, the input image size of original U-net is 572 × 572 × 3, the size of input image should adapt to the data used in this experiment.

The modified U-net has an encoder of a relatively simple CNN of the VGG family that consists of 16 sequential layers and known as VGG16, this kind of structure is improved based on "U-net with VGG11 Encoder pre-trained on ImageNet for image segmentation" (Iglovikov et al., 2018). The original intention of VGG in studying convolutional network depth was to find out how the depth of convolutional network affects the accuracy of large-scale image classification and recognition problem. VGG has 3 fully connected layers, according to the sum of convolutional layers and fully connected layers, VGG could be divided to different networks from VGG11 to VGG19, the differences between different VGG networks can be found in Figure 7. Simonyan et al., (2014) verified that deepening the depth of network is beneficial to improving the network performance. However, VGG also has its own limitations and cannot deepen the network without limit. When the network is deepened to a certain level, the training effect will fade, gradient fading or gradient explosion will occur. There is no obvious accuracy gap between VGG16 and VGG19 from the test results of Simonyan et al., (2014). Therefore, we choose VGG16 structure to be encoder of our U-net.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 7. VGG configurations (show in columns). The depth of the configurations increases from the left (A) to the right (E), the added layers are marked with blod. The convolutional layer parameters are named as "conv(kernel size)-(number of channels)", "FC" means fully connected layer, the last "FC-1000" means the output has 1000 classes (Simonyan et al., 2014).

VGG16 has 13 convolutional layers, all convolutional kernel size is 3 × 3 and each layer followed by a ReLU activation function. It also contains 5 pooling operations, each reducing feature map by 2. Along with the deepening of network, after each max pooling operation, the number of channels double, the number of channels increases to 512 at last. In the lower layer, the number of channels keeps same. In the decoder section, pooling operations were replaced with transpose convolutions layers, which doubles the size of the feature map and reduces half of the number of channels. Then concatenate the output of transpose convolution in each layer with its corresponding output of encoder. The feature map obtained by convolution operation is consistent with the number of channels in the symmetric encoder item, the up-sampling process is repeated 5 times to respectively correspond to 5 max pooling layers, each image is down-sampled twice because the current network implementation only accepts the input image size that can be divided by 32. Figure 8 shows the comparison between U-net with VGG11 encoder and U-net with VGG16 encoder.

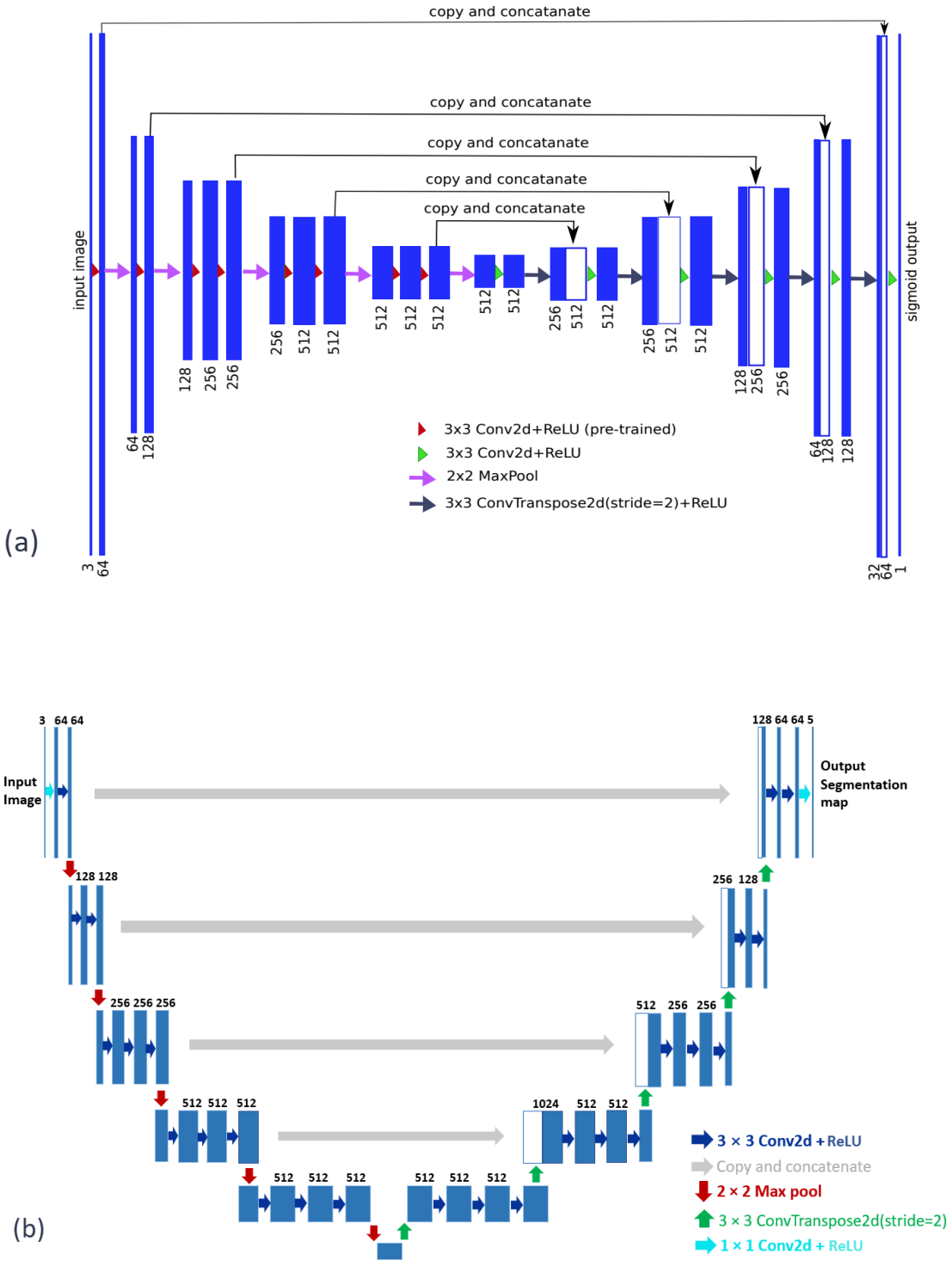


Figure 8. U-net architecture based on (a) VGG11 (Iglovikov et al., 2018), (b) VGG16 which is used in this study

3.1.3. Mobilenets

DCNNs have been widely used in the field of computer vison and achieved great results. The network depth is getting deeper and deeper and the model complexity is higher and higher in order to purse classification accuracy. For example, the depth residual network (ResNet) has as many as 152 layers. While in some real application scenarios, such a large and complex model is difficult to applied. In terms of this issue, recently Google proposed a small and efficient CNN model named Mobilenet which compromises between accuracy and latency. The elementary unit of Mobilenet is depthwise separable convolution which is a kind of deep level separable decomposable convolution: factorized convolutions. It can be separated to two smaller convolutional operations: depthwise convolution and a 1×1 pointwise convolution. A standard convolution filter converts inputs into a new set of outputs in a convolution operation, while separable convolution divides it into two steps: Depthwise convolution filter for each input channel and then followed by a 1×1 pointwise convolution to combine the output of depthwise convolution (Howard et al., 2017). This kind of factorization has drastically reduced computation and model size, this concise operation can be efficiently implemented with different kind of networks. Figure 9(a) shows a standard convolution diagram, Figure 9(b) shows the standard convolution is factorized into a depthwise convolution and Figure 9(c) shows a 1×1 pointwise convolution diagram, Figure 10 shows the Depthwise Separable Covolution implements with a DCNN block.

Assuming a standard convolution layer takes a $D_W \times D_H \times M$ feature map **F** as input and produces a $D_W \times D_H \times N$ feature map **G**, where D_W and D_H are the spatial width and height of input data, M represents the number of input feature channels (input depth). We assume that the output feature map has the same spatial dimensions as the input, so D_W and D_H are respectively the spatial width and height of output feature map, N is the number of output feature channels (output depth). So the dimension size of a standard convolutional layer is $D_K \times D_K \times M \times N$ where K is convolution kernel, D_K is the spatial dimension of the kernel, assuming that the stride is 1 and padding is 0, the output feature map of a standard convolution can be calculated by:

$$G_{k,l,n} = \sum_{i,j,m} K_{i,j,m,n} \cdot F_{k+i-1,l+j-1,m} \qquad \text{Equation 5}$$

The computational cost of standard convolution is:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_W \cdot D_H \qquad \text{Equation 6}$$

It is obvious that the computational cost depends on the kernel size D_K, input feature has M channels, the number of output feature channels N and the feature map size $D_W \cdot D_H$.

For this separable convolution, depthwise convolution is used to filter each input channel, then a 1×1 pointwise convolution is used to linearly combine the output of depthwise layers. Depthwise convolution with one filter per input channel can be presented as:

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m} \qquad \text{Equation 7}$$

Here, $\hat{K}$ represents the depthwise convolutional which has kernel of size $D_K \cdot D_K \cdot M$, and the m_{th} filter in $\hat{K}$ is applied to the m_{th} channel in **F** to produce the m_{th} channel of the filtered output feature map $\hat{G}$.

The computational cost of depthwise convolution is:

$$D_K \cdot D_K \cdot M \cdot D_W \cdot D_H \qquad \text{Equation 8}$$

The depthwise convolution only filters input feature channel but it does not combine them to produce new features, as we introduced previously, 1×1 pointwise convolution is used to combine depthwise convolution layers to create new features, so the cost of depthwise separable convolution is calculated by adding the cost of depthwise and pointwise convolutions.

The cost of depthwise separable convolution can be shown as:

$$D_K \cdot D_K \cdot M \cdot D_W \cdot D_H + M \cdot N \cdot D_W \cdot D_H \qquad \text{Equation 9}$$

Thus, the reduction in computation of this depthwise separable convolution is:

$$\frac{D_K \cdot D_K \cdot M \cdot D_W \cdot D_H + M \cdot N \cdot D_W \cdot D_H}{D_K \cdot D_K \cdot M \cdot N \cdot D_W \cdot D_H}$$

$$= \frac{1}{N} + \frac{1}{D^2{}_K} \qquad \text{Equation 10}$$

From equation 10 we can roughly estimate that the Mobilenet uses 8 to 9 times as much computation as standard convolutions with only a small sacrifice in accuracy.

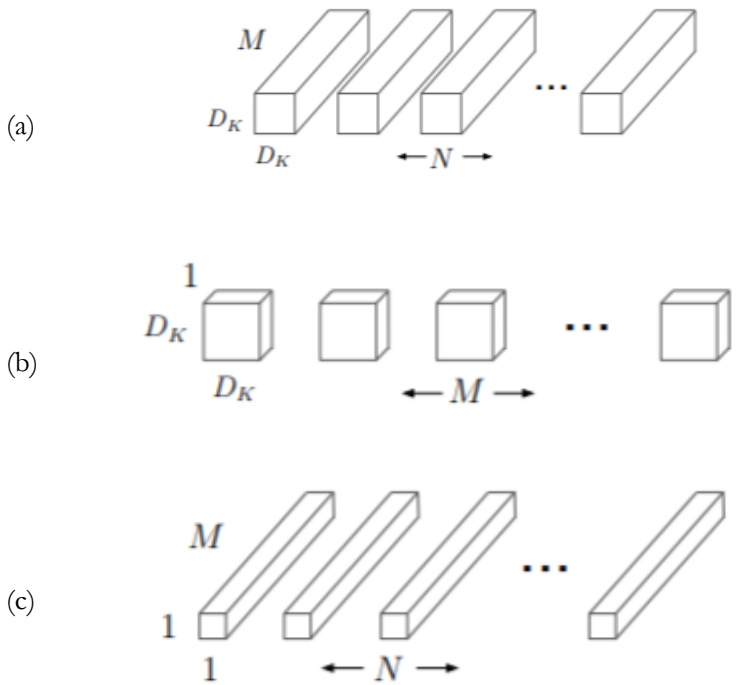


Figure 9. (a) is standard convolution filter (b) is depthwise convolution filter (c) is 1×1 pointwise convolution in the context of Depthwise Separable Convolution (Howard et al., 2017).

