# Semantic Segmentation for Aerial Imagery with Convolutional Neural Networks

March 2016

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering

Keio University

CALAMVS GLADIO FORTIOR

1858

**Keio University**

Graduate School of Science and Technology
School of Integrated Design Engineering

Shunta Saito

# Abstract

Aerial imagery has been utilized as an important information resource for many applications such as map making, urban planning, land use analysis, environmental monitoring, real estate management, and disaster relief, etc. In those applications, extracting ground objects such as buildings and roads from aerial imagery is necessary as a pre-processing. This task is to give an object class label for each pixel in an aerial image, so that it is a highly attentional task for human and a very costly and time-consuming process. Therefore, there have been many attempts at automating this task, but fully automated systems which currently exist still have room for improvement in terms of accuracy and simultaneous extraction of multiple objects.

In this thesis, a new method to train a convolutional neural network (CNN) efficiently for simultaneous extraction of multiple objects is proposed. To formulate the object extraction task as a semantic segmentation problem, a *background* class is often introduced as one of the classes of interest. In such situation, the *background* class has a different characteristic than the other classes such as buildings and roads because the *background* class represents many kinds of objects as a single class. Therefore, a new activation function for the final layer of a CNN to focus on the specialty of the *background* class is proposed. Then, it is shown that the CNN which uses the activation function achieves the state-of-the-art accuracy on a publicly available dataset for aerial imagery labeling. The effectiveness of the proposed method is discussed by empirical evaluation.

Finally, the usefulness of the automatic semantic segmentation for aerial imagery is shown by performing a seamline determination task using the results of semantic segmentation and achieving the optimal result with a simple optimization process.

# Acknowledgments

First, I want to thank Yoshimitsu Aoki for being a tolerant advisor. He gave me many great opportunities to learn various things and a perfect environment to focus on the problems on my research themes despite I could not always respond to his expectations. I also want to thank Hideo Saito for giving me a great opportunity to study abroad. I benefited not only from his deep insights and knowledge but also from his encouragement and sense of humor. I am also grateful to Takayoshi Yamashita for his encouragement and providing valuable feedback. I would like to offer my special thanks to Masayoshi Tomizuka for allowing me to stay in University of California, Berkeley for a year, and giving me chances to start new research themes and study for them. I also want to thank Eiji Okada, Tadahiro Kuroda and Yasue Mitsukura for being co-advisors. I also want to thank Ryota Arai for cooperating on some experiments described in this thesis and all the current and former members of Aoki media sensing laboratory for contributing to a truly great and fun research environment and for many interesting discussions. I also would like to offer my special thanks to Masaaki Mochimaru and Makiko Kochi for giving me many great advices, motivating me, and sharing their philosophy with me when I first entered into the world of research. My first research theme was completely different from this thesis, but I learnt how to design a good story and build a vision to make the theme impactful through the experience on the collaborative researches with them, and the experience is surely utilized in this thesis. I also want to thank Yuji Yoshida and Kohei Yamamoto for being my irreplaceable friends and for many interesting discussions, and for encouraging me a lot. I am also grateful to Tokushi Suzuki for giving me many opportunities to see exciting and varied people. He made me realize the diversity in this world, and then made me feel free to live my life.

I am deeply grateful to my parents, Kimitaka and Emi, and my sister Nanae, for their encouragement and never-ending support. I'm really proud of them. Finally, I would like to thank my wife and the best friend, Nao, for her constant support and for putting up with me over the years.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Aerial imagery is the term that we use to describe photographs that are taken from the sky. Aerial imagery is a important information resource for various applications, for example, map making, urban planning, land use analysis, environmental monitoring, real estate management, and disaster relief, etc. However, to use aerial imagery for those applications, we preliminarily need to extract necessary information from given aerial imagery. Therefore, aerial imagery interpretation as a pre-processing is inevitable. In concrete terms, aerial imagery interpretation is a task to extract ground objects (e.g., buildings and roads) from an aerial image such as shown in Figure 1-1. This task has been usually performed by human experts manually, so that the process has been very costly and time-consuming. Therefore, if aerial imagery interpretation is completely automated, it will produce a great economic effect and enable more

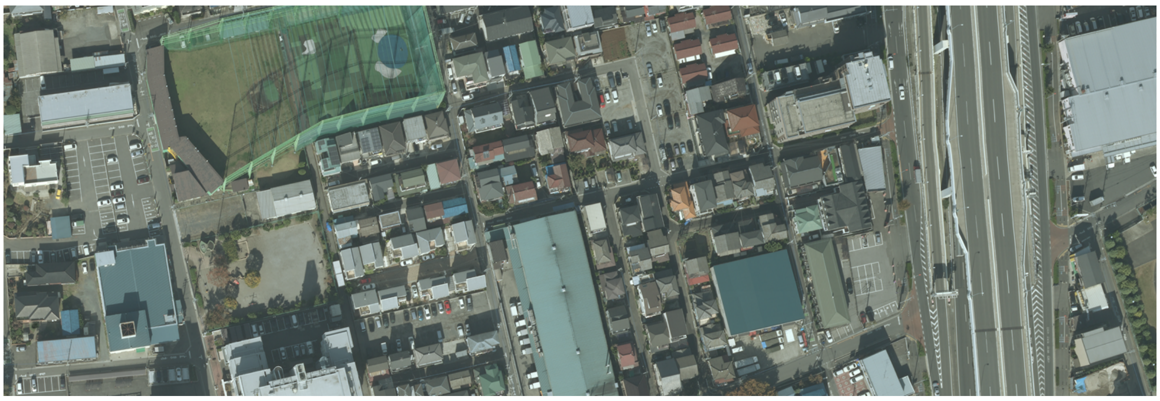


Figure 1-1: An example of high resolution aerial image

varied applications to be achieved. For example, the self-driving car is finishing testing stages in laboratories, so that much frequent updates of road maps based on aerial imagery interpretation began to be necessary.

## 1.1 Automation of Aerial Imagery Interpretation

There have been many applications that use aerial imagery as mentioned above, so that there have also been many attempts at automating aerial imagery interpretation. The early work that used a pattern recognition system to recognize ground objects with computer was proposed by Idelsohn [Idelsohn, 1970]. This system performs adaptive filters on input signals to extract seven binary features, and then classifies them into one of the terrain type classes. After that study, Bajcsy et al. investigated a model of roads to recognize road-like objects that appear in satellite images [Bajcsy and Tavakoli, 1976]. Then, some other knowledge-based and rule-based approaches were also proposed [Kettig and Landgrebe, 1976, McKeown et al., 1985]. These early works were strongly motivated to automate the object extraction task, but there were strict constraints and assumptions to use those systems in practical situations. Therefore, after the dawning age, significant progress has been made for about thirty years. Emmanuel proposed a knowledge-based object extraction method for aerial imagery, and achieved a semi-automated system [Emmanuel, 2004]. Then Mayer further discussed the knowledge-based approach by focusing on statistical modeling [Mayer, 2008]. These approaches seemed to be promising but essentially needed specialized knowledge about target objects which is to be extracted from aerial imagery preliminary to build the system. Therefore, human experts still should work to construct the knowledge structure for those ground objects which appear in the target aerial imagery.

### 1.1.1 Feature Extraction for Image Recognition Tasks

In general, to know what the important cues are to extract objects from images, many kinds of general purpose feature extractors have been studied and used in computer vision literature [Lowe, 1999, Bay et al., 2006, Dalal and Triggs, 2005, Weldon et al., 1996]. Once good features are acquired by good extractors, we can comparatively easily classify the features instead of data themselves. Namely, the features are better cues to know that a focused region in an input image is one of the objects of interest or not. Therefore, feature extractors generally aim at representing local regions in an image with discriminative representation by applying predefined filters, performing specific procedures that extract specific patterns and edges around a target pixel, or converting raw data into another feature space such as frequency space, etc. Thus, finding important cues for recognition tasks finally has been usually addressed as problems of how to choose and combine some local feature extractors to represent data with discriminative representation. However, those local feature extractors and how to combine them for better classification performance should be essentially designed based on the problem settings and data characteristics. Therefore, another kind of manual procedure by human experts is required, after all. On the other hand, data-driven construction of feature extractors using a convolutional neural network (CNN) is attracting attention in computer vision and machine learning field because of its high performance especially since [Krizhevsky et al., 2012]. Although it sill needs to design the network architecture usually by trial-and-error, if we have enough amount of data, a CNN performs very well on image recognition tasks by constructing good feature extractors automatically and classifying those features effectively within a single network.

### 1.1.2 Large-scale Machine Learning for Aerial Imagery

Focusing on some aspects of recent aerial imagery data, first, the resolution of aerial imagery is increasing because of improvement of camera sensors. Second, the total amount of available data is exploding. The first point seriously demands automation of aerial imagery interpretation because images that have higher resolution make manual procedures of object extraction more complicated and highly attentional for humans. Therefore, the cost to interpret them is also increased, while the high resolution aerial imagery enable more varied applications to be realized. On the other hand,

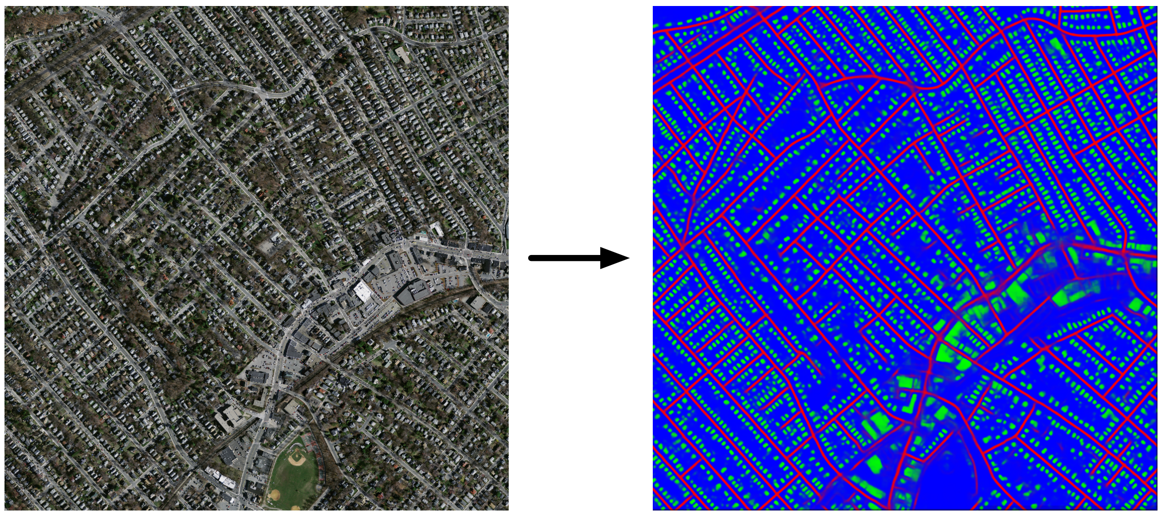


Figure 1-2: An example of semantic segmentation

the second point sheds light on machine learning approaches that utilize large-scale datasets to address those problems. Indeed, recent studies on large-scale machine learning for high resolution aerial imagery have proposed ground object detectors with impressive levels of accuracy [Kluckner and Bischof, 2009, Kluckner et al., 2009, Mnih and Hinton, 2010, 2012].

The object extraction from aerial imagery is usually formulated as a pixel labeling task in those approaches based on machine learning. Given an aerial image such as shown in Figure 1-1, the goal is to assign one of semantic labels of interest (e.g. buildings, roads, trees, grass, and water [Kluckner and Bischof, 2009, Kluckner et al., 2009]) to every pixel in an aerial image, namely, to perform semantic segmentation. For example, Figure 1-2 shows an input aerial image and an example result. Blue, Red, and Green pixels in the right picture denote background, building, and road classes, respectively. This is an example of multiple object extraction, but some studies focused only on a single object that appears in aerial imagery [Dollar et al., 2006, Mnih and Hinton, 2010, 2012].

In general, the pixel labeling is one of the most challenging tasks in the field of computer vision, so that it has been extensively studied [He et al., 2004, Shotton et al., 2008, Farabet et al., 2012]. Recent approaches utilizing deep CNNs achieve significant progress on semantic segmentation for general scenes [Long et al., 2015, Zheng et al., 2015] (e.g., images taken from vehicle video system, surveillance cameras, and daily snapshots, etc.) However, there is still remaining distance to human-level results. On the other hand, in the case of aerial imagery, there is relatively higher feasibility than the case of such general scenes. Because aerial imagery are usually transformed to orthoimages after being taken, so that the view direction of aerial imagery is always orthogonal to the ground plane, and the scale of objects that appear in aerial imagery can be assumed to be fixed within the images that have the same resolution. Those prior conditions reduce the possible variations which should be considered, so that semantic segmentation for aerial imagery is a more tractable task compared to the general image labeling task.

Thus, there have been some methods that address semantic segmentation for aerial imagery by utilizing machine learning approaches combined with large-scale datasets. The state-of-the-art in the aerial imagery labeling is the work by Mnih [Mnih, 2013]. In that paper, a large-scale dataset for the aerial imagery labeling is proposed, and CNNs are trained separately on the dataset for road extraction and building extraction. They further focused on label noise and structured prediction using conditional random fields (CRFs) or post-processing neural networks, but multiple object extraction at the same time is not considered. However, both roads and buildings are very important ground objects for various practical applications. The methods proposed by Mnih [Mnih, 2013] can be applied for any types of objects when labels of all types that we want to extract are available, but a different model is needed to be trained for each target object. It means that the resulting models for different objects are independent from each other. Therefore, if there would be correlation between different objects, each model cannot be utilized it for label prediction. However, there obviously is a mutual exclusiveness between different objects. For example, a pixel that belongs to the road class can never belong to the building class, vice versa. If the exclusiveness can be utilized to improve performance within a single network, there will be advantages not only on the number of models needed to be trained to extract multiple objects but also on performance based on utilization of the correlation.

### 1.1.3 Multiple Object Extraction

In this thesis, we aim at training a single model for simultaneous extraction of multiple objects, roads and buildings, by utilizing the correlation between them. This problem setting is more difficult than the single object extraction problem, so that a naive extension of the conventional model for single object extraction cannot perform well. The conventional method [Mnih, 2013] has a single-channel predicted label image as its output, so the simplest way to extend it for multiple objects is to make the output a multi-channel image. However, it cannot achieve better performance compared to the conventional model that has a single-channel output. To output a multi-channel image with considering the mutual exclusiveness between different objects of interest, a straightforward way is to make a pixel on the output image a vector whose elements add up to one. Then, to design a pixel on the output image as such a vector, we should take a background class into account as one of the classes of interest. Therefore, when we predict two types of objects, roads and buildings, we should consider three types of classes, roads, buildings, and the background that is defined as the none-of-the-above class. Here, the background is treated completely same as the other classes, but it is obviously special. Because the background class fundamentally includes multiple classes in different levels of abstraction, so that the features to represent the concept of the background must be more complicated compared to the other simple objects such as roads and buildings. Therefore, we focus on this specialty. We propose a novel activation function for the final layer of a CNN to consider the specialty of the background class and train the CNN efficiently for multiple object extraction.

## 1.2 An Application of Aerial Imagery Labeling

Once the automatic object extraction from aerial imagery is achieved with high accuracy, there are many possible applications which benefit significantly from the automation. One of the simplest applications of aerial imagery interpretation is seamline determination for aerial imagery mosaicking. Image mosaicking is a task to combine two or more images that have overlapping regions into a single large image. To perform aerial imagery mosaicking, a boundary line between a set of smaller images to be combined should avoid to cross any buildings to make the resulting mosaic natural. The boundary line is called *seamline.* The optimal seamline yields a larger image with seamless boundaries of which nobody can aware. Therefore, this task is usually done by human experts manually by choosing all pixels that construct a seamline, so that the automation of this task is also demanded. In this thesis, we show that the optimal seamline that crosses no buildings can be easily found automatically by utilizing semantic segmentation results.