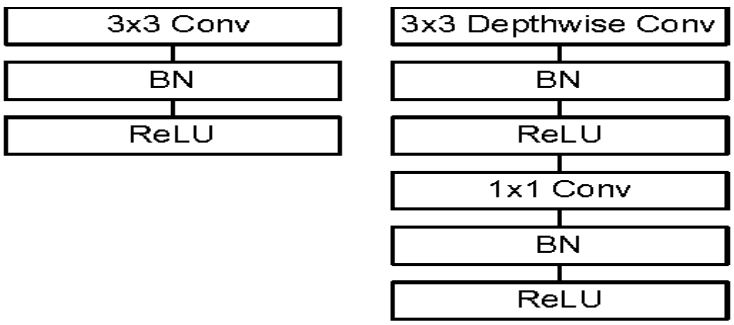


Figure 10. Left: standard convolution operation with Batchnorm and ReLU. Right: Depthwise Separable convolution contains Depthwise and Pointwise layers relatively followed by Batchnorm and ReLU (Howard et al., 2017)

3.1.4. Data augmentation

The premise that U-net works well in a limited number of training examples is extensive use of data augmentation because biological image is often impossible to have a large number of samples and labels. Generalization ability of deep complex neural networks relies on a large number of training data, otherwise, the weight parameters obtained by training cannot be well adapted to other datasets, resulting in overfitting problem at last. The dataset used in this study contains only 136 airborne images which is fairly small to such a complex network, because the network has nearly hundreds of thousands of parameters need to train, when the dataset is small, too many parameters will fit all the features of the dataset rather than the commonality between them.

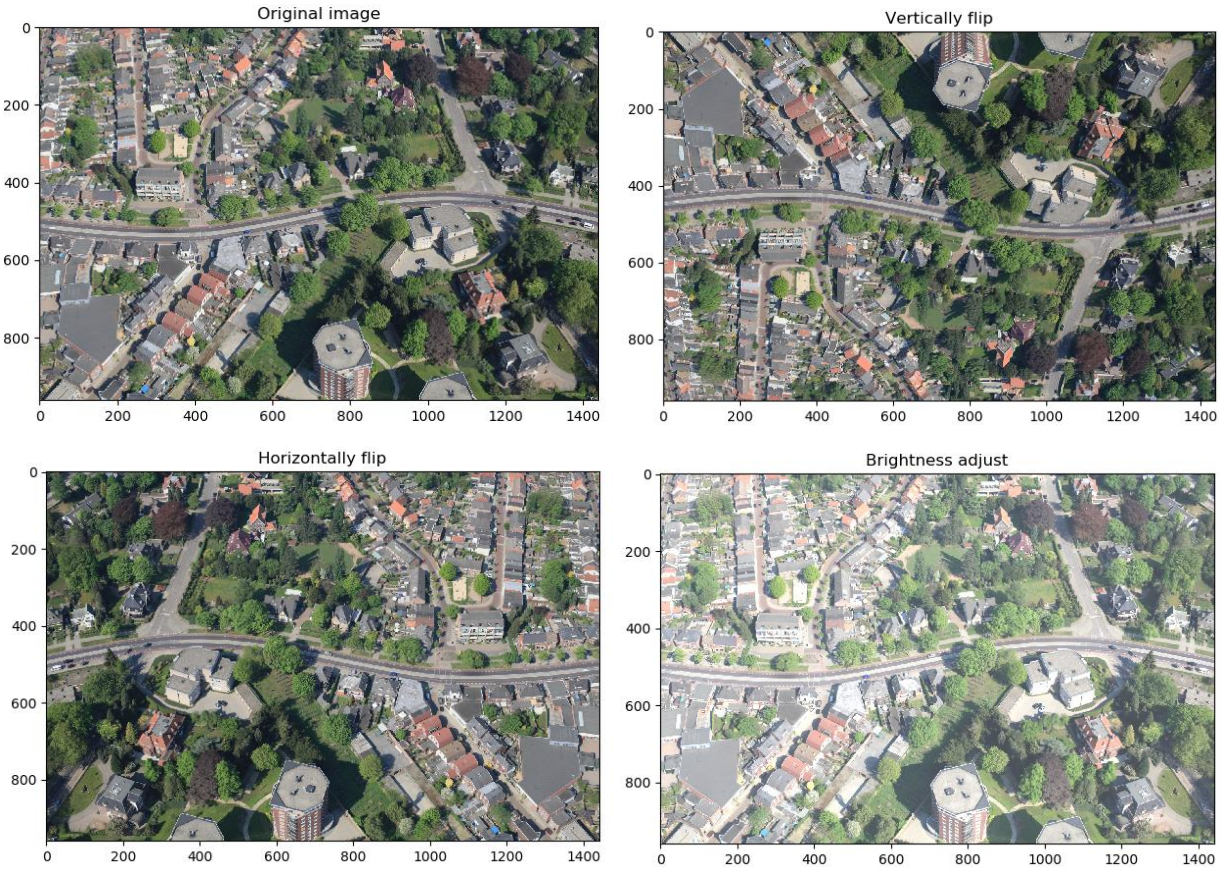


Figure 11. Several examples of data augmentation results

Data augmentation is also applied in this experiment to expand the number of images, solve the problem of data sparsity and to ensure sufficient invariance and robustness of the network. In data augmentation process, the training images are randomly shifted, horizontally and vertically flipped, rotated, and colour jittered that enrich the experiment data and make training data increased several times. In this study, data augmentation method was implemented by Tensorflow framework, this framework allows us to augment data in real time while batch feeding the network without consuming memory processes.

3.1.5. Training

The training or learning of the so-called neural network mainly aims to obtain the value of parameters that is needed by the neural network, in order to solve the specific problem. The parameters here include the connection weight and bias between neurons in each layer. The working process of DCNNs can be divided to forward propagation and backward propagation. In forward propagation, the input images provide the initial information, then propagate these information to the hidden units at each layer and finally produce the output according to the weight matrix and bias vector for the hidden layer. Then, a loss function can be defined based on the difference between the output and the true value of input images. Backward propagation is to calculate the partial derivative (gradient) of weight and bias parameters in each layer based on loss function to update the parameters.

The energy function is calculated from soft-max of the pixel direction in the final feature map combined with the cross entropy loss function, here, the soft-max is defined as:

$$p_k(X) = \frac{\exp\left(a_k(X)\right)}{\sum_{k'=1}^{K} \exp\left(a_{k'}(X)\right)} \qquad \text{Equation 11}$$

Where X is the pixel position of a two dimensional plane Ω, $a_k(X)$ is the output value of the last layer in the network in feature channel k, K is the number of class and $p_k(X)$ is the probability of pixel X belongs to class k.

The loss function uses negative cross entropy, cross entropy is presented as:

$$E = \sum_{X \in \Omega} \omega(X) \log\left(p_{l(X)}(X)\right) \qquad \text{Equation 12}$$

Where $p_l(X)$ represents the output probability of true label in its channel, in particular, a class weight term $\omega(X)$ is added to the cross entropy because of some class should be given more importance in the training. $\omega(X)$ is calculating by the statistics of the frequency of each class in the training dataset. The higher the frequency of the class, the lower the weight should be given, and the lower the frequency, the higher the weight should be given.

Using the input image to map its corresponding segmentation, the network is trained with the implemention of Tensorflow. We replaced the stochastic gradient descent (SGD) with the RMSProp Optimizer to speed up network convergency during training. The image data is highly correlated, when we do initialization, our parameters tend to be zero, so the initial fitting $y = \omega x + b$ is basically through the nearby original point (because of b is close to 0). Additionally, with the deepening of the network depth or in the process of training, the feature distribution is gradually offset or changed. Generally, the reason for slow training convergence is that the overall distribution is gradually close to the upper and lower limits of the value range

of the nonlinear function (Ioffe & Szegedy, 2015). Hence, the network needs to go through many times of learning to gradually reach the final fitting curve, that is, the convergence is very slow. Additionally, the learning process of neural network is to learn the distribution of data, when the test data has different distribution with training data, the generalization ability of the network greatly reduced; in the same way, if there is much difference between the distribution of single batch of training data (batch gradient descent), the network need to adapt the different distribution in each iteration, that will greatly reduce the training speed of the network. Thus, we also added Batch Normalization (Ioffe & Szegedy, 2015) after each ReLU activation to speed-up training. As a method of accelerating training, normalized input mainly has two steps:

$$\text{Zero mean: } \mu = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\text{Normalized variance: } \sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_i - \mu\right)^2$$

For the two-dimensional data, first set the mean value of each dimension to 0, and then normalize the variance of each dimension. After normalized input data, the cost function tends to be isotropic gradient, so that the initialization will not fall into a certain direction and resulting in too long training time (Figure 12). Considering the GPU memory limitation, we prefer large input images to a large batch size, hence, the batch size will be set to 1.

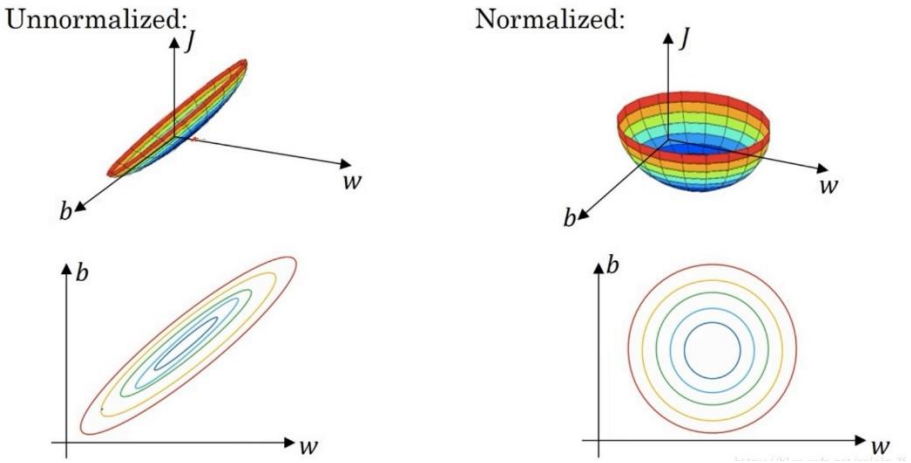


Figure 12. Comparsion between unnormalized data and normalized data, internal covariate shift reduced after normalization[3].

3.2. Fully connected conditional random field

Conditional random fields (CRFs) are a kind of conditional probability distribution model in which one set of input random variables is given and another set of output random variable is given. For prediction problem, the emphasis is on the linear CRF. The problem becomes a discriminant model of input sequence to output sequence prediction in the form of logarithmic linear model. The learning method is usually maximum likelihood estimation or regularized maximum likelihood estimation. The application of linear chain random field to labelling problem was proposed by Lafferty et al., (2001). CRFs are commonly used

[3] https://zhuanlan.zhihu.com/p/33477479

to refine the classification results of other classifiers. Here, fully connected CRF is used in this study to combine with U-net to refine the segmentation results of U-net.

Assuming that X_i is the random variable corresponds to pixel i, which is able to obtain any value from a pre-defined set of labels $L=\{l_1, l_2, ..., l_N\}$. It represents the label assigned to the pixel i. X is the vector constructed by the random variables $\{X_1, X_2, ..., X_N\}$, here N is the number of pixels in the image. Given an undirected graph $G=(V,E)$ which established on X, here, $V=\{X_1, X_2, ..., X_N\}$. Given a global observation over the image I, the conditional random field (I, X) can be modelled by a Gibbs distribution and defined by:

$$P(X \mid I) = \frac{1}{Z}\exp\left(-\sum_{c\in C_G}\phi_c(x_c \mid I)\right) \quad \text{Equation 13}$$

$\phi_c(x_c \mid I)$ is potential function which is defined over variables x_c within a clique c. Clique c is a subgraph of the given undirected graph G, C_G is the set of all cliques in a graph G. Z is a partition function which is a normalization constant. Then the Gibbs energy function can be defined by:

$$E(x) = \sum_{c\in C_G}\phi_c(x_c \mid I) \quad \text{Equation 14}$$

$$Z = \sum_x \exp(-E(x)) \quad \text{Equation 15}$$

x is the labelling that any possible assignment of labels to random variable, $x \in L^N$. For classification problem, the optimal labelling requires a maximum joint probability $P(X|I)$, so the maximum a posteriori (MAP) labelling $x*$ is defined as:

$$x* = \arg\max_{x\in L^N} P(X \mid I) \quad \text{Equation 16}$$

For fully connected pairwise CRF used in this study, the energy function is written as:

$$E(x) = \sum_i \psi_u(x_i) + \alpha\sum_{i<j}\psi_p(x_i, x_j) \quad \text{Equation 17}$$

Where the unary potential $\psi_u(x_i)$ is calculated from probability distribution over labels from classifiers, here this classifier is a deep neural network: U-net, this predicted label does not consider the smoothness and the consistency of the pairwise pixels. Unary potential measure the inverse likelihood of the pixel i taking the label x_i. $\psi_p(x_i, x_j)$ is the pairwise potential which keep the consistency between pixel i and j, encouraging assigning similar labels to pixels which shares the same feature, the pairwise potential can be modelled by a linear combination of weighted Gaussians:

$$\psi_p(x_i, x_j) = \mu(x_{i,} x_j)\sum_{m=1}^{M}\omega^{(m)}\kappa^{(m)}(f_i, f_j) \quad \text{Equation 18}$$

$$\mu(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j, \\ 1 & \text{otherwise,} \end{cases} \quad \text{Equation 19}$$

Where $\kappa^{(m)}$ are Gaussian kernels work on feature vectors. f_i is the feature vector with respect of pixel i which is extracted from image features as spatial location and RGB values (Krähenb et al., 2012). $\mu(x_i, x_j)$ is a Potts model which assigns penalty when two neighbouring pixels have different labels.

3.2.1. Inference

The most probable label assignment x requires to minimizing the above energy function $E(x)$. A mean-field approximation to the CRF distribution is used for approximate maximum posterior marginal inference. It consists in approximating the CRF distribution $P(X)$ by a similar distribution $Q(X)$. $Q(X)$ is a product of independent marginals that can be calculated by minimizing the KL-divergence $D(Q \| P)$ (Koller & Friedman, 2009) which can be written as: $Q(X) = \prod_i Q_i(X_i)$.

The inference is calculated by iteratively passing message, compatibility transform and local update within the approximate field until convergence (Krähenb et al., 2011), update equation is following below:

$$Q_i = (x_i = l) = \frac{1}{Z_i} \exp\left\{ -\psi_u(x_i) - \sum_{l' \in L} \mu(l, l') \sum_{m=1}^{K} \omega^{(m)} \sum_{j \neq i} \kappa^{(m)}(f_i, f_j) Q_j(l') \right\} \quad \text{Equation 20}$$

3.2.2. CRF as RNN

The process of iterative algorithm for approximate mean-field inference was reformulated as an Recurrent Neural Network (RNN) (Zheng et al., 2015). The individual steps of the mean-field algorithm can be described as CNN layer, the detail process is showed in Figure 13. When the steps of inference algorithm are reconstructed as CNN layer, it is important to calculate the error differentials of each layer. Error differentials will be back propagated to the previous layers to optimize the parameters in each layer during training. So, the end-to-end network automatically optimize the weights of the Gaussian kernels and the label compatibility function.

Algorithm 1 Mean-field in dense CRFs [29], broken down to common CNN operations.

$Q_i(l) \leftarrow \frac{1}{Z_i} \exp(U_i(l))$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i} \exp\left(\check{Q}_i(l)\right)$ ▷ Normalizing

end while

Figure 13. Mean-field CRF inference broken down to common CNN operations, source from Zheng et al., (2015).

$U_i(l) = -\psi_u(X_i = l)$ denotes the negative of the unary energy, when CRF is singly used as post-processing operation, $U_i(l)$ is derived from an independent classifier. From Figure 13, the initialization operation $U_i(l) \leftarrow \frac{1}{Z_i} \exp(U_i(l))$ equals to apply a softmax function over the unary potentials for all

the labels at each pixel. After performing the usual back propagation calculations of the softmax transformation, the error difference received at the step output during back propagation can be passed to the unary potential input (Zheng et al., 2015).

It is impossible to directly calculate message passing for every pixel, the complexity of this problem is a quadratic function of the number of pixels in images, the high complexity brought a challenge to inference the CRF. In the image data, inference usually only obtains the approximate solutions, but the accurate solutions can be obtained in the one-dimensional problem, thus, high dimensional Gaussian filtering is proposed to reduce the complexity from quadratic to linear (Krähenb et al., 2011). Message passing is expressed as $\sum_{j\in V} \kappa^{(m)}\left(f_i, f_j\right)Q_i\left(l\right)-Q_i\left(l\right)$ and achieved by applying M Gaussian filters on Q values. The Gaussian filter coefficient is derived based on the position of the image pixel and RGB values reflecting the relationship between one pixel and other pixels (Zheng et al., 2015). Permutohedral lattice (Adams, Baek, & Abraham Davis, 1981) is implemented to speed up the computation of high dimension Gaussian filtering. The bandwidth values of the Gaussian filters used in this study are fixed, besides the Gaussian filter, it is possible to be replaced with other filters such as multiple spatial and bilateral filters with different bandwidth.

In the compatibility transform step, label compatibility function $\mu\left(l,l'\right)$ controls the compatibility between label l and l', learning of the label compatibility function is to learn the weights of this filter. Error differentials can be transferred from the output of this step to the input because this step is a usual convolution operation where the spatial receptive field of the filter is 1 × 1. In adding unary potential step, copying the differentials of the step output to two inputs with appropriate symbols to convey error differentials (Zheng et al., 2015).

After decomposing an iteration of the mean-field algorithm into common CNN operations, multiple mean-field iterations can be realized by repeating the above layer stack mode, Q value is taken in each iteration to estimate the value from the previous iteration and the unary term value in its original form. Therefore, the iterative mean-field inference can be treated as RNN, Figure 14 shows this inference structure:

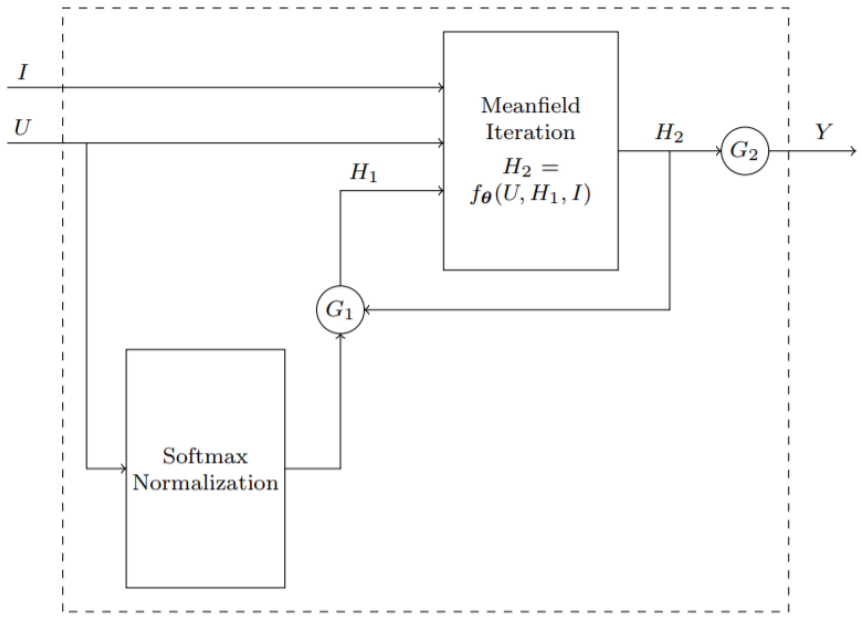


Figure 14. The CRF-RNN Network, the iterative mean-field algorithm is formulated as RNN, gating function G_1 and G_2 are fixed, source from Zheng et al., (2015).

From Figure 14, T represents the number of mean-filed iterations:

$$H_1(t) = \begin{cases} soft\max(U), t = 0 \\ H_2(t-1), 0 < t \le T \end{cases} \quad \text{Equation 21}$$

$$H_2(t) = f_\theta(U, H_1(t), I) \quad 0 \le t < T \quad \text{Equation 22}$$

$$Y(t) = \begin{cases} H_2(t), t = T \\ 0, 0 < t \le T \end{cases} \quad \text{Equation 23}$$

This full network model can be learned by using the standard back propagation through time algorithm. In forward propagation process, when the output of U-net enters the CRF-RNN stage, the data suffers T iteration computation in the loop of RNN. When the loop ends, the output continues the forward propagation and pass through the next step after the CRF-RNN. During the backward propagation process, for propagating the error differentials to the U-net, once the output Y of CRF-RNN gets the error differentials, they need iterations within the loop before reaching the RNN input U . Error differentials are calculated in each component of the mean-field iteration within the loop.

3.3. Quality assessment

Performance of this work is evaluated by annotated testing images and segmentation results are estimated by three measures. Mean intersection over union (IoU) score (Everingham et al., 2010) is usually used as useful instructor in semantic segmentation evaluation, it is to calculate the IoU of each class separately (the intersection ratio of the ground truth and the predicted result), and then calculate the mean of the IoU of all classes. Overall pixel accuracy for entire image is the proportion of pixels marked correctly to the total number of pixels, averaged pixel-wise accuracy for each class is calculating the proportion of pixels correctly classified in each class, they can be formulated by:

mIoU score: $\frac{TP}{TP + FN + FP}$, calculate IoU for each class and then average them.

Overall accuracy: $\frac{TP}{TP + FN}$, calculate the ratio of labelled pixels and the total number of pixels.

Average accuracy: $\frac{TP}{TP + FN}$, calculate accuracy for each class and then average them.

Where TP, FN and FP denote the true positive, false negative and false positive respectively.

4. EXPERIMENT

This chapter introduces how the dataset is prepared and how the model is achieved and how the classifiers are trained to do semantic segmentation. In the experiments, all the programs run on Nvidia Tesla V100 GPU with 16GB RAM on Google Cloud Compute Platform, operation system is Linux Ubuntu 16.04, and the model is implemented by Python in Tensoflow framework.

4.1. Dataset

Airborne images in this study were acquired from an airplane platform with the height of 330m above the urban ground of Enschede, the Netherlands. The dataset consists of 102 nadir view images and 408 oblique view images with 45 degrees tilt angel. The ground sample distance (GSD) of the nadir images is about 10cm, the raw images have resolution of 5616×3744 pixels with RGB channels in JPG format. The contiguous images have 60% overlap because the images are taken consecutively with a short time interval. 136 oblique images were carefully selected to make sure that these images have few common regions.