## 1.3 Contribution

Our goal is to develop an accurate system for automatic semantic segmentation of aerial imagery based on large-scale machine learning approaches and to show the effectiveness of the system on an application of aerial imagery processing. In this thesis, we focus on what we see as the room for improvement in applying image labeling techniques to aerial imagery, and on possible applications of semantic segmentation:

- *Simultaneous extraction of multiple objects*: There are not as many methods for extracting multiple objects simultaneously as for extracting each object separately, although there are many different ground objects in aerial imagery, and applications (e.g., automated map making) cannot be achieved by extracting only a single object. Furthermore, the occurrence of a ground object could be correlated with another object, especially in the case of buildings and roads in urban scenes. Therefore, we assume that the consideration of multiple objects simultaneously to exploit the latent correlation will improve the performance of pixel labeling. Thus, a new technique to train a CNN efficiently for simultaneous multiple object extraction is proposed. The main contribution of this thesis is the technique, a novel activation function for the last layer of a CNN that can suppress the effect of the *background* class that we cannot avoid to consider in multiple object extraction.

- *Seamless results in patch-based approach*: Typical semantic segmentation approaches that utilize CNNs can achieve the state-of-the-art accuracy for aerial imagery, but the fundamental limitation of those approaches is that the size of the receptive field in the input image layer is too small compared to the whole size of an aerial image. Thus, it is inevitable to apply the patch-based approach when we use the conventional network architecture for this task. However, the naive patch-based approach causes that a resulting predicted label image should be synthesized by tiling many predicted label patches, so that the boundaries of patches may not be smooth and degrade the resulting performance. Therefore, we propose a new model averaging method that can avoid producing such patchy results without complicated structured prediction methods such as a conditional random field but with ability to achieve high accuracy.

- *Automatic seamline determination by utilizing semantic segmentation results*: If semantic segmentation for aerial imagery could be performed very accurately, there are many possible applications that benefit from the results. One of such applications is seamline determination [Pan et al., 2014]. A seamline is the boundary between two aerial images that will be combined into a single image to synthesize a larger aerial image. The optimal seamline is defined as a path that never passes through any buildings in those two aerial images. We show the conventional seamline determination method based on Dijkstra's algorithm [Dijkstra, 1959] to calculate the minimum cost path on a preliminary generated cost map can achieve the optimal result when the semantic segmentation result is used as a cost map, while the conventional cost map that is generated based on a mean-shift segmentation method leads the resulting seamline to cross buildings.

The rest of this thesis includes the work previously published in [Saito et al., 2015, 2016], and is organized as follows:

- *Chapter 2* presents the details of related work that utilize machine learning, and summarizes the relationship to our approach by describing the difference between them. Then, an overview of components of a CNN that is the core of our approach is presented. Finally, the related work of a typical application, aerial imagery mosaicking, are presented.

- *Chapter 3* presents the details of the proposed methods. We propose a method to extract buildings and roads simultaneously and accurately from aerial imagery by training a single CNN with a novel activation function for the final layer of the network. The training is performed on a large scale dataset that has many aerial images and corresponding building and road mask images. The new activation function is designed by focusing on the specialty of the background class.

- *Chapter 4* presents empirical evaluation of our method on a large-scale aerial imagery dataset. The comparison results of the effectiveness to the conventional methods are presented. Then we show that the proposed techniques for multiple object extraction have advantages on a simple extended model of the conventional CNN.

- *Chapter 5* presents discussions about the results of the empirical evaluation shown in Chapter 4 to clarify the effectiveness of the trained CNN with the proposed techniques presented in Chapter 3 by visualizing the middle layer outputs and investigating the trained parameters. Then we further discuss about the size of dataset that is required to train a CNN successfully for this task.

- *Chapter 6* shows that the aerial imagery mosaicking can be easily automated by using building extraction results that are obtained by utilizing a trained CNN. We evaluate the resulting seamlines by comparing to results that are obtained by a conventional method based on mean-shift segmentation. Then we show that our approach can achieve the optimal results.

- *Chapter 7* summarizes all techniques and methods proposed in this thesis. Then the remarks on the most important findings are presented. Finally, conclusion and future work of this research are shown.

# Chapter 2

# An Overview of Aerial Imagery Interpretation and Its Applications

Aerial imagery interpretation has many applications as described in Chapter 1. In many cases, the goal of aerial imagery interpretation is to extract ground objects from aerial imagery. Then, ground object extraction is basically performed by labeling each pixel or segment in an aerial image with one of the classes of interest (e.g., buildings or roads), so that this is a complex attentional task for humans. Because, ground objects have a great deal of variation in their shapes and appearances, and an object may be occluded by other objects such as trees and also by buildings' shadows. Therefore, automatic extraction of roads and buildings is highly demanded and many attempts have been proposed in the remote sensing literature. This chapter aims to present an overview of the approaches that utilize machine learning for this task and a brief introduction of the important component of the proposed method, convolutional neural networks (CNNs). Then we focus on the task of seamline determination as an example application of automatic aerial imagery labeling, so the related work of the task is overviewed.

## 2.1 General Segmentation Approaches

The objective of semantic segmentation is to assign a semantic label such as a road and a building to every pixel in an image. In general, segmentation is performed by many ways in terms of how to divide an image into smaller regions or segments. For example, some color segmentation methods such as mean-shift segmentation or simple linear iterative clustering (SLIC) [Achanta et al., 2012] are performed by using color similarity to merge neighboring pixels into a small segment or a super pixel, so that the labels actually assigned to pixels are approximated color information.

On the other hand, semantic segmentation aims at assigning semantic information to each pixel. For example, Rother et al. proposed an interactive foreground extraction method using graph cuts that is called GrabCut [Rother et al., 2004]. In this method, the user needs to give some initial clues for foreground and background pixels, while it can achieve an accurate semantic segmentation result after optimizing process by graph cuts. However, it is based on the spatial consecutiveness of a same object in general pictures, so that it is difficult to apply this method to the aerial imagery labeling task. Because a ground object could be enclosed by another object, different objects that belongs to a same object class could be appeared in distant location. Therefore, the aerial imagery labeling task basically cannot be a simple foreground or background segmentation problem. We should consider a non-interactive segmentation approach that can be applied to different pixels separately.

## 2.2 Early Work on Terrain Classification

Some of the first machine learning-based land use classification methods for multi-spectral and radar images were proposed in late 1980s and in 1990s [Decatur, 1989, Benediktsson et al., 1990, Bischof et al., 1992, Paola and Schowengerdt, 1995], which classify a feature vector $\mathbf{x}_i$ at a pixel $i$ to a discrete class label $\mathbf{c}_i$. In much of those pixel-by-pixel classification work, neural network-based approaches were usually introduced with the comparison to the Bayes classifier. In those work, the Bayes classifier takes a raw pixel value on a multispectral or radar image as an input feature $\mathbf{x}_i$, then the posterior class probabilities $P(\mathbf{c}_i = k|\mathbf{x}_i)$ are predicted using Bayes' rule. Bayes' rule needs assumptions about the class-conditional distributions $p(\mathbf{x}_i|\mathbf{c}_i = k)$ and the prior class probabilities $P(\mathbf{c}_i = k)$. Therefore, the conventional methods assumes that $p(\mathbf{x}_i|\mathbf{c}_i = k)$ have a multivariate normal distribution with mean $\mu_k$ and covariance $\Sigma_k$. Under this assumption, the classifier can learn only linear or quadratic decision boundaries. Especially, when $\Sigma_k$ is diagonal, the classifier turns out to be the naive Bayes classifier with assuming $P(\mathbf{c}_i = k) = 1/K$, and it was used in [Paola and Schowengerdt, 1995].

On the other hand, neural networks can directly model the posterior class probabilities $P(\mathbf{c}_i = k|\mathbf{x}_i)$ without any assumptions by approximating the mapping between input and output with a differentiable objective function that is constructed with trainable parameters [Decatur, 1989, Lee et al., 1990, Bischof et al., 1992]. Therefore, it can sidestep the need to specify the class-conditional distributions $p(\mathbf{x}_i|\mathbf{c}_i = k)$. The neural networks for this task should have at least one hidden layer with non-linear activation functions to learn non-linear decision boundaries [Decatur, 1989, Benediktsson et al., 1990], but all the investigations presented in the previous papers pointed out aspects of the usefulness of neural networks in the classification of pixels on aerial imagery.

One of the drawbacks of those pixel-by-pixel prediction approaches is the lack of consideration of contextual information. Then, some work used spectral values from a small patch (e.g., $7 \times 7$-sized local region) as the input features to their neural network models [Bischof et al., 1992, Boggess, 1993]. The others tried to design hand-crafted features and used them to encode the contextual information [Haralick et al., 1973, Haralick, 1976, Lee et al., 1990]. The feature proposed by Haralick [Haralick, 1976] represents local textural information by encoding the spatial relationship between distant pixels which have the same gray level values.

Those methods could achieve high accuracy when the input aerial images are low resolution images. For example, Bischof et al. [Bischof et al., 1992] reported a classification accuracy of 85.9 % on a classification task of four classes, built-up land, agricultural land, forest, and water, but the resolution of input multispectral images was $30 \times 30$ m$^2$/pixel. If the objective classes are defined from such macroscopic viewpoint for land use analysis, those approaches were sufficient because the input multispectral images essentially are effective features to represent such classes. However, the resolution of recent aerial imagery is getting more than 900 times higher. In this situation, the objective classes are changed to finer object classes such as roads, buildings, cars, and trees, etc. Then, such ground objects in high resolution aerial imagery are difficult to be extracted only by using local cues obtained through deterministic procedures used in the above previous methods.

## 2.3 Object Extraction from High Resolution Aerial Imagery

The Ikonos and Quickbird satellites are launched in 1999 and 2001, respectively. These satellites for remote sensing can take high resolution images of the surface of the Earth. Once those high resolution aerial imagery are available, the classes of interest are shifted from macroscopic land types to smaller ground objects. To extract those objects (e.g., buildings and roads) that appear in high resolution aerial imagery, the methods that use local spectral images and textural features with simple classifiers are insufficient in terms of accuracy. Because roads and buildings may have similar textures, classifying them is a more difficult task than discriminating land use types, for example, distinguishing a forest area from a built-up area using low-resolution aerial imagery. Therefore, to classify the more finer ground objects successfully, varied types of feature extractors are used with more sophisticated classifiers such as SVMs [Vapnik, 1963], random forests [Breiman, 2001], or various types of boosting methods such as AdaBoost [Freund and Schapire, 1997], which can find complex decision boundaries. Other than these, many fusion methods to improve the representational power of local image features and to sophisticate classifiers have been developed.

### 2.3.1 Feature Engineering and Decision Fusion

Sirmacek et al. [Sirmacek and Ünsalan, 2011] proposed a probabilistic framework to detect buildings, utilizing four different local feature extractors. They separately used a Harris corner detector [Harris and Stephens, 1988], gradient-magnitude-based support regions (GMSRs) [Ünsalan, 2006], Gabor filtering in different orientations [Vetterli and Kovačević, 1995], and features from accelerated segment test (FAST) [Rosten et al., 2010] to extract feature vectors from aerial imagery and obtain four different estimation results of building locations. Some parameters of those feature extractors were adjusted independently for the dataset used in the article. Then they combined the separate estimation results from different features into a single building detection output by using data and decision fusion methods.

Senaras et al. [Senaras et al., 2013] also proposed a decision fusion method for building detection. They first performed mean-shift segmentation to an aerial image with a preliminary learned band width parameter, and then calculated the normalized difference vegetation index (NDVI) [Tucker, 1979] from the red color channel of the aerial image and the corresponding infrared image. The NDVI image was binarized with Otsu's method [Otsu, 1975] to extract vegetation segments. They also performed this binarization on a hue-saturation-intensity (HSI) image converted from a three-channel image consisting of infrared-red-green to extract shadow regions. Using the results of these pre-processings, both vegetation and shadow segments were excluded from the group of all candidate segments. To classify the remaining segments into building class or background, they extracted 15 different image features from each segment. Then, they trained 15 different classifiers with those features and classified each segment by the 15 classifiers independently and obtained 15 decisions for a single segment. Finally, all decisions for a segment were combined by utilizing fuzzy stacked generalization [Ozay and Vural, 2012].

These methods based on decision fusion have achieved accurate extraction of ground objects from aerial imagery by constructing complicated features and multiple-stage classifiers. However, those local image features are specially designed for extracting a specific object, and the fusion techniques of multiple classification results are also intended to extract a specific object.

There are not as many methods for extracting multiple objects simultaneously as for extracting each object separately, although there are many different ground objects in aerial imagery, and applications (e.g., automated map making) cannot be achieved by extracting only one object. Furthermore, the occurrence of a ground object may be correlated with other objects, especially in the case of buildings and roads in urban scenes. Therefore, if the simultaneous extraction of multiple objects can be achieved by utilizing the correlation to improve the performance, it will be a important progress on aerial imagery interpretation.

### 2.3.2 Neural Networks with Large-scale Datasets

Extracting features hierarchically and combining many decisions into one output by weighting them are suited to be represented by a single feed-forward neural network architecture. Furthermore, a neural network has a powerful online training scheme such as stochastic gradient descent [Bottou, 1991], so that it can be easily trained on a large-scale dataset.

Approaches utilizing artificial neural networks trained on a large and high resolution aerial image dataset to solve the semantic segmentation problem have achieved good performance. Mnih et al. [Mnih and Hinton, 2010] proposed a road extraction system utilizing a restricted Boltzmann machine (RBM) [Smolensky, 1986]. They formulated the problem of extraction of road pixels from aerial imagery as a patch-based semantic segmentation task. An input aerial image is divided into $64 \times 64$ patches, and principal component analysis (PCA) [Pearson, 1901] is applied to them to reduce the dimensionality. Then those PCA vectors are used to fine-tune an RBM that has been pre-trained in an unsupervised way [Hinton and Salakhutdinov, 2006]. This RBM predicts a road probability map from a PCA vector of an aerial imagery patch. They finally trained a post-processing network that refines the predicted probability maps to incorporate structures such as road connectivity into the final outputs. They evaluated the performance of their method with a large dataset that consisted of aerial imagery and corresponding binary road label images. The dataset covered roughly 500 km$^2$.

This RBM-based road extraction method was updated by replacing the RBM with a locally-connected neural network and incorporating two different noise models [Mnih and Hinton, 2012]. They considered two types of noise occurring in label images, omission noise and registration noise. The former occurs when an object that appears in aerial imagery does not appear in the corresponding label image. The latter occurs when the location of an object in a label image is inaccurate. They proposed asymmetric Bernoulli distribution and translational noise distribution to handle these two types of noise. Finally, they showed the effectiveness of the noise models and achieved the good accuracy in road extraction. However, in the thesis by Mnih [Mnih, 2013], he concluded that label noise has a negative but relatively small effect on prediction results in the methods that utilize neural networks because of its powerful representational ability.

### 2.3.3 Patch-based Approaches with Convolutional Neural Networks

Mnih [Mnih, 2013] proposed a patch-based pixel labeling method that uses a convolutional neural network (CNN) to learn the mapping from raw pixel values to predicted label images directly. In the patch-based approach, the input image is a three channel small patch which is $64 \times 64$-sized Red-Green-Blue (RGB) image, and the output is a single channel small patch which is $16 \times 16$-sized grayscale image whose pixel values represent road or building probabilities. In that thesis, not only the base CNN model is proposed and tested, but also how to incorporate structural dependencies with the final prediction results is discussed. Their approach for structured prediction is two-fold, one is to use a conditional random field (CRF) to learn the pixel dependencies in the output label image. Another is to train post-processing networks to refine the output of the base CNN, so that the system has a cascading structure. These attempts at structured prediction make the performance better. Therefore, in the patch-based formulation of semantic segmentation for large aerial imagery, how to incorporate the consecutiveness of object appearance and the context with the final prediction results is very important.