The study area is a densely-built urban area with very heterogeneous appearance, mainly covered by man-made objects such as buildings, roads, squares, railways and greening. These highly dense buildings have many complex architectural details, roofs are usually made of different materials. Sometimes, the contrast between the roof and ground might be low because of the different reflective properties of the roof materials and make it difficult to detect the detail boundary. Most building areas are located in the centre of the city rather than in the suburbs, and there are all kinds of objects, such as parking lots, vehicles, fence and trees which interacts and disturbs with each other. In fact, almost all the semantic segmentation of aerial images are vertical perspective, buildings and trees are less likely to shelter with each other. The airborne oblique images in this study look more serried and complex, especially in city centre, most buildings are residential houses with small yard and some green plants inside it. These difficulties make the problem of semantic segmentation challenging. What is noteworthy is that the whole city has been paved with bikeways, most of bikeways intersect with vehicle roads. These roads are carefully distinguished because that some of it can be passed through by both bikes and vehicles. Figure 15 gives a detail view of some experiment samples.

(a)

(b)

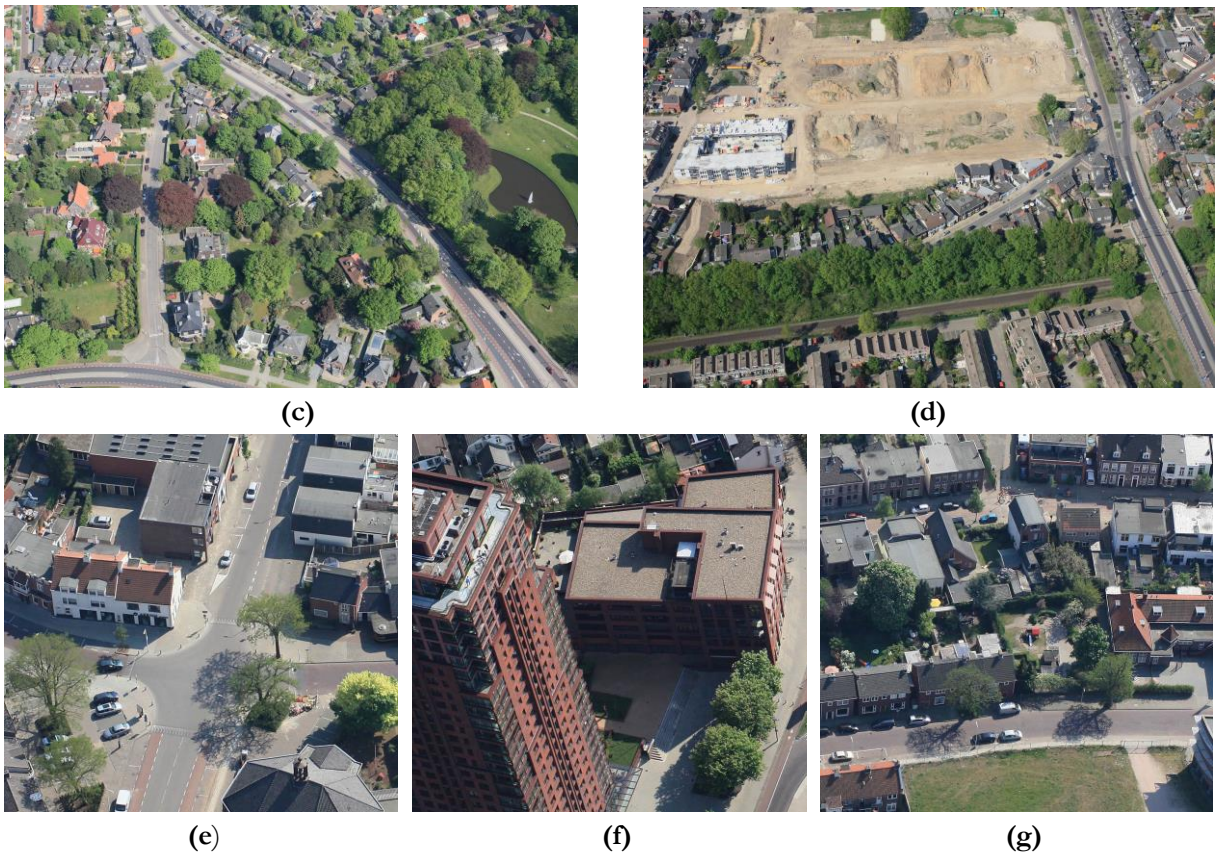


Figure 15. Several samples of dataset used in this study, (a) (b) (c) (d) are respectively four typical urban scenes: urban construction are extremely intensive, different kinds of road intersect with each other, low vegetation and trees highly cover, bare ground are fused with other landcover. (e) is a local view of blurry road boundary, (f) shows the shadow and occlusion caused by buildings, (g) is a detail view of small yards connect with houses.

4.1.1. Annotation

These selected 136 airborne oblique view images were manually annotated with the annotation tool which is developed by Y. Lyu (Ye), it took an average of four hours to label an image. Four classes were fine annotated including: Building (RGB: 214, 13, 56), it is worth noting that baffles, fences and the ground in the small yard were filtered and it is not part of buildings, Vegetation (RGB: 116, 207, 54) contains all the low vegetation and trees, Road (RGB: 255, 223, 13) only refers to vehicle road, excluding bikeway and pedestrian road, Terrain (RGB: 194, 110, 56) refers to the bare ground, Void/background (RGB: 255, 0, 0) contains all classes do not belong to above class. Figure 16 shows an example of annotated image (Ground truth).

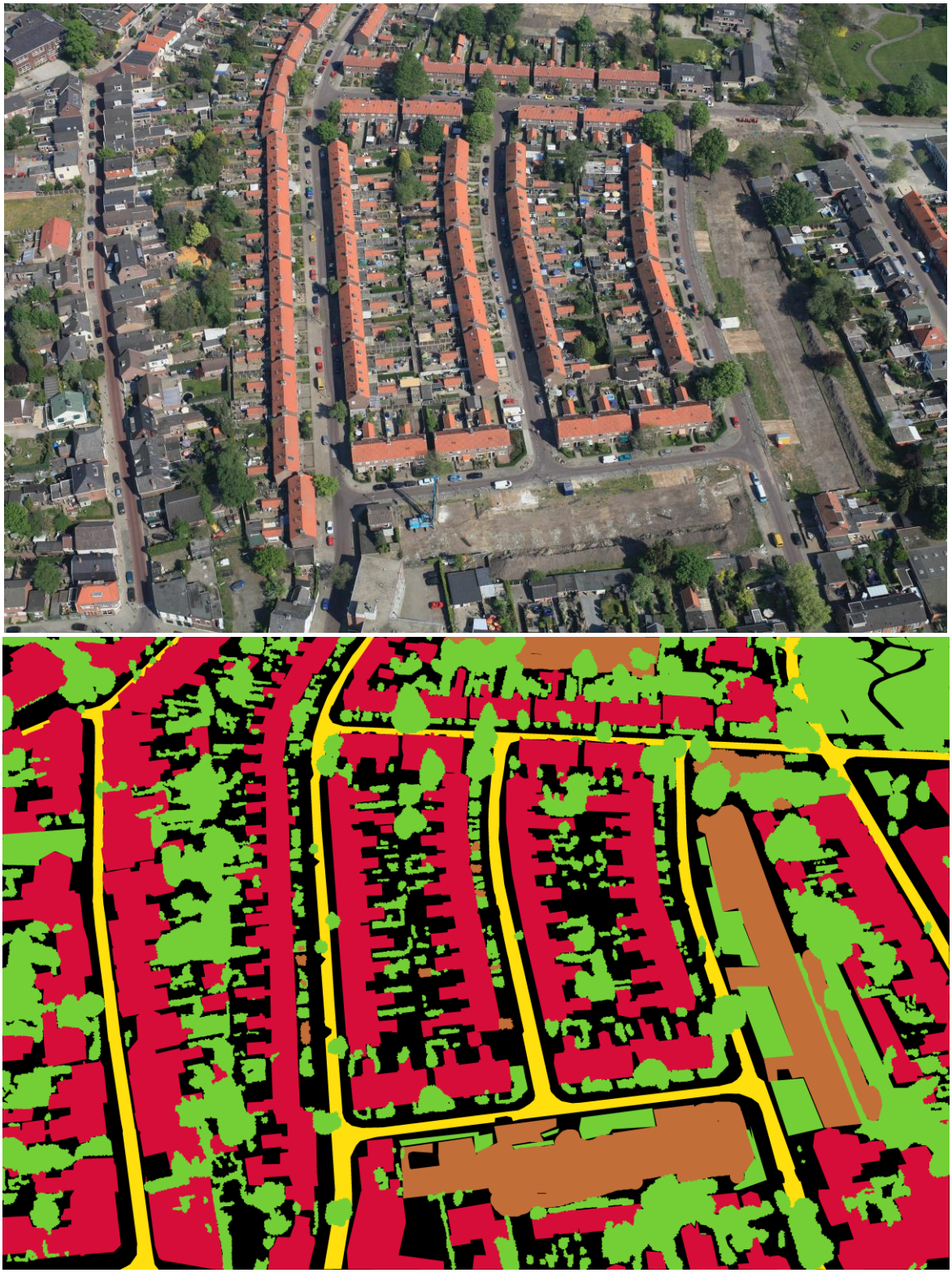


Figure 16. Top: raw image and bottom: ground truth.

4.1.2. Data pre-processing

It is well known that the memory occupied by the neural network model is related about the parameters of the model itself and the output of the model. The raw image has size of 5616×3744 pixels and three feature channels (RGB), if this model uses float32 as data computation value type, that means at least about 63MB (batch size × channels× width× height) memory of GPU will be taken up just for input layer. Considering the GPU memory, the raw images are respectively resized to 768×512 pixels and 1440×960 pixels to experiment. For remote sensing images, it is usually to crop the raw image to smaller patches and feeding them into the network, after that, to generate predictions for larger images, sliding window (with certain overlapping rate of windows) to predict label for each patch and stitched the resulting predictions together. While for these densely-located urban airborne oblique images only with height of 330m, it will be fragmented if cropping it into tiles, especially for buildings and roads. Although cropping approach preserves initial high resolution which contributes to localization, the global contextual information will be seriously lost. Sometimes sematic segmentation must balance between localization and contextual information. Here, two kinds of different size images were used to proceed experiment, 136 images were randomly divided into training, validation and testing dataset with ratio of 6:2:2, therefore there were about 82 images in training dataset and 27 images were respectively in validation and testing dataset. However, pixel distribution over different classes are very unbalanced, Figure 17 shows a statistics of different class pixel proportion over the whole dataset.

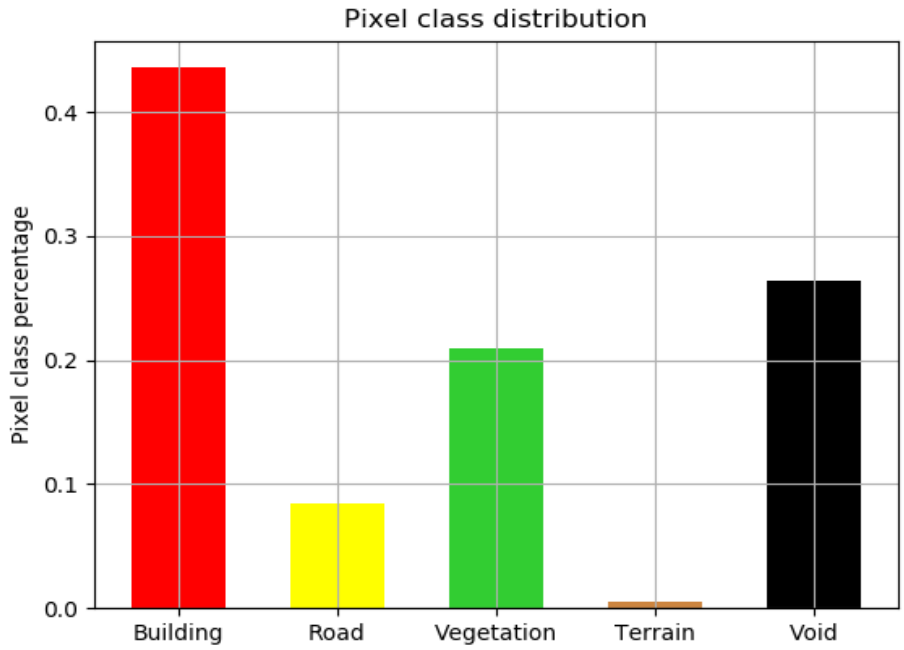


Figure 17. The percentage of each class to the total number of pixels.

4.2. Model parameters

The effect of deep learning largely depends on the parameter adjustment, the parameter adjustment of deep learning often has no standard rules. Parameter is actually a more general name, because it includes not only some number adjustment, it also includes the network structure adjustment and some function adjustment. Parameters can be divided into two classes: training process and training related parameters, network dependent parameter such as the number of layers, the number of nodes and filters. The network dependent

parameter relates to the network structure which is introduced in chapter 3, this chapter mainly explain the training related parameters, also known as hyperparameter.

4.2.1. U-net parameters

First of all, there are two direct purposes for parameter tuning: when there are training errors in the network, parameters are naturally needed to be adjusted; training is available but need to improve the training accuracy of the whole network. A very important parameter is the learning rate of optimizer. Learning rate controls the learning progress of the model, learning rate was denoted to 0.0001 and decay was 0.995. The hyperparameters were picked by evaluating performance using the held-out validation data. Validation accuracy is a good indicator to choose the regularization hyperparameter, the batch size and the network structure parameters, these parameters are designed to improve the final classification accuracy of the test data. Besides this, in the implementation of neural network, some parameters do not directly affect the classification accuracy, but affect the network performance. For example, the learning rate is idiosyncratic, it is impossible to choose the learning rate according to the accuracy result of validation set or test set, its main purpose is to really control the step size in gradient descent. Monitoring the loss function value is better to detect whether the learning step size is too large or too small. The learning rate is quite sensitive, a very large learning rate will lead to abnormal loss curve, while a small learning rate makes loss curve drop slowly and convergence is difficult. On the contrary, the large learning rate will cause the loss curve to drop very fast at the beginning and then converge quickly, which usually reaches the local optimum (Figure 18). The used learning rate and decay show a good performance in this model (Figure 19, Figure 20). A good learning rate loss curve is relatively smooth and has a good performance in the test dataset at last.

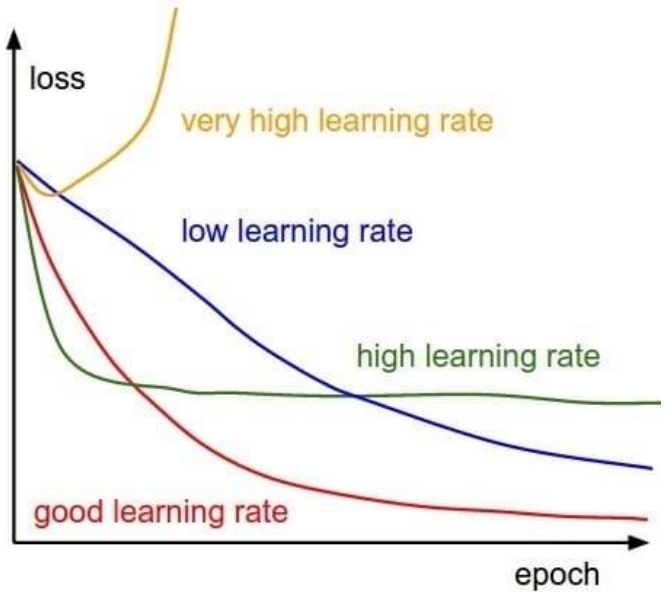


Figure 18. The change curve of different learning rate with respect to loss (Lowndes, 2015).

In actual, the learning rate attenuation can be realized by doing some small constant factor attenuation every few time periods, or by exponential attenuation, which is realized in the mathematical form of exponential of several time periods. In the present case, the rate of decline is also negative: it refers to the rate of decline. But it is necessary to reduce the learning rate, because in the process of training, the higher learning rate is likely to fall into the local minimum. In training, it should be avoided the zigzag bounce of the gradient. Momentum is a parameter of an adaptive learning rate method, it helps to speed up or slow down the learning rate to adapt to changes in the gradient and ultimately leads to changes in the learning rate of the

network rather than changes in its position on the surface of the loss function. Momentum makes the learned network more resistant to noise and randomness in the input numbers. RMSProp Optimizer was applied in this study, it normalizes the current gradient by keeping the moving average of the squared gradient for each weight, increases the resistance to fluctuating and random noises.

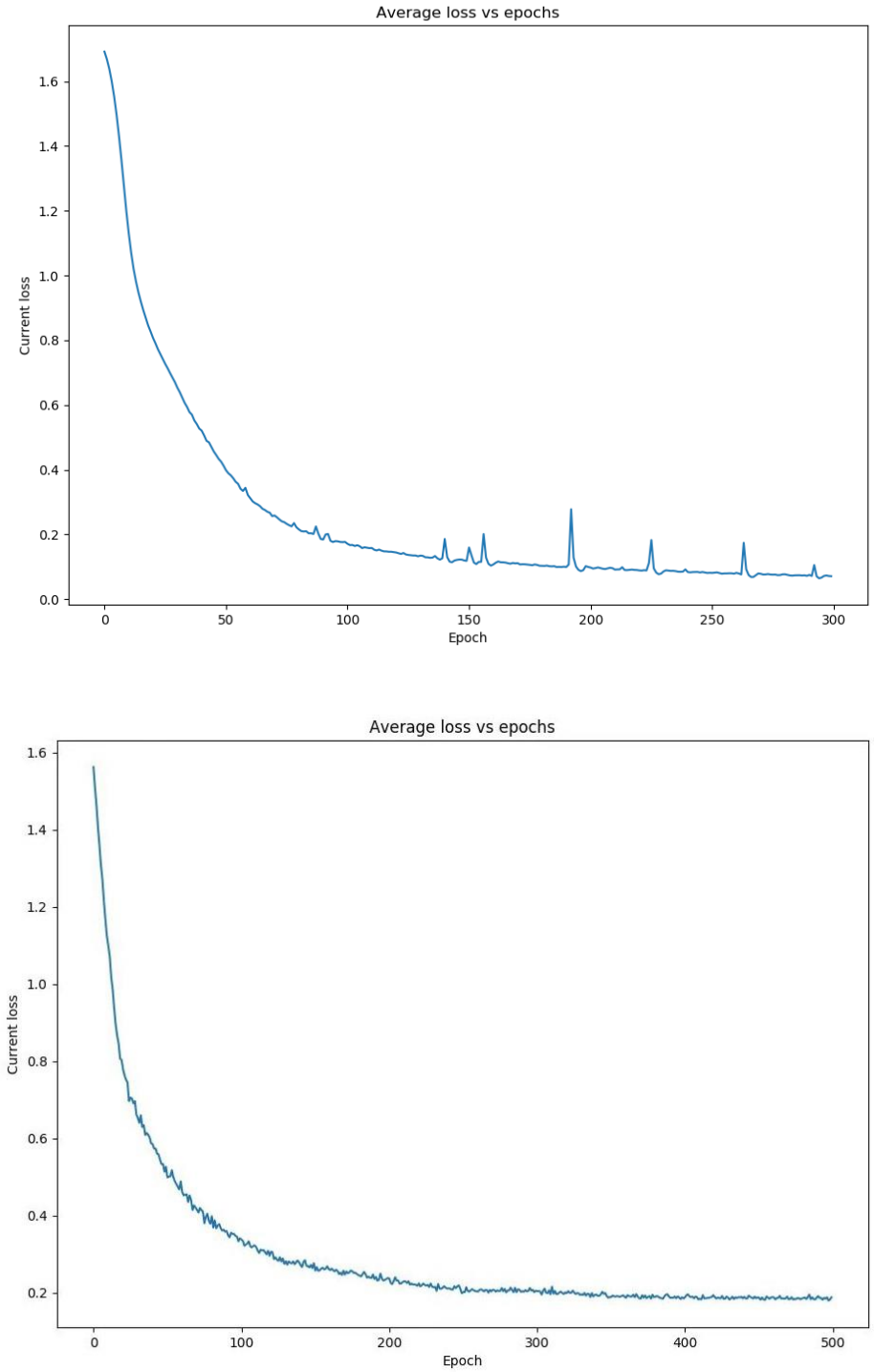


Figure 19. Average loss function curve after network converged, top: image size of 768×512 pixels, bottom: image size of 1440×960 pixels.

When training the neural network model, gradient descent is the most commonly used method, there are three kinds of gradient descent methods: Batch Gradient Descent (BSD), Mini-batch Gradient Descent (MBGD) and Stochastic Gradient Descent (SGD). Here in this model, due to GPU memory, batch size is set to be 1 which is actually a kind of SGD, the idea of SGD is to train only one sample at a time to update parameters. Batch size decides the direction of gradient descent at first, if the dataset is small, it is better to take the form of a full dataset because of the direction of the whole dataset can better represent the sample population and determine its extreme value, also it is difficult to select a global learning rate due to the gradient values of different weights vary greatly.

Nevertheless, larger batch size contributes to the usability of GPU memory and the model requires less iteration for one epoch. Within a certain range, the larger the batch size is, the more accurate the downward direction it determines and the smaller the training shock will be. In conclusion, the result of a large batch size is that the network can easily converge to some bad local optimum, the batch size is too small also has some problems, such as slow training speed and difficulty in convergence, the selection of specific batch size is related to the number of samples in the training set. From the loss function curve (Figure 19) and the validation accuracy curve (Figure 20 and Figure 21), it is not difficult to found that the learning rate and batch size are suitable and efficient in this model.