Furthermore, a CNN can successfully predict the label images from raw pixel values when it has been trained on large-scale datasets, but there is a limitation of the size of input images because of a practical problem of memory limitation of a graphics processing unit (GPU) that is usually used to train a neural network. Thus, an input aerial image should be divided into small patches, so that the network predicts the labels for each patch. Hence, the resulting outputs may be very patchy because the large jumps of predicted values can appear around the boundaries of neighboring predicted patches. To address this problem of discontinuity around the boundaries, the structured prediction approaches are required, and in deed, it improves the performance.

### 2.3.4 Our Approach for Multiple Object Extraction

In this thesis, we aim at automatic acquisition of good feature extractors for both buildings and roads, which are applied to raw pixel values directly with no preprocessing. We achieve this by utilizing a CNN that is trained on a publicly available large-scale aerial imagery dataset [Mnih, 2013]. We show that our CNN can also classify all pixels in aerial imagery into buildings, roads, or the background more accurately than previous work [Mnih and Hinton, 2010, 2012, Mnih, 2013]. Our method does not need to design image features manually. The training of multiple classifiers independently for each ground object that is to be extracted is also not needed. Therefore, our method does not need to consider how to fuse multiple decisions, and the output of our CNN inherently constructs three-channel label images (buildings-roads-background). At the present stage, using a CNN seems to be the best way when a large-scale dataset is available. On the other hand, the multiple object extraction task is more difficult than the task to extract a single ground object. Thus, the result of the naive extended version of the conventional CNN model, which has multiple-channel output, cannot surpass the accuracy that achieved in [Mnih, 2013].

Then, we propose a new activation function for the final layer of a CNN that can train the CNN for multiple object extraction task more effectively by focusing on the specialty of the background class. By training a CNN with our proposed techniques, *multiple object extraction by a single model with high accuracy* is achieved. There are some advantages of our proposals. Achieving high accuracy is essentially important for the practical objectives in various applications as mentioned in Chapter 1. Furthermore, extracting multiple objects simultaneously is also effective on many practical situations, because we should keep only one model for multiple objects, and we can obtain the multiple object predictions using the model only once, namely, it needs a single feed-forward computation. Then, we further propose a new model averaging technique to avoid the patchy results. We show the effectiveness of those techniques in empirical evaluation experiments.

## 2.4 Convolutional Neural Networks

In this section, a brief introduction of convolutional neural networks is presented. In this thesis, it is the most important model to learn the mapping from input images to output labels that we want to predict to perform semantic segmentation for aerial imagery.

In recent years, convolutional neural networks have attracted much attention in the computer vision area. A convolutional neural network (CNN) can be trained as robust feature extractors from raw pixel values and, at the same time, learn classifiers for object recognition tasks [Krizhevsky et al., 2012], regressors for human pose estimation tasks [Toshev and Szegedy, 2014], or mappings for semantic segmentation tasks [Farabet et al., 2013, Long et al., 2015, Zheng et al., 2015]. The CNN is a biologically inspired variant of a multi-layer perceptron. The base idea was introduced by Fukushima [Fukushima, 1980] as a neural network model for visual recognition tasks. The model, Neocognitron, stacks convolutional layers and pooling layers alternately and is trained in unsupervised manner. These layer architectures are inspired by the receptive fields and the hierarchical structure in the cat's visual cortex found by Hubel and Wiesel [Hubel and Wiesel, 1962]. After that, LeCun et al. [LeCun et al., 1989, 1998] succeeded to train a hierarchical neural network model that is inspired by Neocognitron with a classical gradient descent-based parameter optimization method called backpropagation for hand-written digits classification. Backpropagation has been used to optimize parameters of multi-layer perceptrons since the work of Rumelhart et al. [Rumelhart et al., 1988], but stochastic gradient descent with backpropagation is also effective for optimizing a CNN with weight sharing. The model used for this task was the first appearance of the modern CNN.

After over ten years, a CNN was focused again in 2012. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Russakovsky et al., 2015] was held in the year, which is a competition whose goal is to estimate the content of photographs for the purpose of retrieval and automatic annotation using a very large hand-labeled ImageNet training dataset that consists of over 1,280,000 labeled images depicting 1,000 object categories. The winner in the competition in 2012 used a deep convolutional neural network that has eight layers [Krizhevsky et al., 2012]. The competition has been continued and the winner's model is getting deeper and deeper every year. In this 1000 class classification task, the error rate of human raters is known as about 5.1 % [Russakovsky et al., 2015], but in 2015, some deep convolutional neural network models have surpassed the human-level accuracy [He et al., 2015a, Ioffe and Szegedy, 2015, He et al., 2015b]. Then, the state-of-the-art in this 1000 class image classification task achieved the error rate of 3.57 %, and the model used in the work had 152 layers [He et al., 2015a]. To achieve those results, many techniques have been proposed to train a CNN successfully on such large dataset. For example, ReLU [Fukushima, 1980, Nair and Hinton, 2010] and PReLU [He et al., 2015a] have been proposed as alternative activation functions to the logistic sigmoid or the tanh function. The inception module [Szegedy et al., 2015], batch normalization [Ioffe and Szegedy, 2015], and deep residual learning [He et al., 2015b] have been proposed as a new network component, a normalization layer, and a new architecture for more deeper networks, respectively. However, the fundamental component of those modern CNNs is still a convolutional layer that has the same characteristics as the one proposed in [LeCun et al., 1989] and the optimization procedure is basically the conventional stochastic gradient descent, though many sophisticated optimizers have been proposed (e.g., AdaGrad [Duchi et al., 2011], AdaDelta [Zeiler, 2012], and Adam [Kingma and Ba, 2014], etc.)

### 2.4.1 The Overview of Components of a Convolutional Neural Network

A convolutional layer has fixed sized filters. Let $M$, and $H \times H$ be the number of the convolution filters, and the size of the filters, respectively, a convolutional layer takes a $K$-channel $W \times W$-sized image as its input and outputs a $M$-channel $(W - H + 1)/s \times (W - H + 1)/s$-sized image, where $s$ is a stride parameter. The stride means a step of convolution over the input $W \times W$-sized image. Then, each channel in the output image is called a feature map.

Let $z_{ijk}$, $h_{pqkm}$, and $u_{ijm}$ be a pixel value at $(i, j)$ in $k$-th channel in the input image, a weight value at $(p, q)$ in $m$-th convolution filter, which is connected to $k$-th channel in the input image, and a pixel value at $(i, j)$ in $m$-th feature map, respectively, $u_{ijm}$ is calculated as:

$$u_{ijm} = \sum_{k=0}^{K-1} \sum_{p=0}^{H-1} \sum_{q=0}^{H-1} z_{s \cdot i+p, s \cdot j+q, k} h_{pqkm} + b_{ijm}, \tag{2.1}$$

where $b_{ijm}$ is a bias parameter to be added to a weight value at $(i, j)$ in $m$-th convolution filter. It should be noted that all $b_{ijm}$ with different $(i, j)$ have the same value in many cases. The convolutional layers also have tied weights, so that $h_{pqkm}$ is shared among all $(p, q)$ for reducing the number of parameters. If the stride $s > 1$, filters are convolved at intervals of $s$, so that the sizes of all feature maps are $(W - H + 1)/s \times (W - H + 1)/s$. Then, activation function $f$ is applied to the resulting feature maps $u_{ijm}$. Therefore, the output of a convolutional layer is

$$z_{ijm} = f(u_{ijm}). \tag{2.2}$$

While the logistic sigmoid function $f(u_{ijm}) = 1/(1 + \exp(-u_{ijm}))$ is well known as a typical activation function for neural networks, we use $f(u_{ijm}) = \max(u_{ijm}, 0)$ instead in this thesis. The units that have the latter activation function with the max operation are called rectified linear units (ReLU) and the effectiveness for convergence performance and learning speed is reported in [Nair and Hinton, 2010]. In a convolutional layer, $h_{pqkm}$ and $b_{ijm}$ are learnable parameters, so we optimize them in the training stage of the CNN. In the following part of this thesis, we describe a convolutional layer that has $M$ filters with the size of $H \times H$ and stride $s$ by $C(M, H \times H/s)$.

A pooling layer takes feature maps from the convolutional layer in the next lower layer and performs subsampling to them. We use max pooling for the pooling layer of our CNN. The pooling layer performs this calculation:

$$u_{ijk} = \max_{(p,q) \in P_{ij}} z_{pqk}, \tag{2.3}$$

where $P_{ij}$ denotes a set of pixels in the $H \times H$-sized region centered at $(i, j)$ in the input feature map. A max pooling operation outputs the max value in the receptive field $P_{ij}$, and this operation is applied at intervals of $t$. Accordingly, the input $K$-channel $(W - H + 1)/s \times (W - H + 1)/s$-sized feature map is downscaled to the size of $(W - H + 1)/(s \cdot t) \times (W - H + 1)/(s \cdot t)$. Here, it should be noted that when $t = 1$, the pooling layer does not change the size of input. Additionally, in the max pooling layer, there is no learnable parameter. In the following part of this thesis, we describe a pooling layer with $H \times H$-sized receptive field and stride $t$ by $P(H/t)$.

The characteristics of a CNN are alternatively stacked convolutional layers and spatial polling layers, often followed by one or more fully-connected layers as in a multi-layer perceptron. All units in a fully-connected layer connect to all units in the next lower layer. We describe a fully-connected layer with $N$ units by $FC(N)$.

As explained above in detail, a convolutional layer has a number of filters and convolves them on an input image for extracting features. A pooling layer applies subsampling to the output of the next lower layer for acquiring translational invariance. Figure. 3-3 shows the architecture which is used for semantic segmentation

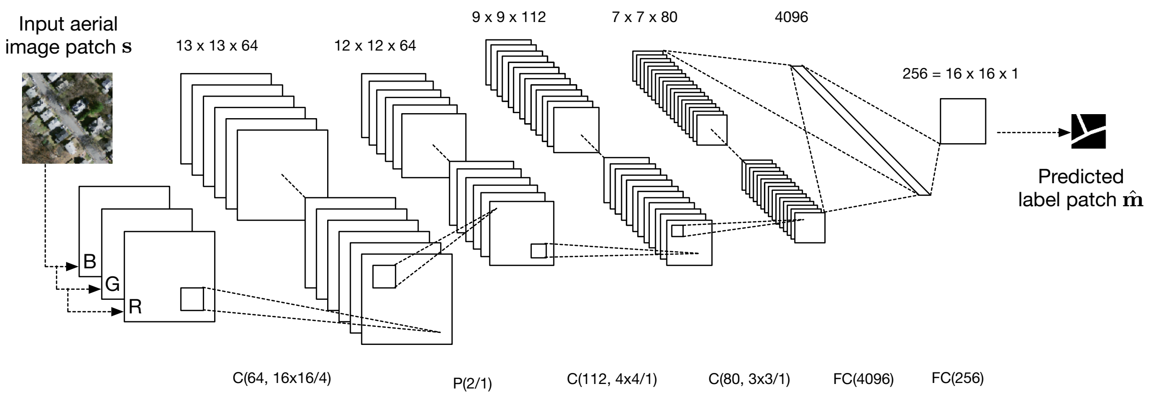


Figure 2-1: The architecture of a CNN used in the paper [Mnih, 2013]

for a single channel prediction in [Mnih, 2013]. This architecture is described as $C(64, 16 \times 16/4)$ - $P(2/1)$ - $C(112, 4 \times 4/1)$ - $C(80, 3 \times 3/1)$ - $FC(4096)$ - $FC(256)$.

## 2.5 An Application of Semantic Segmentation: Aerial Imagery Mosaicking

In this section, to show the demand of automatic semantic segmentation for aerial imagery on practical applications, we focus on an example application, seamline determination for aerial imagery mosaicking. Then the related work of automatic seamline determination are presented.

Essentially, there are many possible applications that use large aerial images. However, the size of aerial images that can be captured by a camera at one time is limited by the sensor size. Therefore, aerial images are usually captured with overlapping regions with each other and tiled using the overlaps to construct a large aerial image. The important thing when tiling the images is that the boundaries seem to be natural, which means the boundary lines never cross any buildings. Buildings are generally higher than digital elevation model (DEM) that represents the ground elevation, and so have different appearances in aerial images captured from different viewpoints. If the boundary between two aerial images passes through a building, then the resulting mosaic image can have an unnatural appearance where they join as shown in Fig. 2-2. The red line in Figure 2-2b is the boundary line, and it is called

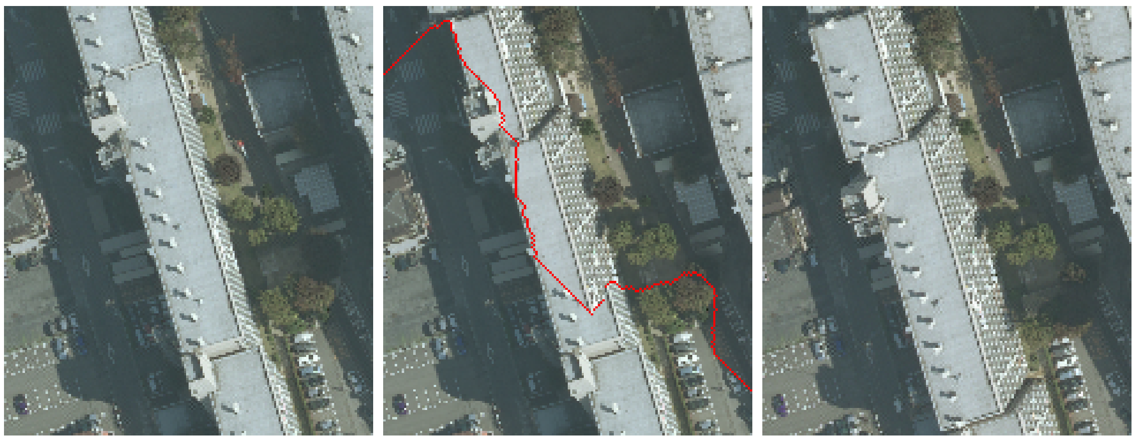


Figure 2-2: A result example of mosaic image with a wrong seamline that passes through a building: (a) The bottom side of an aerial image before mosaicking. (b) The resulting mosaic image based on the red line that shows a failed seamline which passes through a building. (c) The upper side of an aerial image before mosaicking.

a seamline in the remote sensing field.

### 2.5.1 Conventional Approaches to Aerial Imagery Mosaicking

There have been many attempts at automatic seamline determination. One major approach is creating cost maps from input aerial images, and then choosing pixels that pass through the minimum cost path. In the conventional work, creating cost maps depends on hand-crafted feature engineering. The simplest way to define the cost map is using the difference of two images. Milgram [Milgram, 1975] and Fernandez et al. [Fernandez et al., 1998] chose seam points by minimizing the differences between pixel intensities on the seamline.

This type of approach implicitly assumes that objects with significant height (e.g., tall buildings) will appear differently in an overlapping region because of the difference in viewpoint, so that small differences of pixel intensity at the same location might mean that the location is ground or a preferred region that can be passed through by a seamline. However, if there is a large building that spans a large area in an aerial image, the region around the center of the building may not produce such large differences when the appearance of the roof is flat even though it should not be crossed by the seamline.

Wang et al. [Wang et al., 2012] and Wan et al. [Wan et al., 2012] overlaid vector data of roads on an overlapping region of a set of aerial images to build a weighted graph, and calculated the lowest-cost path on the graph to obtain a seamline candidate. Then they finally refined the candidate by considering its surrounding pixels to fix the optimal seamline. Therefore, their method needs correct vector roads as necessary inputs, so that the input aerial images should have the additional data of vector roads.