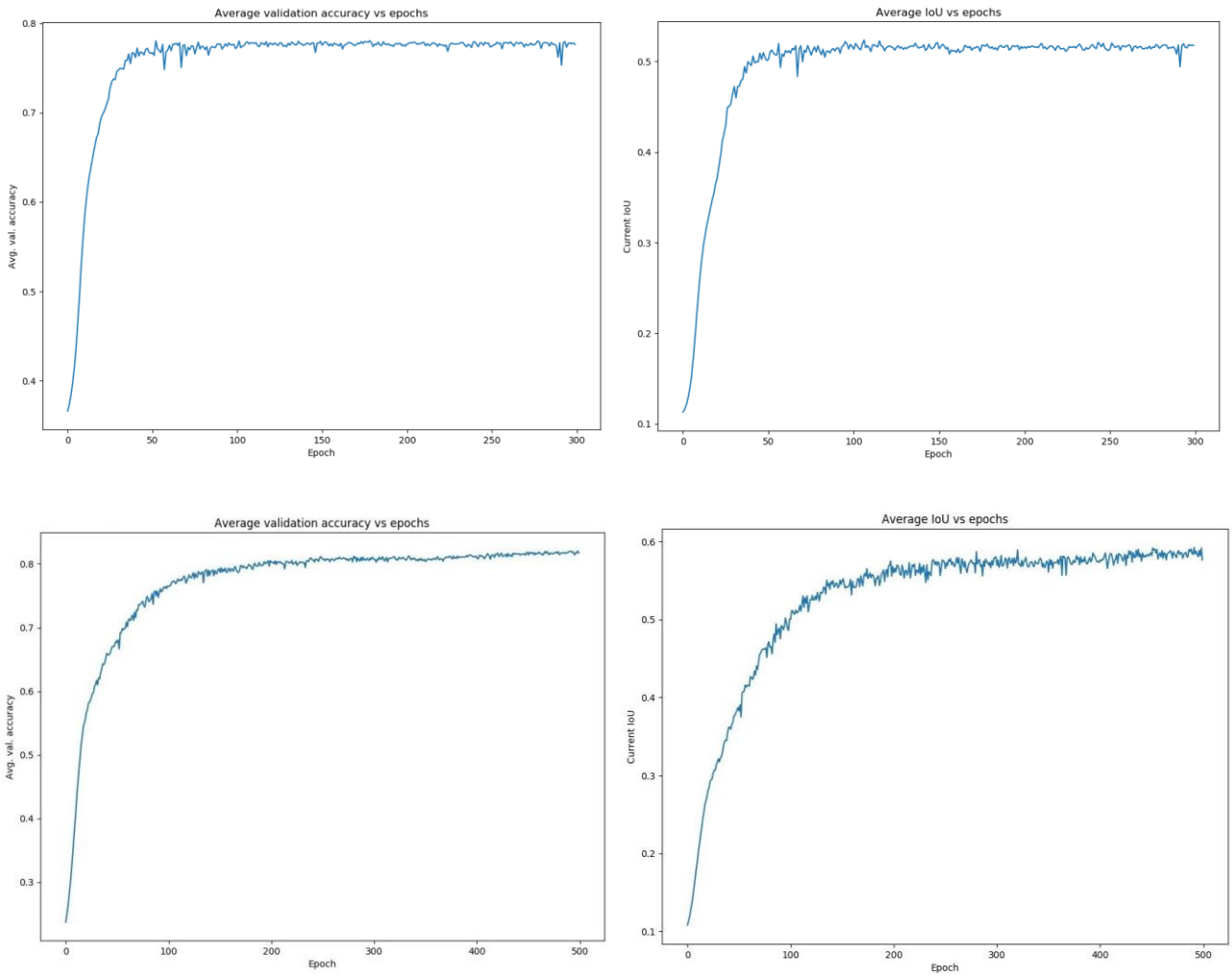


Figure 20. Average validation accuracy and mean IoU curve, the first row: image size of 768×512 pixels; the second row: image size of 1440×960 pixels.

4.2.2. CRF as RNN implementation

A strong pixel-wise U-net classifier was well trained in previous work, this CRF-RNN approach was plugged in as a part of a conventional neural network. While training the U-net and learning CRF parameters together, it is feasible to pass on error differentials from its outputs to input during back propagation.

As the first part, U-net was well trained in previous work and output unary with good quality, so the U-net is initialized by the well-trained weight to get unary. During training this complete system, the batch size still keeps to be 1, a single image is used to compute the error at each pixel out of the network by using softmax loss for the ground truth of the image. In the inference process, Stochastic Gradient Descent (SGD) is used for back propagation, the gradient is directly sent to unary after back calculation by softmax. The compatibility transform parameters of CRF-RNN are initialized by a Potts model, the filter weight and kernel width parameters are initialized by the empirical value. The learning rate for compatibility transform matrix was set to be 1×10^{-13} and momentum is 0.99. The number of mean-filed iteration T was set to 5 in training.

5. RESULTS

5.1. U-net

Table 1. U-net segmentation results from image size of 768×512 pixels.

	Validation (%)	Test (%)
Building	90.2	91.2
Road	40.4	41.3
Vegetation	92.3	94.1
Terrain	54.1	48.2
Average class accuracy	69.2	68.7
Overall pixel accuracy	78	78.5
Mean IoU	53.2	52.8

Table 5.1 show results from U-net classifier for image size of 768×512 pixels. From the table, 91.2% of building pixels were correctly labelled, 94.1% of vegetation pixels were correctly labelled, while the accuracies of road and terrain were 41.3% and 48.2%. So, building class gained the highest accuracy among these classes. Here, average class accuracy was calculated for every class and then averaged, the average class accuracy is about 68. 7%, because two classes have higher accuracy while the other two classes have relative low accuracy. The overall accuracy represents the ratio of all correctly classified pixels to the total number of image pixels, 78.5% of overall accuracy is not low. Here, mean IoU receives more attention to be an authoritative indicator to assess the network, 52.8% of mean IoU is a reasonable result since the resized image lost a lot of details.

Observing Figure 22 (the third row), the classification results for almost all classes are too fragmented. The segmentations of the large objects are relatively intact, for instance, there are some holes in the segmentation results of large buildings. This usually happens in buildings with some sort of special reflective material on the roof, or the holes might be caused by the shadows cast on the roof by the tilt angle of image. In the Figure 22 (the third row) and Figure 23 (the third column), the residential building block which highly intersect with vegetation has the biggest errors. In addition, the bikeway was misclassified to road, road only refers to the road used for vehicles in this experiment.

Table 2. U-net segmentation results from image size of 1440×960 pixels.

	Validation (%)	Test (%)
Building	90	93.3
Road	48.1	50.7
Vegetation	86.7	91.7
Terrain	58.2	46.8
Average class accuracy	70.8	70.7
Overall pixel accuracy	82	84.1
Mean IoU	58.8	59.9

When increased the input image size to 1440×960 pixels (Table 2), the accuracy improvement of building was 2.1%, in Figure 22 (the fourth row), small holes inside buildings were partly corrected, most part of buildings were no longer classified to a whole building block, small buildings were well distinguished. The biggest improvement in accuracy was road, it has been improved about 9.4%. The misclassified road was completely removed, road continuity has been greatly improved and the boundary of road is much smoother and regular. On the contrary, the accuracy of vegetation slightly declined 2.4%, there are still some errors and fragments inside the vegetation (Figure 23 the fourth column). The accuracy of terrain also decreased 1.4%, it is still hard to recognize bare terrain after improving the resolution. Mean IoU significantly improved 7.1%, the object boundary for each class was much more accurate and smoother. Figure 21 gives a visual view of accuracy changes by image size.

The increase of spatial resolution of the image result in the area covered by a single pixel is smaller and the reduced the mixed pixels, preserves more details. It is easier to distinguish the object boundary and percept localization information, so mean IoU has significantly improved 7.1%. Terrain always got the lowest accuracy within these classes mainly because of very little training pixels for terrain. Besides this, the bare terrain is often mixed with vegetation and ground so that produce a lot of mixed pixels.

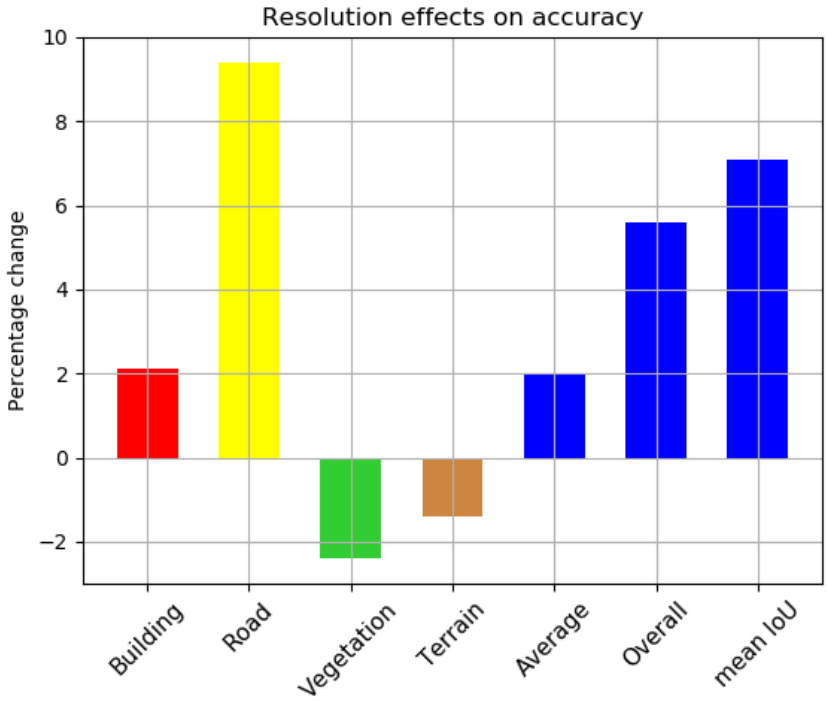


Figure 21. Accuracy changes effected by image size

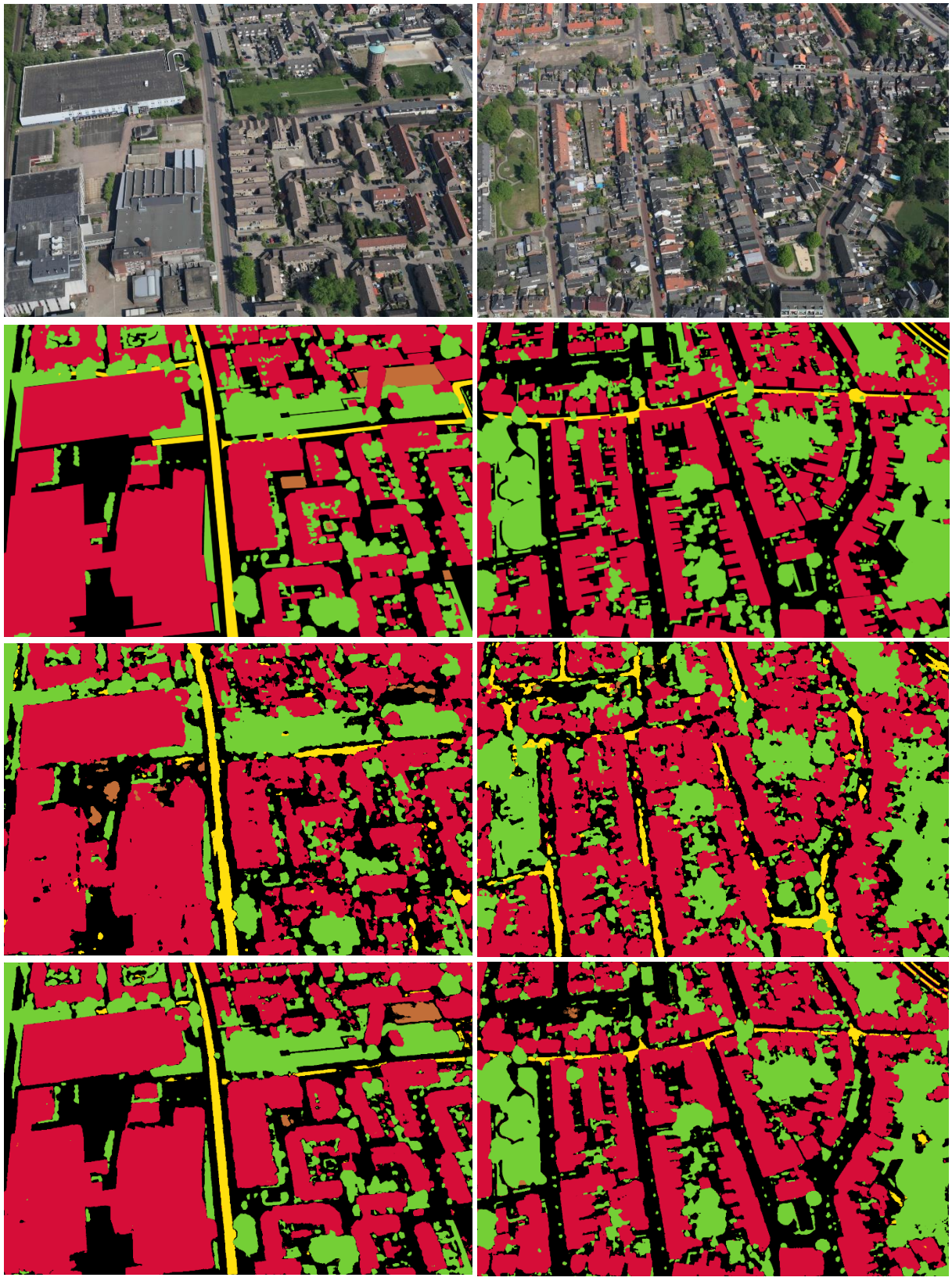

Building
Road
Vegetation
Terrain
Void

Figure 22. Examples from U-net segmentation of (First row: raw image; Second row: ground truth; Third row: results from input image size of 768×512 pixels; Fourth row: results from input image size of 1440×960 pixels.