Afek and Brand [Afek and Brand, 1998] presented an algorithm that integrates global feature matching algorithms into the process of selecting a seamline, so that they considered not only determining seamlines but also geometric transformation between two aerial images. This type of approach is efficient when the input aerial images are not aligned and not orthogonalized. However, if it can be assumed that input aerial images are always orthoimages and preliminary aligned, the geometric transformation is not needed to be considered as in the previous work [Pan et al., 2014].

Kerschner [Kerschner, 2001] defined an energy function focusing on color similarity and texture similarity over an input aerial orthoimage, and searched for the optimal seamline by using snakes [Kass et al., 1988], an active contour model. However, this approach still cannot consider the meaning of pixels. If the cost map is not explicitly designed to put higher values on building regions, the small values on the cost map could be found on building regions. It causes that the resulting minimum cost path on such cost map could potentially cross buildings, even if the seamline search method is too sophisticated. Therefore, how to create the cost map is more important problem in seamline determination.

As described above, there are many approaches that use various cost (energy) maps and optimization methods to determine seamlines. When human experts draw seamlines on overlapping regions of aerial image pairs, they focus on the object type represented by the pixels in the images. Therefore, automatically extracting pixels that belong to objects which should not be crossed by a seamline is desirable, if it can be provided with high accuracy. To extract such pixels, pixel labelling, namely, semantic segmentation of the aerial images is necessary.

The segmentation-based method for aerial image mosaicking has been presented in [Pan et al., 2014]. They proposed a seamline determination method based on a cost map calculated by mean-shift segmentation. However, mean-shift segmentation requires some sensitive parameters for implementation on input aerial images, such as a bandwidth parameter. Furthermore, this method requires preferred regions to be determined from input images using the results of mean-shift segmentation. The preferred regions are defined as segments larger than a specified size parameter $s_T$ that is chosen manually according to the actual size of the largest objects in the target aerial images. The preferred regions are expected not to include objects such as buildings because of the assumption that most objects are smaller than roads or grassed areas and thus belong to smaller segments. However, buildings are not always smaller than roads or fields, especially in urban areas.

Therefore, although segmentation is used in [Pan et al., 2014] to create cost maps, they did not consider the meaning of pixels because object-based recognition is a very difficult task, and instead applied mean-shift segmentation which only considers the similarity between colors. Furthermore, as mentioned in [Pan et al., 2014], the method depends strongly on the parameters selected for mean-shift segmentation. Then, another problem is that the size threshold for selecting preferred regions needs to be chosen manually for each input image.

### 2.5.2 Aerial Imagery Mosaicking Based on Semantic Segmentation

In this thesis, we aim at automatic determination of optimal seamlines that do not cross any buildings by utilizing the accurate semantic segmentation method proposed in Chaper 3. Our method needs only the input aerial images, so that we do not use any other resources such as road vector maps, because we extract buildings automatically by using a CNN that has been trained for semantic segmentation and solve the minimum cost path problem by Dijkstra's algorithm [Dijkstra, 1959] on the result of semantic segmentation as an input cost map. It should be noted that we achieve the difficult task, as mentioned in [Pan et al., 2014], of accurate object extraction from aerial imagery by training a CNN using a large-scale aerial image dataset. We show the optimal seamline is easily found by utilizing the semantic segmentation result as the cost map.

# Chapter 3

# Semantic Segmentation for Aerial Imagery

This chapter presents our main contribution on semantic segmentation for aerial imagery with a convolutional neural network (CNN) that has multi-channel outputs. Our contributions to this problem are threefold. First, we propose a multi-channel prediction method with a single CNN based on the patch-based formulation of semantic segmentation of aerial imagery [Mnih and Hinton, 2010]. We train the CNN that has a three-channel output layer and predict buildings, roads, and the background simultaneously. Second, we propose a new activation function for the final layer of a CNN to consider the specialty of the background class. Third, we propose a new model averaging method that can avoid producing patchy results such as the example shown in Figure 3-1. As shown in this figure, if we predict the labels in the patch-by-patch manner and tile them to construct the final results such as this figure, the boundaries between neighboring patches may have large jumps of predicted values and produce the non-contiguous and non-smooth results. It leads to a bad evaluation result and further manual processing, but it cannot be solved by simply smoothing the output values. Therefore, we propose another approach to address this problem other than the approaches using post-processing networks proposed in [Mnih, 2013], which need further training stages.

The first and second proposals are deeply related with each other. The labels

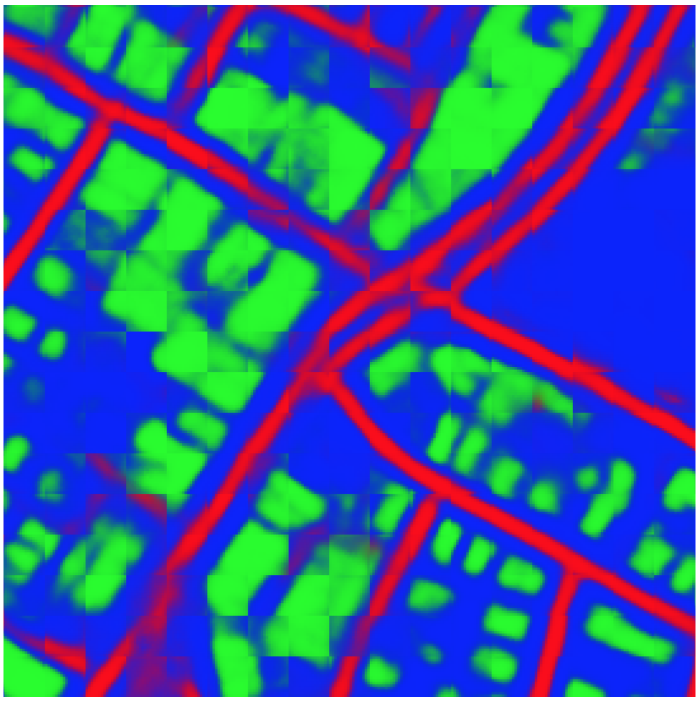


Figure 3-1: Example of a patchy result

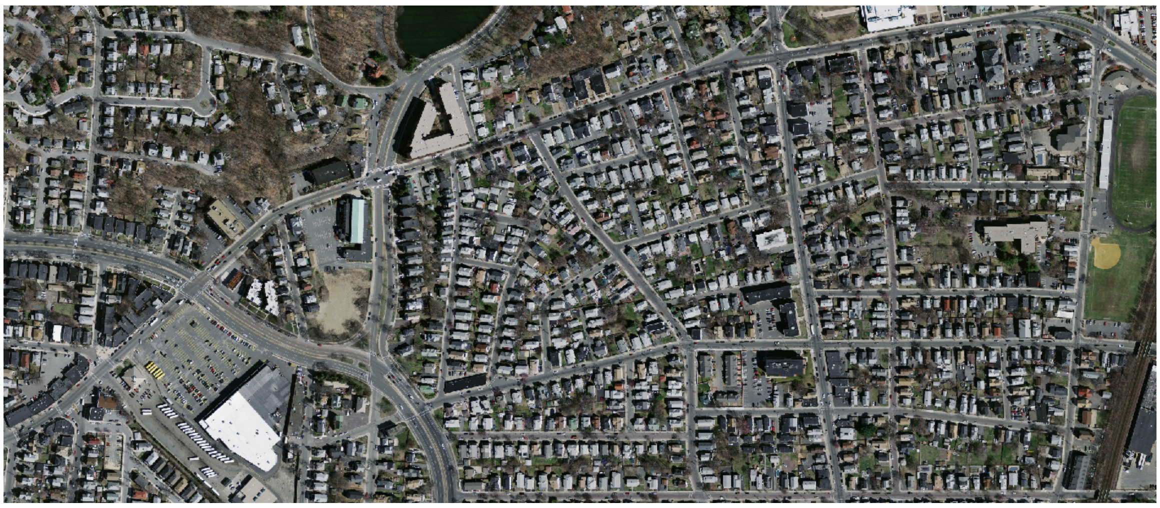
that we consider, namely, buildings and roads, are correlated with each other, so we assume that if we exploit the correlation effectively with a CNN, the simultaneous multi-channel prediction will be more accurate compared to the case of predicting each channel independently with different CNNs that have a single-channel output. Thus, we predict the probabilities of both buildings and roads at each pixel. To predict multiple labels, we have to consider the background class at the same time. However, there are potentially a large number of classes in the background (e.g., tree, river, sea, dog, human, car, etc.), so the feature to represent the background class would be difficult to obtain compared with the feature to represent a single class, namely, buildings or roads. Therefore, we assume that if we suppress the effect of the background class in the loss function for the training of a CNN, the CNN can learn about building and road representation more efficiently. To use such loss function in the training stage, we propose a new activation function for the final layer of a CNN. Utilizing this function during training improves the performance of the CNN that predicts multi-channel label images. Our CNN surpasses the state-of-the-art performance without using any further machinery such as noise models and structured prediction which are utilized in conventional methods [Mnih, 2013, Mnih and Hinton, 2010, 2012].

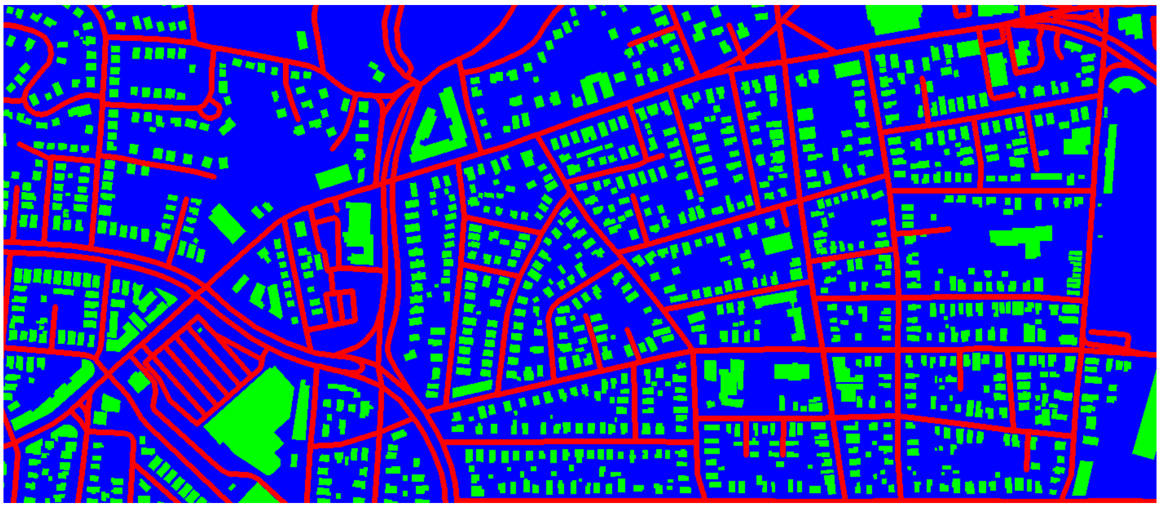
As discussed in Chapter 2, our goal is to predict a multi-channel label image $\tilde{\mathbf{M}}$ from an input aerial imagery $\mathbf{S}$. Figure 3-2 shows an example of $\mathbf{S}$ and $\tilde{\mathbf{M}}$. The resolution of these pictures is 1.0 m$^2$/pixel and the image size of these pictures is $1415 \times 610$. We directly learn a mapping from raw pixel values in $\mathbf{S}$ to a true label image $\tilde{\mathbf{M}}$ by training a CNN. Figure 3-2b shows an example true label image. This figure represents a binary road label map as the red channel of the picture, a binary building label map as the green channel, and the XOR of these label maps that denotes a binary background label map as the blue channel. Figure 3-2c shows an example prediction result from Figure 3-2a, and the colors in this figure have the same meanings as Figure 3-2b, but the pixel values are continuous differently from Figure 3-2b.

Let $K'$ be the number of object classes of interest; label image $\tilde{\mathbf{M}}$ has $K' + 1$ channels. Because it is difficult to consider all objects that can appear in aerial imagery as the classes of interest, we consider the background class to represent a pixel that belongs not to any of $K'$ classes of interest. In this thesis, we extract two objects, buildings and roads, so that a label image consists of three channels, buildings, roads, and the background. Let $K$ denote the number of all classes including the background, $K = K' + 1 = 3$. Here, a label image is a $K$-channel image, so that each single pixel on the label image is a $K$-dimensional vector. In a label image, the sum over all elements of a pixel vector is always one because each pixel should always be either buildings, roads, or the background.

## 3.1 The Network Architecture

We modify the CNN architecture that was used in [Mnih, 2013] as shown in Figure 2-1. We aim to predict multiple object labels at the same time, so that the final layer of the CNN should be a multi-channel image form. Therefore, we propose the modified version of the CNN as shown in Figure 3-3. This architecture is described


(a) Aerial imagery $\mathbf{S}$


(b) Ground truth $\tilde{\mathbf{M}}$

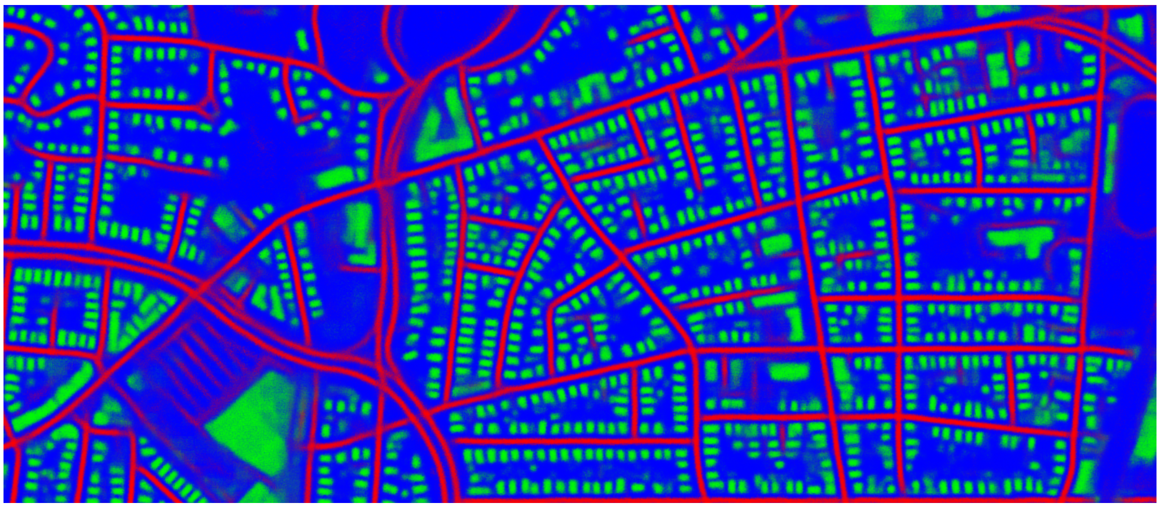

(c) Predicted label image $\hat{\mathbf{M}}$

Figure 3-2: An example of resulting predicted label image

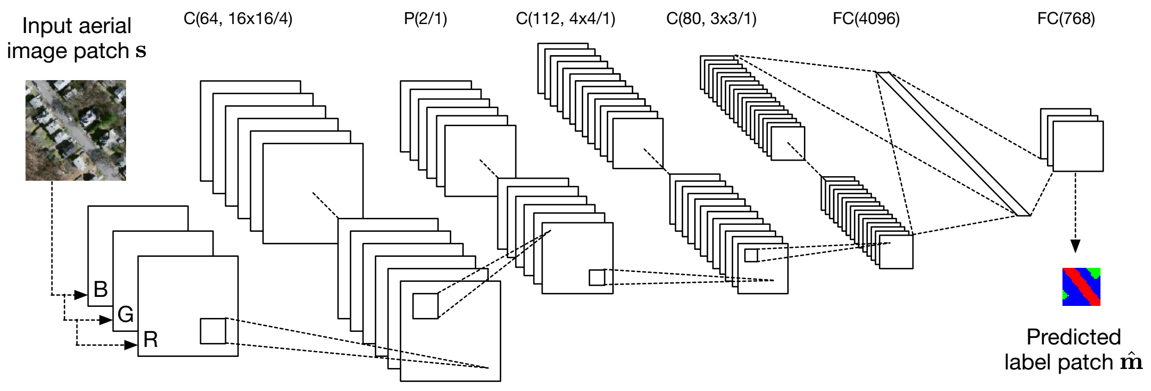

Input aerial image patch $\mathbf{s}$
C(64, 16x16/4)
P(2/1)
C(112, 4x4/1)
C(80, 3x3/1)
FC(4096)
FC(768)
B
G
R
Predicted label patch $\hat{\mathbf{m}}$

Figure 3-3: The architecture of our CNN used in this thesis

as $C(64, 16 \times 16/4)$ - $P(2/1)$ - $C(112, 4 \times 4/1)$ - $C(80, 3 \times 3/1)$ - $FC(4096)$ - $FC(768)$. Figure 3-3 shows the feature maps after each convolutional layer and pooling layer with white rectangles.