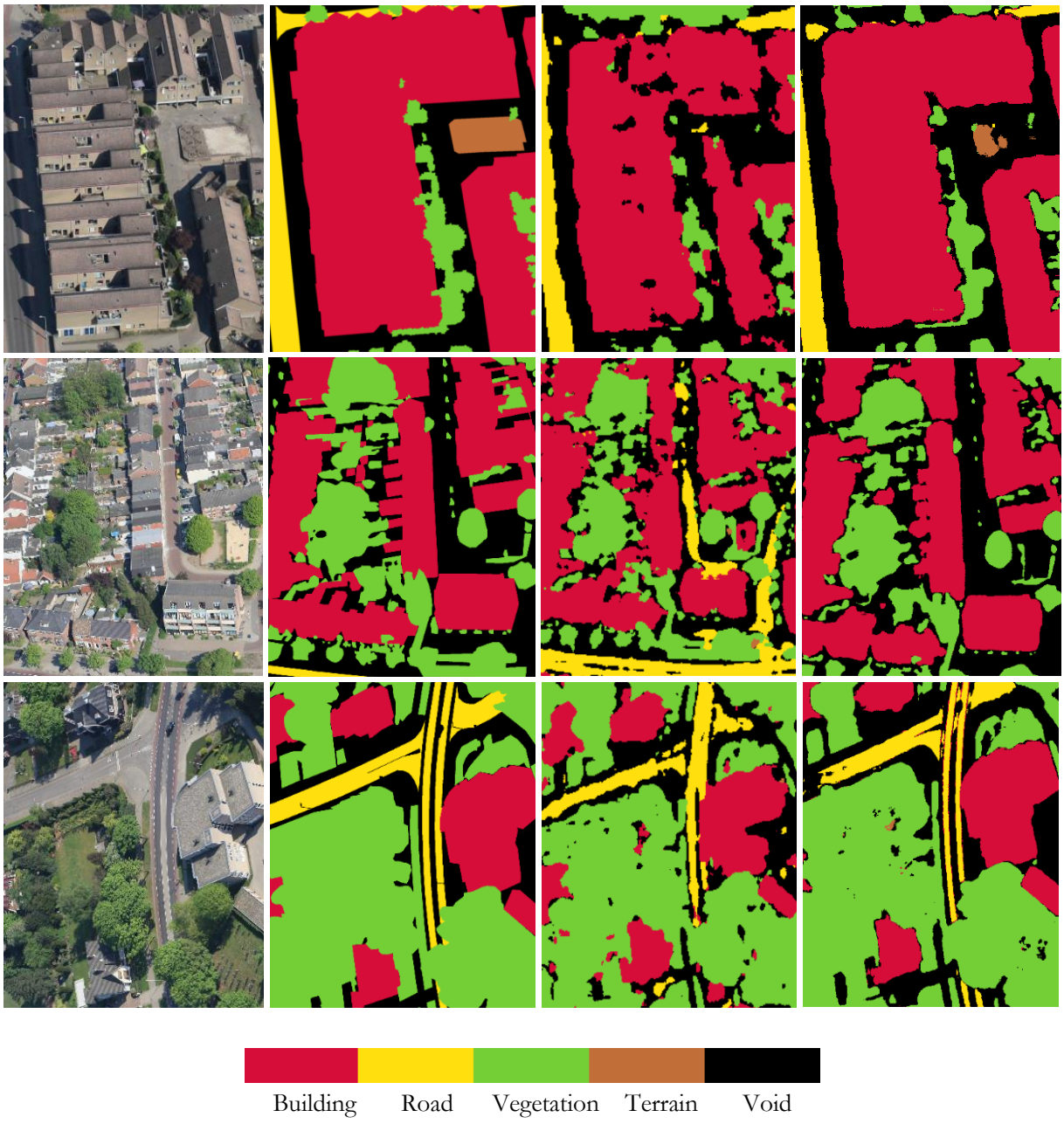


Figure 23. Local details of U-net segmentation results: (First column): Raw image; (Second column): Ground truth; (Third column): Semantic segmentation results from image size of 768×512 pixels; (Fourth column): Semantic segmentation results from image size of 1440×960 pixels.

5.2. Fully connected CRF as RNN

Table 3. CRF as RNN based on U-net segmentation accuracy.

	U-net (small image)	U-net (large image)	CRF-RNN
Building	91.2	93.3	95.8
Road	41.3	50.7	55.3
Vegetation	94.1	91.7	90.8
Terrain	48.2	46.8	47.0
Average class accuracy	68.7	70.7	72.7
Overall pixel accuracy	78.5	84.1	86.5
Mean IoU	52.8	59.9	62.3

Because the U-net obtained better segmentation results on 1440×960 pixels images, this result was used as unary term to apply to CRF-RNN. Fully connected CRF was implemented with U-net to formulate an end-to-end model, from Table 3, all the segmentation results have different degree of improvement in accuracy. The accuracy of building and vegetation respectively reached 95.8% and 90.8%. 55.3% of road pixels were correctly labelled, the problem of road discontinuity has been greatly improved, some misclassification of road was also removed. Terrain has slightly improved 0.2% on accuracy. Figure 24 shows the CRF-RNN segmentation result sample compared to U-net result and ground truth. It is not difficult to find that the noise and fragments inside the vegetation were smoothed out, the output the U-net has a lot of noisy pixels around the road boundary, in this CRF-RNN optimization, these noisy pixels were greatly reduced. The boundaries of road and building are clearer and more regular. There are also some small objects were cleared which should be correct labels. Mean IoU has improved 2.4% through CRF-RNN, it has better performance on road and building mainly because a lot of noise pixels were removed.

Raw image

Ground truth

U-net

CRF as RNN

Figure 24. Local details of semantic segmentation for urban airborne oblique images.

6. DISCUSSION AND CONCLUSION

6.1. U-net

The modified U-net architecture and the implementation of CRF-RNN meet the needs proposed in research objectives in this study. The U-net architecture based on VGG16 demonstrated the capability of U-net structures to adapt to small dataset and large-scale urban airborne images. This network preserves features from each down-sampling layer and fused it in up-sampling layers, these features are concatenated not added which increased the thickness of features to enhance the network ability of pixel-level classification from a highly mixed class environment. Additionally, different from conventional CNN architecture, successive layers where pooling operations are replaced by up-sampling operations avoid losing more localization information.

Two different size of images were used to experiment in this study because the raw image is too large. Usually in order to retain the original image details, the large images are cropped to equal patches and feed them into the network to train, predict along these patches and stitch them together. But the stitching process may lead to edges of each patches, and other algorithms are needed to eliminate splicing traces. Here we have made experiment in such way in order to preserve details as much as possible, but this directly led to large number of prediction error. The image data, used in this study has only 330m height above ground, study area is located in city centre and urban landscape is highly dense, the method of cropping it into patches badly broke the semantic information. Small patch can only percept very local information and consider very few context information. Considering the GPU using, 1440×960 pixels size of images which were resampled by bilinear interpolation were used in this study to carry out the next step. We still used smaller images in experiment to observe the effect of spatial resolution, the experiment confirmed that the original image details greatly influence the accuracy of the network. Here is a limitation of input data, a method is needed to preserve the details of the input as much as possible while consider the GPU memory limitation.

An efficient computation reduction approach: Mobilenet in this network achieved fast and less-parameter computation, for example, testing a 1440×960 pixels size image only requires 0.25s (Table 4). The depthwise separable convolution effectively improved performance, this network is suitable for dense and fast segmentation.

Table 4. Performance of U-net used in this study.

Image size (pixels)	Training pixel number	GPU using (G)	Training time (h)	Test time per image (s)
768×512	3.20×10^7	7. 8	8.5	0.014
1440×960	1.13×10^8	15	15	0.25

6.2. Fully connected CRF

Fully connected CRF as RNN outperformed single U-net classifier in this study, buildings and roads have obvious improvement on accuracy. CRF-RNN is flexible to applied with U-net and works well in practice.

Although the CRF-RNN achieved an end-to-end manner to automatically learn parameters, training the whole model from beginning may be very time-consuming, because CRF-RNN only support 1 batch size computation. The common approach is to train the DCNN but not to reach the convergence state. From our previous work. In Figure 19 and Figure 20, the loss function dropped rapidly less than 100 epoch training, and then slowly decrease. at this time, the accuracy has reached a relatively optimistic value. Therefore, the parameters at this point can be directly used to initialize the network when training CRF-RNN. Such a non-convergence pre-training makes it feasible to train the CRF-RNN from the beginning.

6.3. Answers to research questions

Objective 1:

How to make DCNN adapt to the airborne oblique urban images in this study and what are optimal parameters in this classifier?

Here we choose U-net to be this DCNN classifier, and in the following questions, this DCNN classifier should be U-net. As we mentioned before, as the network deepens, more advanced features can be extracted. Based on original U-net architecture with VGG11 encoder which is proposed by Iglovikov & Shvets, (2018), U-net with deeper layer network structure was used in this study. Here, VGG16 was selected to implemented as decoder. Figure 7 compared the differences between VGG families, it can be concluded that VGG16 can deepen the network without adding more parameters, and it outperformed VGGG11 in many tasks. While VGG19 continues to deepen the network, but the performance is not significantly improved. Therefore, we replaced VGG11 convolution block with VGG16 convolution block in encoding process considering the receptive field of objects, the depth of network proved appropriate.

Considering that VGG16 has more than one million parameters need to train, and a series of data augmentation techniques make the original training data several times larger, this has a certain impact on the applicability of the network, because the training time must be kept at a reasonable level. For this experiment, depthwise separable convolution is used to decompose a standard convolution operation in order to reduce the number of parameters and speed up training.

For parameters in this network, learning rate was denoted to 0.0001 and decay was 0.995, batch size was set to 1 considering the computation of GPU. RMSProp Optimizer was applied in this study to normalize the current gradient by keeping the moving average of the squared gradient for each weight.

What is the accuracy matrix of this classifier?

Two different size of images were used to experiment on this modified U-net: image size of 768×512 pixels and 1440×960 pixels, the two different size images were resampled by bilinear interpolation. Because U-net recoveries the details which is lost by down-sampling and pooling operation as much as possible in encoding process, so we focus on how the quality of input image affected the accuracy of the network.

For image size of 768×512 pixels, the overall accuracy is 78.5% and mean IoU is 52.8% (Table 1). For image size of 1440×960 pixels, overall accuracy is 84.1% and mean IoU is 59.9% (Table 2). Buildings always got the highest accuracy, but terrain gained the lowest accuracy.

How can the image resolution influence the classification results?

From comparing the segmentation results of two different size images, road was most affected by image resolution. The accuracy of road and building were respectively improved 9.4% and 2.1%, however the accuracy of vegetation and terrain slightly declined.

The higher resolution makes the coverage area inside a single pixel smaller, resulting in the mixed pixel greatly reduced, it facilitates localization and reduces the similarity between classes. In general, mean IoU has improved 7.1%, the improvement of mean IoU is larger than that of accuracy, the boundary of segmentation object has been greatly improved, the resolution of input data has great influence on localization although we try to preserve the details as much as possible within the architecture of network.

Objective 2:

How to apply the CRF-RNN model to the segmentation problem?

From segmentation results in Figure 22 and Figure 23, the output of U-net contains a lot of noisy pixels, especially along the road and inside the vegetation, part of building boundary is coarse, the regional block learning of neural network limits the understanding of context. A fully connected CRF which considers the long-range interactions between pixels was used to refine this output. Usually treat the CRF as a post-processing operation to optimize the output of the other classifier. This kind of operation makes the parameter adjustment of the model have certain randomness, the Gaussian kernel weights, spatial colour parameters are combined to experiment and the obtain the optimized parameter value combination according to inference.

In this study, fully connected CRF was joint with U-net to train an end-to-end model. A mean-field inference of CRF was formulated as a stack of common CNN layers, repeating the stack of these layers so that multiple average field iterations can be implemented, each with an estimation of the previous iteration's marginal probabilities and a unary value of the original form. In actual, the repeating mean-field inference is regarded as a king of RNN process. The key of CRF-RNN is error differentials in CRF can be back propagated to the unary classifier part, the error differential is also the difference between the predicted value and the ground truth in classification problem. While the parameters are automatically learned in this complete system, the learning of filter weights is to learn the compatibility of two labels. Often this complete system contains a DCNN classifier to obtain unary, so this end-to-end model needs to train from beginning, here we had well trained U-net, so the U-net was directly initialized by our well-trained parameters, only learn the parameters in CRF.

How to construct this fully connected CRF? What are the parameters required for specifying the fully connected CRF model?

We plugged our CRF-RNN inference layer with the U-net architecture based on its original network structure. The Gaussian filters is combined with bilateral filter and spatial filter. Bilateral filter considers not only the distance between pixels but also the pixel value differences, the smaller difference gets the bigger weight, on the contrary, big difference receives the smaller weight. The spatial filter removes some isolate pixels of output. But the bandwidth of the kernels still needs to be manually defined. Here, the bandwidth value used in this study was got from empirical value. In addition, the iteration number of mean-field inference should be specified, we used 5 iterations for training in this experiment, this value is an empirically value comes from Zheng et al., (2015), they got a conclusion that too many iterations cause vanishing gradient problem so that the unary receive a small error gradient.

Objective 3:

What is the difference in segmentation accuracy between DCNN after CRF-RNN fused and single U-net?

Comparing the segmentation results with single U-net classifier, the overall accuracy has improved 4.2% and IoU improved 3.2% (Table 2, Table 3). After U-net was fused with a fully connected CRF to formulate an end-to-end model, it is obviously that some small misclassification fragments were basically removed, and boundary of objects is sharp than that in single U-net classifier. Building improved 2.5% on accuracy, road got 4.6 % improvement while vegetation declined about 0.9%. In general, end-to-end training of U-net and CRF-RNN outperformed single deep neural network classifier U-net.

Which class can gain the most benefits from fully connected CRF and which class gain the least benefits?

From Table 2 and Table 3, road gained the most benefits of 4.6% from fully connected CRF but vegetation gained the least, the accuracy declined 0.9%. There are lots of isolate pixels in vegetation such as trees and grass, such small vegetation pixels are easily filtered out by spatial filter.

6.4. Future work

- In this study, bilateral filter and spatial filter are shared for all classes in CRF but the accuracy of some classes declined. It may be due to the different sensitivity of different class to the filter, considering using different filters for different classes can be investigates in the future.

- The future direction tends to be weakly supervised learning, the application of fully connected CRF is the compatibility learning of pairwise, considering introduce a weak prior to the pairwise potential to guide learning is a new way.

- 3D information has been considered as a feature to help image segmentation. We made a small experiment that established a match between the image and the point cloud data, the point cloud was projected into the corresponding pixel of the image to obtain a 3D feature map, and then fed it into U-net as the fourth feature cannel to train. The point cloud we used in this study is "Actueel Hoogtebestand Nederland (AHN2)" which is obtained by laser scanner. But the point cloud projection is not good because the point cloud density is not enough compared to the image resolution. In addition, AHN2 data that covers Enschede only has ground and non-ground classes which is not feature representative in our study. If there is point cloud data that can provide efficient semantic information, then the 3D could be a strong feature representation in DCNN.

LIST OF REFERENCES

Adams, A., Baek, J., & Abraham Davis, M. (1981). Fast High-Dimensional Filtering Using the Permutohedral Lattice. *Eurographics 2010* (Vol. 0). Retrieved from https://graphics.stanford.edu/papers/permutohedral/permutohedral.pdf

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D City Models: State of the Art Review. *ISPRS International Journal of Geo-Information*, *4*(4), 2842–2889. https://doi.org/10.3390/ijgi4042842

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2014). Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. arXiv:1412.7062

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). 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. doi: 10.1109/TPAMI.2017.2699184

Chen, R. (2011). The development of 3D city model and its applications in urban planning. In *2011 19th International Conference on Geoinformatics* (pp. 1–5). DOI: 10.1109/GeoInformatics.2011.5981007

Claesson, L., & Hansson, B. (2017). Deep Learning Methods and Applications Classification of Traffic Signs and Detection of Alzheimer's Disease from Images. Retrieved from http://publications.lib.chalmers.se/records/fulltext/248445/248445.pdf

Döllner, J., Kolbe, T. H., Liecke, F., Sgouros, T., & Teichmann, K. (2006). THE VIRTUAL 3D CITY MODEL OF BERLIN-MANAGING, INTEGRATING AND COMMUNICATING COMPLEX URBAN INFORMATION. Retrieved from http://misc.gis.tu-berlin.de/igg/htdocs-kw/typo3_src/fileadmin/citygml/docs/udms_berlin3d_2006.pdf

Everingham, M., Van Gool, L., I Williams, C. K., Winn, J., Zisserman, A., Everingham, M., … Zisserman, A. (2010). The PASCAL Visual Object Classes (VOC) Challenge. *Int J Comput Vis*, *88*, 303–338. DOI 10.1007/s11263-009-0275-4

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., & Garcia-Rodriguez, J. (2018). A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, *70*, 41–65. Retrieved from https://doi.org/10.1016/J.ASOC.2018.05.018

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., … Wang, G. (2018). Recent Advances in Convolutional Neural Networks. *Pattern Recognition*, Volume 77, 354-377. Retrieved from https://arxiv.org/pdf/1512.07108v3.pdf

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778.Retrieved from http://arxiv.org/abs/1512.03385

He, X., Zemel, R. S., & Carreira-Perpiñán, M. A. (2004). Multiscale Conditional Random Fields for Image Labeling. Proceedings of the *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004. CVPR 2004. DOI: 10.1109/CVPR.2004.1315232

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. Retrieved from https://arxiv.org/pdf/1207.0580.pdf

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., & Andreetto, M. (2017).

MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. Retrieved from https://arxiv.org/pdf/1704.04861.pdf

Iglovikov, V., & Shvets, A. (2018). TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation. arXiv:1801.05746.

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Retrieved from http://arxiv.org/abs/1502.03167

Kohli · L'ubor Ladický, P., Torr, P. H. S., Kohli, P., Ladický, L., & Torr, P. H. S. (2009). Robust Higher Order Potentials for Enforcing Label Consistency. *Int J Comput Vis*, *82*, 302–324. DOI 10.1007/s11263-008-0202-0

Koller, D., & Friedman, N. (2009). Probabilistic Graphical Models - Principles and Techniques. *Undefined*. Retrieved from https://www.semanticscholar.org/paper/Probabilistic-Graphical-Models-Principles-and-Koller-Friedman/04f39720b9b20f8ab990228ae3fe4f473e750fe3

Krähenb, P., Krähenbühl, K., & Koltun, V. (2012). Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *Advances in Neural Information Processing Systems* 24 (2011) 109-117. arXiv:1210.5644

Krähenbühl, P., & Koltun, V. (2012a). Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. Retrieved from http://arxiv.org/abs/1210.5644

Krähenbühl, P., & Koltun, V. (2012b). Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. Retrieved from http://arxiv.org/abs/1210.5644

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in neural information processing systems* 25(2). DOI: 10.1145/3065386

Ladicky, L., Russell, C., Kohli, P., & Torr, P. H. S. (2009). Associative hierarchical CRFs for object class image segmentation. In *2009 IEEE 12th International Conference on Computer Vision* (pp. 739–746). IEEE. Retrived from https://doi.org/10.1109/ICCV.2009.5459248

Lafferty, J., Mccallum, A., Pereira, F. C. N., & Pereira, F. (2001a). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. Proceedings of the *18th International Conference on Machine Learning 2001* (ICML 2001), pages 282-289. Retrieved from http://portal.acm.org/citation.cfm?id=655813

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. DOI: 10.1109/5.726791

Leung, T., & Malik, J. (2001). Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons. *International Journal of Computer Vision*, *43*(1), 29–44. Retrived from https://doi.org/10.1023/A:1011126920638

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3431–3440). Retrived from https://doi.org/10.1109/CVPR.2015.7298965

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision* (pp. 1150–1157 vol.2). DOI: 10.1109/ICCV.1999.790410

Lowndes, A. (2015). Deep Learning with GPU Technology for Image & Feature Recognition. DOI: 10.13140/RG.2.1.3361.6800

Marmanis, D., Galliani, S., Schindler, K., Zurich, E., Marmanis, D., Wegner, J. D., … Stilla, U. (2016). SEMANTIC SEGMENTATION OF AERIAL IMAGES WITH AN ENSEMBLE OF CNNS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume III-3. DOI: 10.5194/isprs-annals-III-3-473-2016.

Moser, G., Serpico, S. B., & Benediktsson, J. A. (2013). Land-Cover Mapping by Markov Modeling of Spatial–Contextual Information in Very-High-Resolution Remote Sensing Images. *Proceedings of the IEEE*, *101*(3), 631–651. DOI: 10.1109/JPROC.2012.2211551

Nguyen, T. T., Dinh, S. V., Quang, N. T., & Binh, H. T. T. (2017). Semantic segmentation of objects from airborne imagery. In *2017 Fourth Asian Conference on Defence Technology - Japan (ACDT)* (pp. 1–6). DOI: 10.1109/ACDTJ.2017.8259608

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp 234-241. DOI:10.1007/978-3-319-24574-4_28

Schmid, C. (n.d.). Constructing models for content-based image retrieval. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* (Vol. 2, p. II-39-II-45). DOI: 10.1109/CVPR.2001.990922

Sherrah, J. (2016). Fully Convolutional Networks for Dense Semantic Labelling of High-Resolution Aerial Imagery. arXiv:1606.02585

Shotton, J., Winn, J., Rother, C., & Criminisi, A. (2009a). TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. *International Journal of Computer Vision*, *81*(1), 2–23. Retrived from https://doi.org/10.1007/s11263-007-0109-1

Shotton, J., Winn, J., Rother, C., & Criminisi, A. (2009b). TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. *International Journal of Computer Vision*, *81*(1), 2–23. Retrived from https://doi.org/10.1007/s11263-007-0109-1

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. Retrieved from http://arxiv.org/abs/1409.1556

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going Deeper with Convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1-9. DOI:10.1109/CVPR.2015.7298594

Tang, C. I., & Yiu, S. M. (2018). Classification for Pathological Images Using Machine Learning Intermediate Report. Retrieved from https://pdfs.semanticscholar.org/3815/83ec24d7250f354d7a97d46243cbb760265d.pdf?_ga=2.28234364.1938803323.1552906088-2079848305.1547106806

Vezhnevets, A., Ferrari, V., & Buhmann, J. M. (2012). Weakly supervised structured output learning for semantic segmentation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 845–852). DOI: 10.1109/CVPR.2012.6247757

Volpi, M., & Tuia, D. (2017). Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(2), 881–893. DOI: 10.1109/TGRS.2016.2616585

Yu, H., Yang, Z., Tan, L., Wang, Y., Sun, W., Sun, M., & Tang, Y. (2018). Methods and datasets on semantic segmentation: A review. *Neurocomputing*, *304*, 82–103. Retrived from https://doi.org/10.1016/J.NEUCOM.2018.03.037

Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). *Pyramid Scene Parsing Network*. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2881-2890. arXiv:1612.01105

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., … Torr, P. H. S. (2015). Conditional Random Fields as Recurrent Neural Networks. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1529–1537. DOI:10.1109/ICCV.2015.179