We tried another architecture that has 16 layers, VGG-16 [Simonyan and Zisserman, 2014] which has deeper architecture, and ensured that the model could perform better on the semantic segmentation task for aerial imagery with the same dataset. However, finding the optimal architecture beyonds the scope of this thesis. In this thesis, we focus on the exclusiveness of different classes and how to use it for improving the performance. Then, we propose a new activation function and show its effectiveness by comparing with the results reported in [Mnih and Hinton, 2012, Mnih, 2013] that use basically the same network architecture. Using the same architecture is important to compare the exact effect of the difference of the activation function.

## 3.2 Patch-based Formulation

In this thesis, we formulate the pixel labeling task in a similar way to what has been proposed by Mnih et al. [Mnih and Hinton, 2012, Mnih, 2013] We train the CNN to predict a $w_m \times w_m$-sized true label patch $\tilde{\mathbf{m}}$ from a $w_s \times w_s$-sized aerial imagery patch $\mathbf{s}$. Each pixel at location $i$ in a true label patch $\tilde{\mathbf{m}}$ is a $K$-dimensional one-hot vector, $\tilde{\mathbf{m}}_i = [\tilde{m}_{i1}, \tilde{m}_{i2}, \ldots, \tilde{m}_{iK}]$. We describe the output of the CNN by $\hat{\mathbf{m}}$ that is a $w_m \times w_m$-sized predicted label patch. Each pixel at $i$ in a predicted label patch $\hat{\mathbf{m}}$

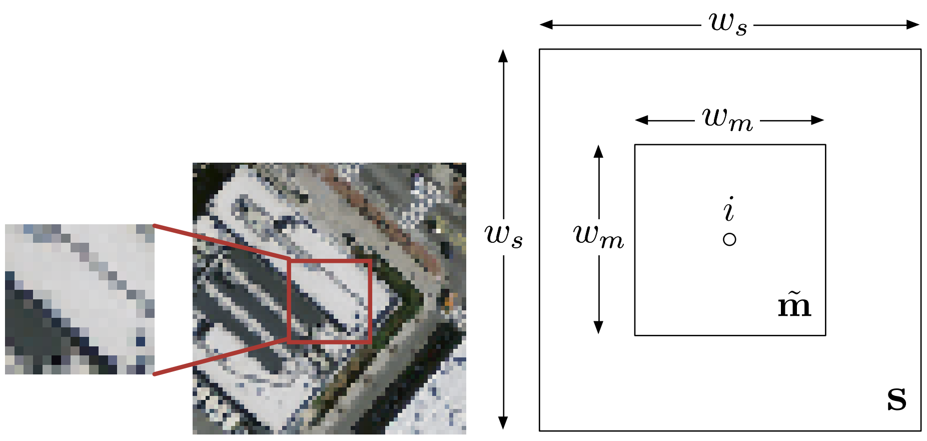


Figure 3-4: The size difference between label image patch $\tilde{\mathbf{m}}$ and aerial imagery patch $\mathbf{s}$

is also a $K$-dimensional vector. Here, we assume that all pixels in a true label patch $\tilde{\mathbf{m}}_i (i = 1, \ldots, w_m^2)$ are conditionally independent of each other given a corresponding aerial imagery patch $\mathbf{s}$. Therefore, the posterior of a true label patch given an aerial imagery patch is represented as

$$p(\tilde{\mathbf{m}}|\mathbf{s}) = \prod_{i=1}^{w_m^2} p(\tilde{\mathbf{m}}_i|\mathbf{s}). \tag{3.1}$$

We train the CNN to maximize this posterior by minimizing the negative log likelihood defined as

$$\mathcal{L} = -\sum_{i=1}^{w_m^2} \ln p(\tilde{\mathbf{m}}_i|\mathbf{s}). \tag{3.2}$$

Figure 3-4 shows the size difference between the input and output patches. For example, focussing on a small region as depicted in the leftmost patch in Figure 3-4, it is difficult to recognize what it is. However, if an input aerial imagery patch has wider region as depicted in the center image in Figure 3-4, some context information can be utilized to predict labels, and it can be recognized as a part of a building. Therefore, we set the size of an input patch $w_s$ larger than the size of a predicted label patch $w_m$, as depicted in the rightmost figure in Figure 3-4. This technique for context consideration is also utilized by Mnih et al. [Mnih and Hinton, 2010] and improves the performance.

## 3.3 Channel-wise Inhibited Softmax

In this thesis, $w_s = 64$, $w_m = 16$, and $K = 3$, so we reshape the last layer of the CNN (a 768-dimensional vector) to a $16 \times 16 \times 3$-sized image patch form. Let $\mathbf{x}_i = [x_{i1}, x_{i2}, x_{i3}]^{\mathrm{T}}$ denote the $i$th pixel in this output patch. Here, the softmax operation

$$\hat{m}_{ik} = \frac{\exp(x_{ik})}{\sum_j \exp(x_{ij})} \tag{3.3}$$

is applied to each element of $\mathbf{x}_i$ to convert it into a label probability vector $\hat{\mathbf{m}}_i = [\hat{m}_{i1}, \hat{m}_{i2}, \hat{m}_{i3}]^{\mathrm{T}}$. Then, each posterior at a pixel $i$ is represented as

$$p(\tilde{\mathbf{m}}_i|\mathbf{s}) = \prod_{k=1}^{K} \hat{m}_{ik}^{\tilde{m}_{ik}}, \tag{3.4}$$

where $\tilde{\mathbf{m}}_i = [\tilde{m}_{i1}, \tilde{m}_{i2}, \tilde{m}_{i3}]^{\mathrm{T}}$ is a one-hot vector to represent a true label vector at pixel $i$. A one-hot vector has only one element that has 1 as its value and all the other elements have 0 as their values. Then, the negative log likelihood is calculated as below, and we minimize this loss by stochastic gradient descent [Bottou, 1991] with backpropagation [Rumelhart et al., 1988]:

$$\mathcal{L} = -\sum_{i=1}^{w_m^2} \sum_{k=1}^{K} \tilde{m}_{ik} \ln \hat{m}_{ik}. \tag{3.5}$$

The above formulation treats all classes equally in the loss function. Then, to minimize the loss value, the network will obtain the representation of the target concepts, namely, buildings, roads, and the background. However, the background class might originally have many kinds of classes (e.g., river, cars, pedestrians, trees, sea, etc.) Therefore, the feature representation for the background class should be more complex than the case of buildings and roads. Then, we consider how to suppress obtaining the complex feature representation for the background to get the network to focus more on obtaining the feature representation for buildings and roads.

Here, we propose a new activation function for the final layer of a CNN called *channel-wise inhibited softmax (CIS)*. It is used instead of the normal softmax described in Eq. 3.3. Using CIS, we inhibit all units in a specific channel in the output layer of the CNN before calculating softmax. In concrete terms, we put *zero* in all the output units of the CNN in the background channel, namely, $\forall i, x_{i1}$, and then calculate softmax by Eq. (3.3).

Let $\boldsymbol{\pi}(\cdot)$ denote an element-wise operation of the normal softmax to a vector and $\odot$ denote an element-wise product. The predicted three-channel probability vector at a pixel $i$ that is calculated with CIS is

$$\begin{aligned} \hat{\mathbf{m}}_i^{\mathrm{CIS}} &\equiv \boldsymbol{\pi}(\mathbf{c} \odot \mathbf{x}_i) \quad (3.6) \\ \text{where} \quad \mathbf{c} &= [c_1, c_2, \ldots, c_K]^{\mathrm{T}}, \\ \hat{m}_{ik}^{\mathrm{CIS}} &= \frac{\exp(c_k x_{ik})}{\sum_j \exp(c_j x_{ij}),} \\ c_k &= \begin{cases} 0 & \text{if } k = 1 \\ 1 & \text{otherwise.} \end{cases} \end{aligned}$$

Then the loss function with CIS is defined as

$$\mathcal{L}^{\mathrm{CIS}} = -\sum_{i=1}^{w_m^2} \sum_{k=1}^{K} \tilde{m}_{ik} \ln \hat{m}_{ik}^{\mathrm{CIS}}. \quad (3.7)$$

The derivative of this loss function at $x_{ij}$ is

$$\begin{aligned} \frac{\partial \mathcal{L}^{\mathrm{CIS}}}{\partial x_{ij}} &= -\sum_{k=1}^{K} \tilde{m}_{ik} \frac{1}{\hat{m}_{ik}^{\mathrm{CIS}}} \frac{\partial \hat{m}_{ik}^{\mathrm{CIS}}}{\partial x_{ij}} \\ &= -\sum_{k=1}^{K} c_j \tilde{m}_{ik} (\delta_{jk} - \hat{m}_{ij}^{\mathrm{CIS}}) \\ &= c_j \left\{ \hat{m}_{ij}^{\mathrm{CIS}} \sum_{k=1}^{K} \tilde{m}_{ik} - \sum_{k=1}^{K} \tilde{m}_{ik} \delta_{jk} \right\} \\ &= c_j (\hat{m}_{ij}^{\mathrm{CIS}} - \tilde{m}_{ij}), \quad (3.8) \end{aligned}$$

where

$$\frac{\partial \hat{m}_{ik}^{\text{CIS}}}{\partial x_{ij}} = \frac{\partial}{\partial x_{ij}} \frac{\exp(c_k x_{ik})}{\sum_{l=1}^{K} \exp(c_l x_{il})} = \begin{cases} c_j \hat{m}_{ik}^{\text{CIS}}(1 - \hat{m}_{ij}^{\text{CIS}}) & \text{if } j = k \\ c_j \hat{m}_{ik}^{\text{CIS}}(0 - \hat{m}_{ij}^{\text{CIS}}) & \text{otherwise,} \end{cases}$$

$$\delta_{jk} = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{otherwise.} \end{cases}$$

It should be noted that

$$\sum_{k=1}^{K} \tilde{m}_{ik} = 1 \tag{3.9}$$

is applied to the last but one line in Eq. 3.8. We use the new loss function defined with the new activation function, CIS, described in Eq. 3.7 to train a CNN with the consideration of the specialty of the background class.

## 3.4 Model Averaging with Spatial Displacement

To improve the resulting predicted label image, we further perform model averaging techniques in a new manner. In general, model averaging is an ensemble learning method to avoid overfitting. In the training stage, multiple models are trained on the same dataset, and in the inference stage, the outputs from those trained models are averaged to obtain the final results. Recently in the context of neural networks, Dropout [Hinton et al., 2012] is known as a way to obtain the similar effect of model averaging within a single network. It is a technique that randomly drop units from the network during training. On the other hand, in our model, we explicitly perform model averaging by training multiple CNNs that have the same architecture but are initialized with different weights, while we perform dropout with dropout ratio = 0.5 to the $FC(4096)$ layer for regularizing the network. Therefore, we train a CNN which has the same architecture eight times. Each training stage is started from different initial weights that are sampled from the same Gaussian distribution with different random seeds.

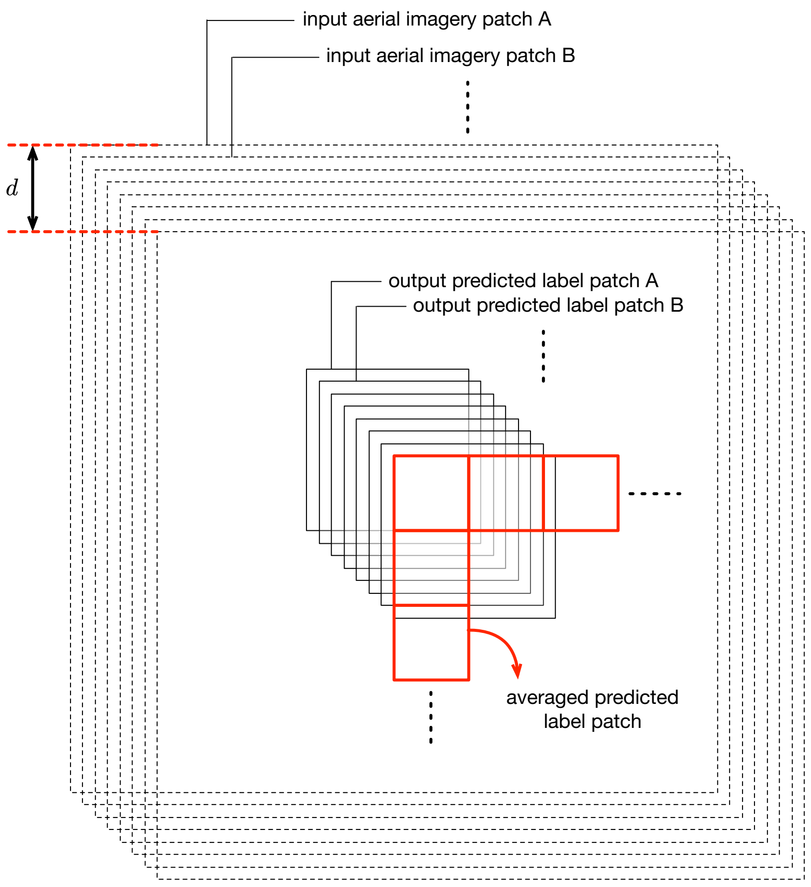

input aerial imagery patch A
input aerial imagery patch B
d
output predicted label patch A
output predicted label patch B
averaged predicted label patch

Figure 3-5: Model averaging with spatial displacement

In the inference stage, we give eight displaced input aerial images with different offsets to those eight versions of a single model. We call this operation *model averaging with spatial displacement (MA)*. Figure 3-5 shows that eight inputs for eight different versions of a single model are displaced by $d (0 \leq d \leq 7)$ pixels from the original image location. The predicted label patches of those versions are tiled with the same displacement $d$ to synthesize a final predicted label image. After tiling those patches, we divide all pixel values by eight for averaging.

## 3.5 Learning

We learn all parameters in the CNN by minimizing the loss function with mini-batch stochastic gradient descent with momentum. During learning, we reduce the learning rate by multiplying by a fixed reducing rate every $\tau$ epochs. Furthermore, we regularize the network with L2 weight decay. Therefore, hyper-parameters in the learning stage are the mini-batch size, the learning rate (LR) $\eta$, the LR reducing rate $\gamma$, the LR reducing frequency $\tau$, a weight of the momentum term $\alpha$, and a weight of the L2 weight decay $\beta$. The learning rate $\eta$ is started from $\eta_0$, and we use the following values for all experiments in this chapter: $\eta_0 = 0.0005$, $\tau = 100$, $\gamma = 0.1$, $\alpha = 0.9$, $\beta = 0.0005$, with the mini-batch size = 128. The training is continued until the epoch reaches 400. These values are empirically chosen based on the settings in the paper proposed by Mnih [Mnih, 2013].

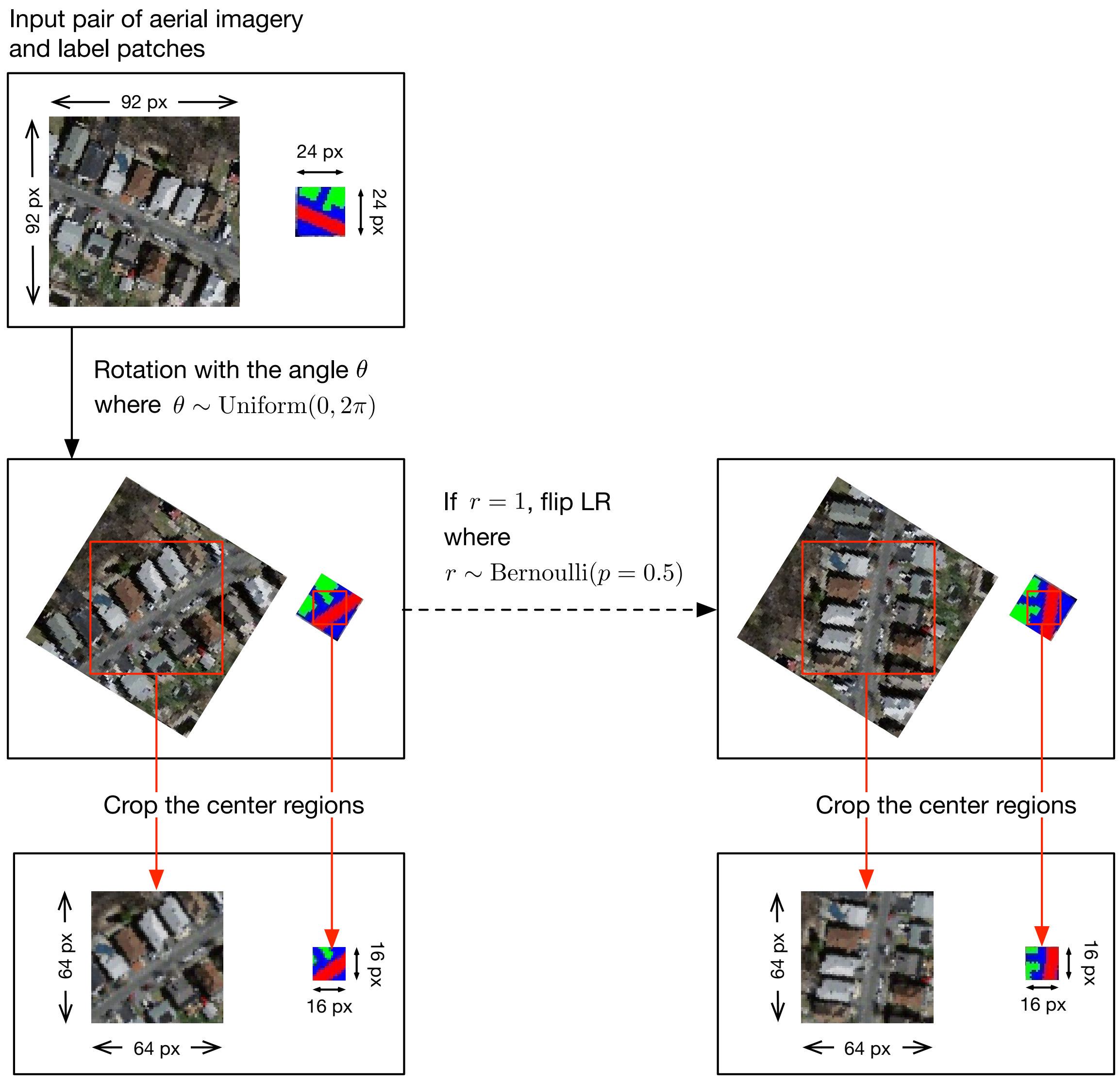

Input pair of aerial imagery and label patches
92 px
92 px
24 px
24 px
Rotation with the angle $\theta$ where $\theta \sim \mathrm{Uniform}(0, 2\pi)$
If $r = 1$, flip LR where $r \sim \mathrm{Bernoulli}(p = 0.5)$
Crop the center regions
Crop the center regions
64 px
64 px
16 px
16 px
64 px
64 px
16 px
16 px

Figure 3-6: The process flow for the data augmentation.

### 3.5.1 Data Augmentation

During training of the CNN, we perform real-time data augmentation to extend the number of patches in the dataset by performing two kinds of transformations to both the input aerial imagery patch and the corresponding true label patch. We apply rotation with a random angle $\theta$ and random L-R flip to them. The random angle $\theta$ is sampled from the uniform distribution, $\theta \sim \text{Uniform}(0, 2\pi)$, for every patch. Both the input aerial imagery patch and the true label patch are rotated around the center point with the same angle. L-R flipping is also performed randomly and equally to both inputs and true label patches.

To apply random rotation to a pair of aerial imagery and label patches, we firstly take a pair of a $92 \times 92$-sized aerial imagery patch and a $24 \times 24$-sized true label patch. Then we apply the random rotation to both patches with the same angle $\theta \sim \text{Uniform}(0, 2)$. The left and right flipping operation is performed only when $r \sim \text{Bernoulli}(p = 0.5)$ is 1. Finally, the center $64 \times 64$-sized and $16 \times 16$-sized regions of both patches are cropped. This process flow is summarized in Figure 3-6. We apply this process to all pairs in a mini-batch during training.

### 3.5.2 Normalization

After performing transformations to input and true label patches, the input aerial imagery patch is normalized by subtracting mean value and dividing by the standard deviation. This procedure is called global contrast normalization [Goodfellow et al., 2013]. It reduces the variation caused by the difference of light conditions while capturing aerial imagery. Let $s_{ik}$ denote the $i$th pixel value of the $k$th channel in the input aerial imagery patch $\mathbf{s}$, the normalized value $s'_{ik}$ is calculated as below:

$$s'_{ik} = (s_{ik} - \mu_k)/(\sigma_k + 10^{-5}), \tag{3.10}$$

$$\text{where} \quad \mu_k = \frac{1}{w_m^2} \sum_{i=0}^{w_m^2} s_{ik}, \tag{3.11}$$

$$\sigma_k = \left( \frac{1}{w_m^2} \sum_{i=0}^{w_m^2} (s_{ik} - \mu)^2 \right)^{\frac{1}{2}}. \tag{3.12}$$

In Eq. 3.10, we added $10^{-5}$ to the standard deviation $\sigma_k$ to avoid the numerical error caused by division by zero. It should be noted that the mean and standard deviation are calculated for each channel of the aerial imagery separately.

## 3.6 Dataset

To predict multiple labels simultaneously, we built a new dataset by selecting data from two datasets, *Massachusetts Buildings Dataset (Mass. Buildings)* and *Massachusetts Roads Dataset (Mass. Roads)*, which are released by Mnih [Mnih, 2013] and publicly available on the website [1]. These two datasets have many aerial images and corresponding binary label images that are aligned to those aerial images. The size of all images in these datasets is $1500 \times 1500$ and the resolution is $1\text{m}^2$/pixel. *Mass. Buildings* consists of 151 pairs of aerial images and corresponding building label binary images, and covers a large area of roughly 340 $\text{km}^2$. This dataset is divided into three groups. The training, validation, and test subsets of this dataset comprise 137, 4, and 10 images, respectively. On the other hand, *Mass. Roads* consists of 1171 pairs of aerial images and corresponding road label binary images, and these are separated into 1108 training images, 14 validation images, and 49 test images. In general, a training dataset is used for learning of parameters of a model, and a validation dataset is used to determine some hyper parameters such as the number of layers or units and when to stop the training, and a test dataset is used to evalu-

[1] http://www.cs.toronto.edu/˜vmnih/data/

Table 3.1: The overview of datasets

| Dataset name | Training | Validation | Test |
|---|---|---|---|
| Mass. Roads [Mnih, 2013] | 1108 | 14 | 49 |
| Mass. Roads-Mini | 137 | 4 | 10 |
| Mass. Buildings [Mnih, 2013] | 137 | 4 | 10 |
| Mass. BR | 137 | 4 | 10 |

ate the performance of the trained model and compare it with other models. Then, we merged *Mass. Buildings* and *Mass. Roads* to create *Massachusetts Buildings and Roads dataset (Mass. BR)* which has multi-channel label images. Table 3.1 shows the composition of all datasets we use in the experiment described in the next chapter.

To create *Mass. BR* dataset, we selected aerial images that had both building labels and road labels from *Mass. Buildings* and *Mass. Roads*. Then we found that the all aerial images in *Mass. Buildings* are included in *Mass. Roads*. Therefore, we added road labels to the *Mass. Buildings* dataset. We synthesized three-channel label images by stacking building, road, and background label images as the three

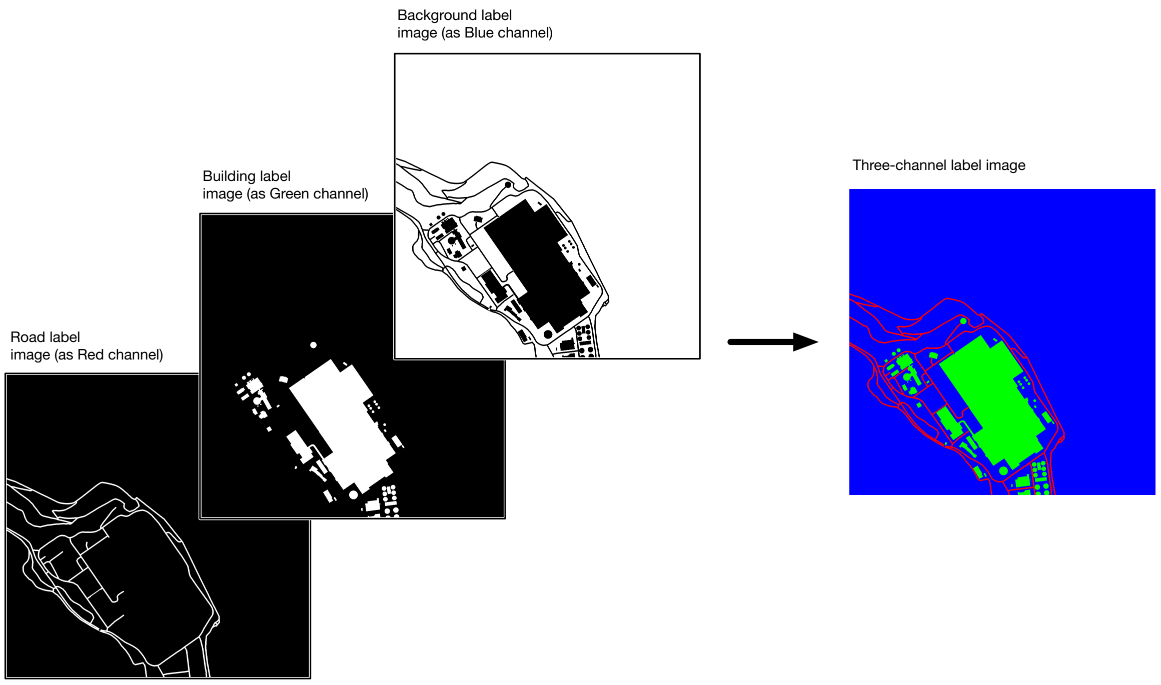


Figure 3-7: How to construct a three-channel label image

channels as shown in Figure 3-7. Background label images were created by calculating the XOR of building and road label images. Then we found some obviously incorrect labels in road channels of some test images that are chosen from training images in *Mass. Roads*, so that we correct the labels manually for more precise evaluation. Examples of an aerial image and the corresponding three-channel label image in this new dataset are shown in Figure 3-8.

Finally, we found that we cannot compare the result of a model trained on *Mass. BR* with another model trained on *Mass. Roads* directly, because some images in the training set of *Mass. BR* are included in the test set of *Mass. Roads*. Therefore, we took subsets of the *Mass. Roads* dataset and created the *Mass. Roads-Mini* dataset. The training, validation, and test images in *Mass. Roads-Mini* are completely same as in *Mass. BR*, but *Mass. Roads-Mini* has only road labels. Hence, using *Mass. Roads-Mini* dataset to train a single-channel CNN, we can directly compare the road prediction result with the road *channel* result of a multi-channel CNN.

Figure 3-8 shows six aerial images in the left top block. These images are chosen from 1171 aerial images in the Mass. Roads dataset, so that there are six corresponding road label images as shown in the center top block. However, the aerial images shown in the left column in the left top block are only in Mass. Roads, so that there are no corresponding label images in Mass. Roads-Mini, Mass. Buildings, and Mass. BR. It shows that the aerial images which are included in Mass. Roads-Mini, Mass. Buildings, and Mass. BR are the same and a subset of the images in Mass. Roads. Then, as shown in this figure, all aerial images and label images are $1500 \times 1500$-sized images. Label images in Mass. Roads, Mass. Roads-Mini, and Mass. Buildings are binary images, so that those images have only a single-channel. On the other hand, the label images in Mass. BR are three-channel color images, and each channel corresponds to road, building, or background label images as shown in Figure 3-7.

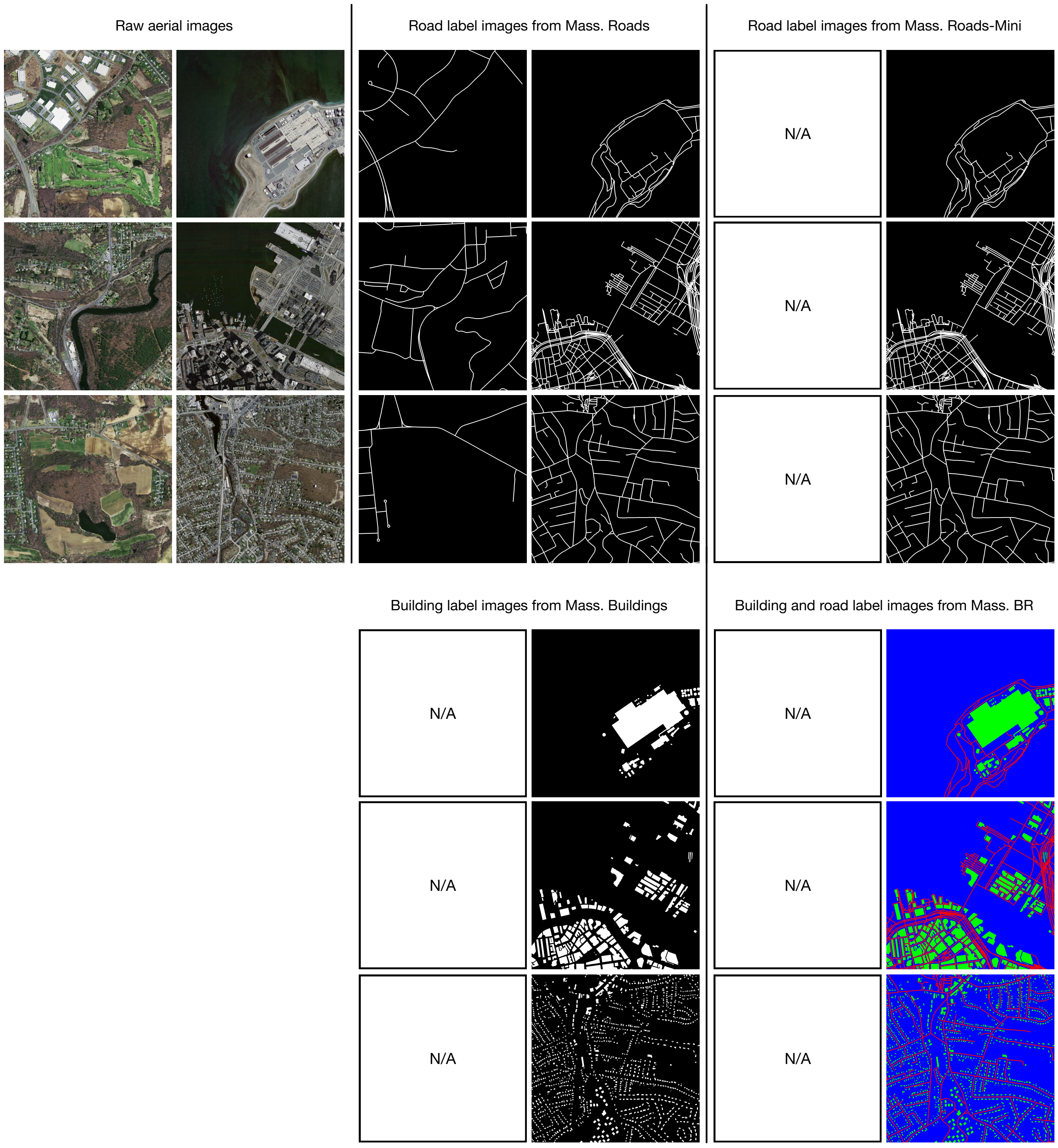

Raw aerial images
Road label images from Mass. Roads
Road label images from Mass. Roads-Mini
N/A
N/A
N/A
Building label images from Mass. Buildings
Building and road label images from Mass. BR
N/A
N/A
N/A
N/A
N/A
N/A

Figure 3-8: Examples of aerial and label images from each dataset

# Chapter 4

# Empirical Evaluation

In this chapter, we aim to show the effectiveness of our proposed methods, channel-wise inhibited softmax (CIS) and model averaging with spatial displacement (MA), by comparing the models with these methods with the previous models that were proposed by Mnih [Mnih, 2013]. First, we evaluate a single-channel CNN that has $FC(256)$ instead of $FC(768)$ in the final layer, which is trained on *Mass. Buildings* and *Mass. Roads* datasets separately with data augmentation and model averaging with spatial displacement (MA). We describe this single-channel model *Single-channel with MA*. On the other hand, our proposed multi-channel models are described as *Multi-channel with/without CIS and with/without MA* in all tables and figures. In this chapter, we perform two different experiments to evaluate those conventional models and our proposed models.

## 4.1 Evaluation Metric

First, the evaluation metric is important to show the practical effectiveness of our methods. The most common metrics for evaluating building and road extraction results are precision and recall. In the remote sensing literature, these are also called *correctness and completeness* [Wiedemann et al., 1998]. The precision is the ratio of the number of true building or road pixels in label images to the number of pixels detected as belonging to building or road in the predicted label images, while the recall is the ratio of the detected pixels to the true pixels. In a binary classification problem, the precision and recall are defined as

$$\text{precision} = \frac{\text{TP}}{\text{FP} + \text{TP}} \tag{4.1}$$

$$\text{recall} = \frac{\text{TP}}{\text{FN} + \text{TP}}, \tag{4.2}$$

where TP denotes the number of true positive pixels that means the number of correctly positive predictions [1] , FP denotes the number of false positive pixels that means the number of wrongly positive predictions, and FN denotes the number of correctly negative predictions. In this thesis, this metric is called the exact precision and recall. Precision and recall are calculated for each channel of predicted label patch. Let $\hat{m}_{ik}$ and $\tilde{m}_{ik}$ denote the predicted label value and true label value at the $i$th pixel in the $k$-th channel, and $\delta(C)$ denote the Dirac's delta function that takes 1 only when the given condition $C$ is true. TP, FP, and FN are calculated as below:

$$\text{TP} = \sum_{i=0}^{w_m^2} \delta(\hat{m}_{ik} \geq t \wedge \tilde{m}_{ik} = 1) \tag{4.3}$$

$$\text{FP} = \sum_{i=0}^{w_m^2} \delta(\hat{m}_{ik} \geq t \wedge \tilde{m}_{ik} = 0) \tag{4.4}$$

$$\text{FN} = \sum_{i=0}^{w_m^2} \delta(\hat{m}_{ik} < t \wedge \tilde{m}_{ik} = 1), \tag{4.5}$$

where $t \in [0, 1]$ is the threshold to binarize the continuous predicted label value $\hat{m}_{ik}$.

However, to compare our results with the conventional methods' results reported by Mnih [Mnih, 2013], we should use the same metric to evaluate our results. They used *relaxed precision and recall scores* instead of exact precision and recall scores for all experiments. The relaxed precision is defined as the fraction of detected pixels that are within $\rho$ pixels of a true pixel, while the relaxed recall is defined as the

[1] In a binary classification task that aims at giving 0 (negative) or 1 (positive) to an input instance, positive prediction means that a prediction that has 1 as its value, which denotes the input instance belongs to the target class.

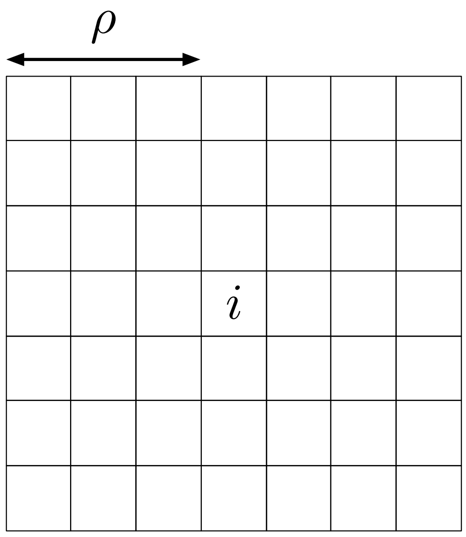


Figure 4-1: The region that $r(i, \rho = 3)$ indicates.

fraction of true pixels that are within $\rho$ pixels of a detected pixel. In other words, the definitions of TP, FP, and FN are modified as below:

$$\mathrm{TP}_{\text{relaxed}} = \sum_{i=0}^{w_m^2} \delta\left(\hat{m}_{ik} \geq t \wedge \sum_{i \in r(i,\rho)} \tilde{m}_{ik} > 0\right) \tag{4.6}$$

$$\mathrm{FP}_{\text{relaxed}} = \sum_{i=0}^{w_m^2} \delta\left(\hat{m}_{ik} \geq t \wedge \sum_{i \in r(i,\rho)} \tilde{m}_{ik} = 0\right) \tag{4.7}$$

$$\mathrm{FN}_{\text{relaxed}} = \sum_{i=0}^{w_m^2} \delta\left(\hat{m}_{ik} < t \wedge \sum_{i \in r(i,\rho)} \tilde{m}_{ik} > 0\right), \tag{4.8}$$

where $r(i, \rho)$ denotes the set of all pixel locations of the $(\rho + 1) \times (\rho + 1)$-sized region centered at $i$. An example region that $r(i, \rho = 3)$ indicates is shown in Figure 4-1. Relaxation the precision and recall in this manner is also used in Wiedemann et al. [Wiedemann et al., 1998] Then, in all experiments in this chapter, the slack parameter $\rho$ is set to 3, which is the same value as used in Wiedemann et al. [Wiedemann et al., 1998] and Mnih [Mnih, 2013].

By changing the threshold $t$ in Eq. 4.6, 4.7, and 4.8, a precision and recall curve which consists of many sets of precision and recall values at different thresholds $t$ can be drawn as shown in Figure 4-4. Then, we summarize this curve with a recall at breakeven point. At the breakeven point, the precision and recall values are equal. In other words, the breakeven point is the cross point of the line $y = x$ in the precision-recall curve figure and the precision-recall curve itself. All values in Table 4.1, 4.2, and 4.3 are recalls at breakeven points.

The reason why the breakeven point is used for summarizing the evaluation results is based on the objective of object extraction from aerial imagery. The precision is related with over detection, while the recall is related with miss detection. We want to extract all objects correctly, but we also want to suppress the wrongly detected pixels as target objects. Because, wrong results should be corrected manually regardless of whether that is false negative or false positive. Therefore, the balanced point between precision and recall should be considered as the best point that yields the optimal threshold for the results.

Table 4.1: Recall at breakeven on test dataset of single-channel prediction models

| Model | Mass. Buildings | Mass. Roads | Mass. Roads-Mini |
|---|---|---|---|
| Mnih-CNN [Mnih, 2013] | 0.9150 | 0.8873 | N/A |
| Mnih-CNN + CRF [Mnih, 2013] | 0.9211 | 0.8904 | N/A |
| Mnih-CNN + Post-processing net [Mnih, 2013] | 0.9203 | 0.9006 | N/A |
| Sinngle-channel with MA | **0.9426** | **0.9047** | **0.9005** |

Table 4.2: Recall at breakeven on test dataset of multi-channel prediction models.

| Model | Building channel | Road channel |
|---|---|---|
| Multi-channel with MA | 0.9523 | 0.9098 |
| Multi-channel with CIS | 0.9442 | 0.9039 |
| Multi-channel with CIS + MA | **0.9528** | **0.9111** |

## 4.2 Performance Evaluation

The first experiment is performed on all datasets. We train a conventional model (Single-channel) with the training set of *Mass. Buildings*, *Mass. Roads*, and *Mass.Roads-Mini* with data augmentation, and then create the predictions with the test set by performing MA. After that, we calculate the precision and recall curves with the test set of all the above datasets.

Table 4.1 shows the performances of the single-channel predictions of the conventional method [Mnih and Hinton, 2012] in the top three rows, and the performance of the same single-channel prediction with MA in the last row. Each model is trained and tested on the dataset indicated in the first row. Table 4.2 shows each channel's result of our proposed multi-channel models. It should be noted that all of our models are trained with data augmentation, though we do not describe this explicitly in all tables. *Multi-channel with CIS* means that the model is trained on *Mass. BR* and the loss function is calculated by our CIS version of cross entropy loss (Eq. 3.7). The results of the road channel column described in Table 4.2 can be compared to the result of models trained and tested on *Mass. Roads-Mini* directly, because *Mass. Roads-Mini* is created to have the same training/test images as *Mass. BR* dataset. All values in these tables are recalls at breakeven points of the *relaxed precision-recall curves* depicted in Figure 4-4.

For single-channel prediction, Table 4.1 shows that *Single-channel with MA* is the best result on all datasets. All the models in this table are to predict a single-channel label image that represents either building or road probability. It should be noted that the conventional methods shown in the top three rows perform noise modeling. Furthermore, *Mnih-CNN + CRF* and *Mnih-CNN + Post-processing net* use a conditional random field and an additional multi-layer perceptron for structured prediction, respectively. On the other hand, our model does not model label noise explicitly and has no post-processing. However, *Single-channel with MA* is more accurate than all the conventional models on all datasets. The differences between *Mnih-CNN* and *Single-channel with MA* are the use or not of data augmentation during training and

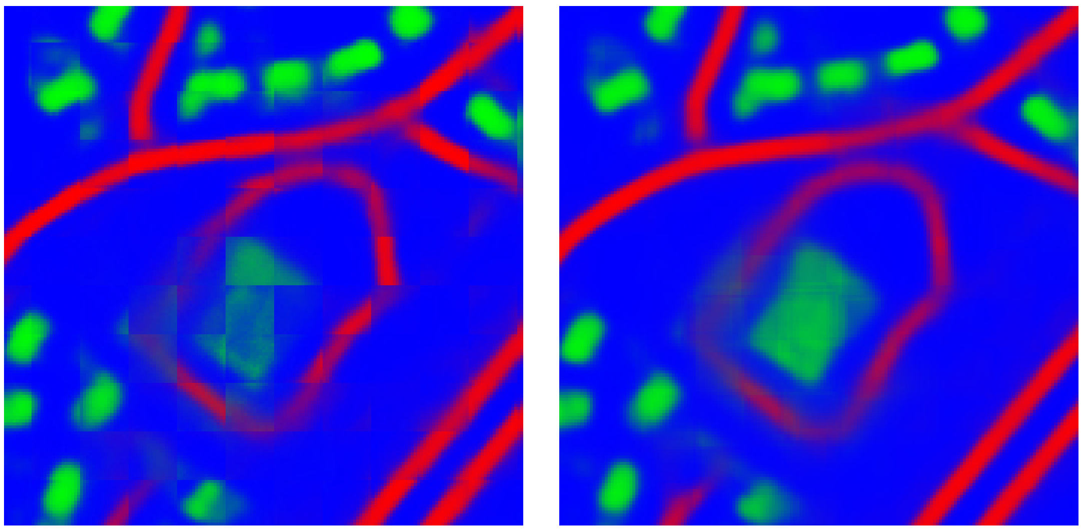

(a) (b)

Figure 4-2: A prediction result of *Multi-channel*: (a) *without* model averaging with spatial displacement. (b) *with* model averaging with spatial displacement.

MA in the inference stage. Although MA does not perform refinement of results by recognizing any meanings of pixels, MA smoothes the outputs over the boundaries of predicted label patches effectively and improves the performance significantly. This effectiveness is also shown in the case of multi-channel prediction. Figure 4-2 shows the effectiveness of MA in a concrete example.

Our multi-channel CNNs output three-channel predicted label patches. Therefore, we extracted each channel to evaluate the accuracy of each class. In Table 4.2, we compared the performance of our proposed techniques on the same base architecture depicted in Figure 3-3. In this table, *Multi-channel with CIS + MA* achieves the best results both in building and road channels even compared with the results of single-channel predictions. The number of training images in *Mass. BR* is one-eighth compared with *Mass. Roads*, but the performance of this model trained on *Mass. BR* is better than all of the results of single-channel models trained on *Mass. Roads*, though the test images in these two datasets are different from each other. Adding CIS always improves the performance. Therefore, we found that CIS which assigns *zero* to all units in the background channel and performs softmax is a better way to train a CNN for multi-channel semantic segmentation of aerial imagery, and performing

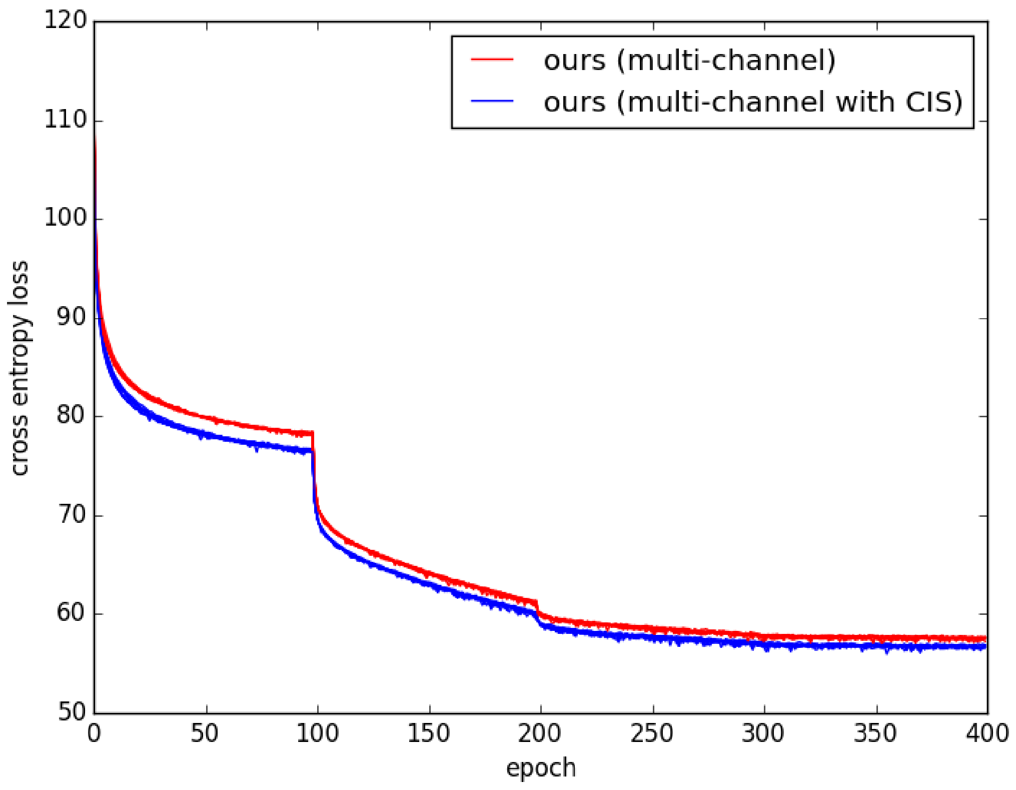


Figure 4-3: Comparison of cross entropy loss between two models during training.

MA in inference stage is also effective.

Figure 4-3 shows a comparison of cross entropy loss between our models with or without CIS during training. The red curves show the loss values of *Multi-channel* (without CIS), and the blue curves show the loss values of *Multi-channel with CIS*. We trained each model eight times, so all curves of the eight versions of each model are drawn in this figure. We reduced the learning rate every 100 epochs, so there are some cliffs on the curves. When the learning rate is reduced by multiplying by 0.1, the loss get decreased with larger margin especially after 100 epoch. It should be noted that the loss curves of the models with CIS are always under the curves of the models without CIS. This observation simply shows that CIS can effectively work for training the CNNs.

Table 4.3: Recall at breakeven on the selected region of test images.

| Model | Mass. Buildings or Buildings channel | Mass. Roads-Mini or Road channel |
|---|---|---|
| Single-channel with MA | 0.9601 | 0.9197 |
| Multi-channel with MA | 0.9617 | 0.9236 |
| Multi-channel with CIS + MA | **0.9642** | **0.9276** |

## 4.3 Evaluation on Urban Area

To show the benefit of using the correlation between buildings and roads in aerial images, we perform further evaluation. We assume that if our model utilizes the correlation effectively, the performance should be better than the conventional model, with a larger margin in urban regions in which both buildings and roads appear. Because if a patch has only road or building pixels, the correlation cannot be utilized. Therefore, we extract $16 \times 16$-sized patches that have both $N_b$ building pixels and $N_r$ road pixels, where

$$N_b > \frac{w_s^2}{K}, \tag{4.9}$$

$$N_r > \frac{w_s^2}{K}, \tag{4.10}$$

from the test label images. Here, $w_m^2$ and $K$ are the number of pixels in an aerial image patch ($16 \times 16 = 256$) and the number of target classes ($K = 3$), respectively. Thus, the extracted patches always include both buildings and roads with a larger area than a third of the whole area of the patch. We evaluate three models on the extracted test patches.

Table 4.3 shows the resulting recalls at breakeven points that are evaluated on urban areas of test images. The row of *Single-channel with MA* shows the results of the single-channel models trained on *Mass. Buildings* or *Mass. Roads-Mini* independently, and MA is used to create final outputs. In Table 4.3, all the recall values of *Multi-channel with CIS + MA* are better than for the single-channel model in the same table and better than for all recall values shown in Table 4.2. These improvements show that our models have more advantage in urban regions that have both buildings and roads appearing at the same time within a small region because of utilization of the correlation. This advantage is important in some applications that require high accuracy in crowded urban areas, for example, real estate management and updating maps in areas that have many small houses and roads.

Focusing on the results in road channel shown in Table 4.3, all scores are increased from the results shown in Table 4.2. The percentages of the increases of *Multi-channel with MA* and *Multi-channel with CIS + MA* are 1.52 %, and 1.81 %, respectively. The latter model shows the bigger performance increase compared with the former model. In the evaluation results on whole test images shown in Table 4.2, the relative performance increase from *Multi-channel with MA* to *Multi-channel with CIS + MA* is $0.9111 - 0.9098 = 0.0013$ %, while in those results on extracted regions, the relative increase is $0.9276 - 0.9236 = 0.004$ %, so that there is a over three times larger improvement. This result also supports the effectivity of CIS.

## 4.4 Discussion

### 4.4.1 The Related Techniques to CIS

The reason why CIS improves the performance might be related to the inherent redundancy of the softmax function. In general, subtracting a constant vector from an input vector given to softmax does not affect the output of softmax. For example, let $\mathbf{x}$ be a $K$ dimensional vector and $\pi$ denote the softmax function. The $i$th element of the softmax result of $\mathbf{x} - \mathbf{c}$, where $\mathbf{c} = (c, c, \ldots, c)^{\mathrm{T}}$ (a $K$ dimensional constant vector), is

$$\pi(x_i - c) = \frac{\exp(x_i - c)}{\sum_{k=1}^{K} \exp(x_k - c)} \quad (4.11)$$

$$= \frac{\exp(-c)\exp(x_i)}{\exp(-c)\sum_{k=1}^{K} \exp(x_k)} \quad (4.12)$$

$$= \pi(x_i). \quad (4.13)$$

One of the typical strategies is to take a certain class $k$ to set $c = x_k$, so that $k$th element of the input vector $\mathbf{x}$ will be zero. On the other hand, it should be noted that we set zeros to the units in the background channel of the output layer, so that it is different from subtracting a constant value from all units. The CIS does not change all the units in the channels of the other classes other than the background channel. Then, there have also been some methods to eliminate this redundancy. One of them is to take $x_k \equiv 0$ for some preferred class $k$, which was introduced by Ripley [Ripley, 1996], and used in Andersen [Andersen et al., 1997] as a *normalized exponential transformation*. Hence, our CIS is same as performing this transformation to each pixel vector in the output layer of the CNN. The CIS might have the similar effect of this transformation for eliminating the inherent redundancy. Then, in a semantic segmentation task using a convolutional neural network, we finally showed that choosing the background channel and giving zeros to all units in the channel improve the performance in the task. It could be useful also in another task that cannot avoid considering a *background* class as one of the objective classes.

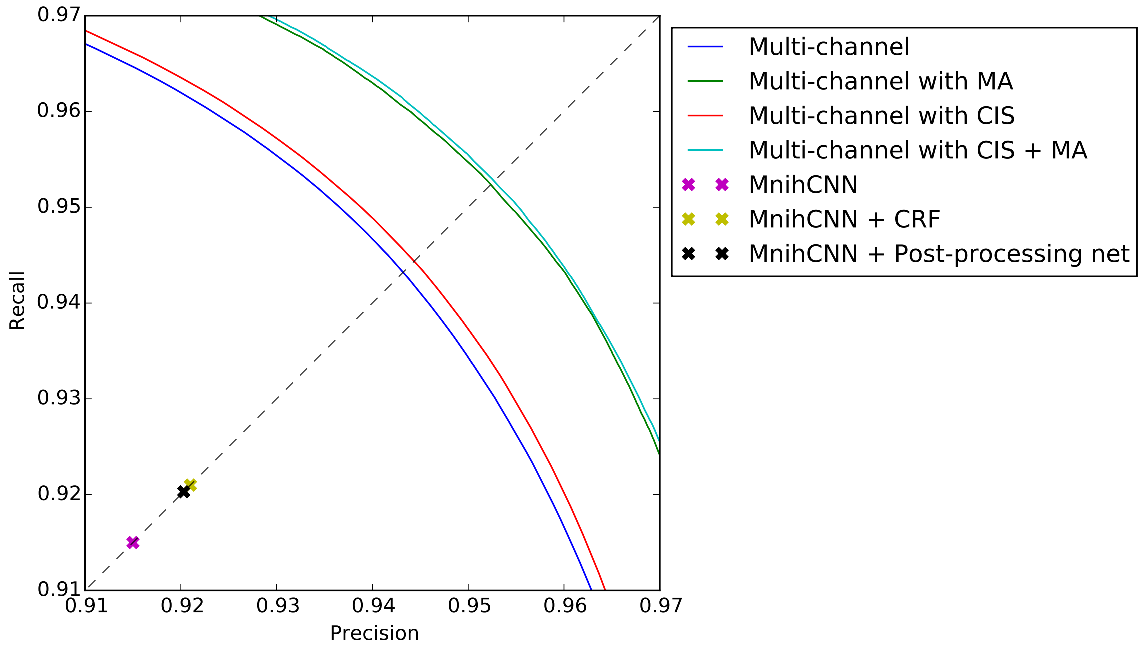
In the image classification task, Zhang and LeCun [Zhang and LeCun, 2015] proposed some regularization techniques to use unlabeled data called *universum prescription*. The fundamental idea of the proposed techniques is to use the unlabeled data for regularizing the network during training. They proposed some types of modified loss functions to be optimized during training. One of them called *background class* adds an additional term to the softmax function to consider the correct class of all unlabeled data as *background* and utilize them during training. The formulation of the loss function considering *background class* is completely same as our CIS. Therefore, in that paper, our CIS is introduced as *background class model with background constant* $\tau = 0$. Then, they showed the effectiveness of the new loss function on the image classification task with some widely used datasets, CIFAR-10 and CIFAR-100 [Krizhevsky and Hinton, 2009] and STL-10 [Coates et al., 2011]. Furthermore, they gave a qualitative justification of the regularizing effect of this technique using Rademacher complexity. Therefore, the regularizing ability of CIS for training a convolutional neural network might be also qualitatively justified.

### 4.4.2 Precision-Recall Curves

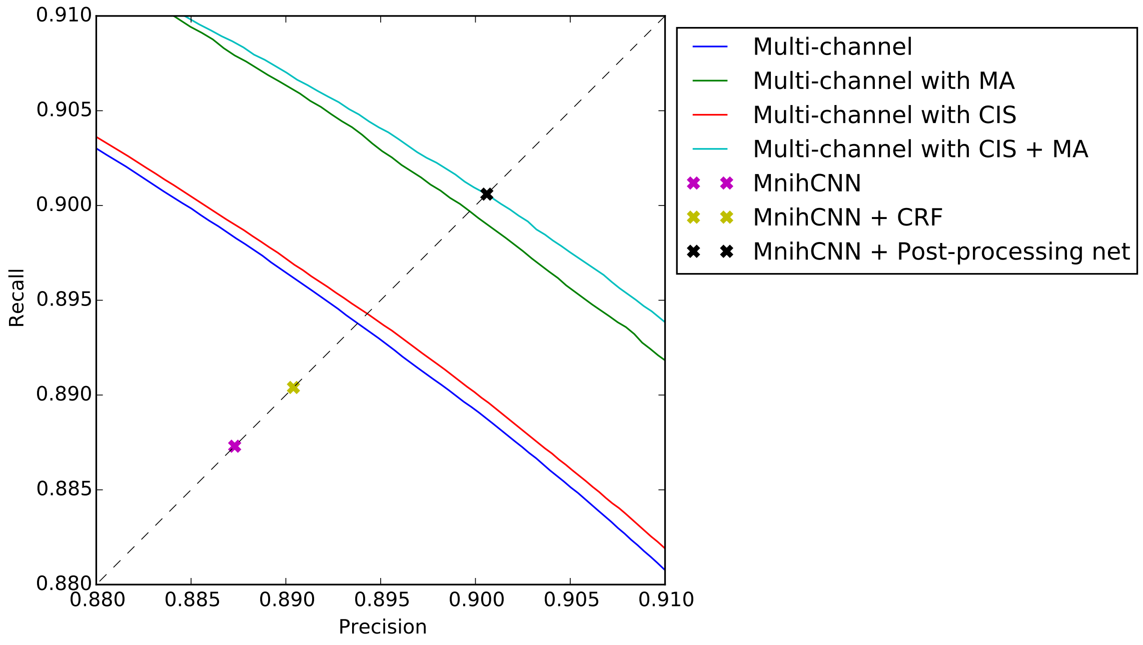
Figure 4-4 shows the *relaxed precision-recall curves* of all results. The red, green, and blue cross marks mean the recalls at breakeven points of the conventional models proposed in Mnih [Mnih, 2013]. All curves of *Multi-channel with CIS + MA* are located above those cross marks in both building and road prediction. As shown in both graphs, the results with *CIS* are always better than the results without *CIS*, and the results with *CIS + MA* are further better than the results with only *CIS*. This shows that *CIS* is effective for multi-channel model and *MA* can further improve the performance when it is used in the inference stage.

### 4.4.3 Computational Cost

The additional computational cost of CIS has little influence on the training time and the processing time to output predicted label patches. In the training stage, the average time to train one epoch was 9.248 minutes for CIS model and 9.441 minutes for the model without CIS. The whole training time for 400 epochs was 61.66 hours for the model with CIS and 63.00 hours for the model without CIS. The training stage is implemented with Chainer [Tokui et al., 2015], a neural network framework based on Python. In the inference stage, all of the processing times to predict a label image from a $1500 \times 1500$-sized aerial image were almost the same between all models when calculated on a NVIDIA Tesla K80 with Caffe [Jia et al., 2014],



(a) Precision-recall curve of building label prediction.



(b) Precision-recall curve of road label prediction.

Figure 4-4: Precision-recall curves

Table 4.4: The number of parameters in each layer of our CNN.

| | conv1 | conv2 | conv3 | fc4 | fc5 | total |
|---|---|---|---|---|---|---|
| Weight | 49512 | 114688 | 80640 | 16056320 | 3145728 | 19446528 |
| Bias | 64 | 112 | 80 | 4096 | 768 | 5120 |
| Total | 49216 | 114800 | 80720 | 16060416 | 3146496 | 19451648 |
| Percentage [%] | 0.2530 | 0.5902 | 0.4150 | 82.57 | 16.18 | 100 |

a fast neural network framework mainly implemented using C++ . The average processing times of *Multi-channel with CIS* and *Mnih-CNN* were 2.76 seconds and 2.80 seconds, respectively, for each aerial image. However, although the conventional method [Mnih, 2013] needs two different models to create a multi-channel result, our models can predict building and road labels simultaneously with a single feed-forward. This means that the conventional method needs about twice the processing time to obtain building and road predictions compared with ours. However, if we perform MA by using eight different versions of the model to obtain more accurate results, it requires about eight times longer processing time compared with the case of the conventional method. However, if multiple GPUs are available, all the calculations of eight versions can be parallelized, so the drawback turns out to be marginal.

### 4.4.4 The Number of Parameters

The total number of parameters in our model is about 20 millions. The detailed composition of the number of parameters in each layer of our model is shown in Table 4.4. In this table, all layers appear in the base architecture which is depicted in Figure 3-3 are represented as follows: $C(64, 16 \times 16/4)$ is described as conv1, and $C(112, 4 \times 4/1)$ is described as conv2, and $C(80, 3 \times 3/1)$ is described as conv3, and $FC(4096)$ is described as fc4, and $FC(768)$ is described as fc5. On the other hand, one can think this model has too many parameters, so the model can remember all combinations of input and output in the weight of the network. However, *Mass. BR* dataset has 137 aerial images, and each image has 3 channels, and all of the image sizes are equally $1500 \times 1500$, so the total number of values in the training dataset is 925 millions. The label images also have the same size and the same number of channels, so the total number of label values is the same. Therefore, the total number of parameters of our model is 1.05 % of the number of values in the dataset. This large gap means that the model performs some good feature extraction to the input images, and obtains good mappings to the output labels. Focusing on the percentage of parameters in the fully-connected layers, fc4 layer has over 80% of the whole parameters in the single layer.

There is a recent progress on how the fully-connected layers can be removed without performance decrease. Long et al. [Long et al., 2015] proposed a fully-convolutional networks for semantic segmentation. This model replaces the conventional fully-connected layers with $1 \times 1$-sized convolutional layers. This can reduce the total number of trainable parameters significantly. For example, if a fully-connected layer $FC(4096)$ in our model, which has 4096 units, is replaced with a $1 \times 1$ convolutional layer which has 4096 filters, the number of parameters in this layer is reduced to about 2% (16056320 to 327680). Because, there are 80 output feature maps in the next lower layer (see the base architecture depicted in Figure 3-3), and the shape of each map is $7 \times 7$, so that there are $80 \times 7 \times 7 = 3920$ units in the layer. Then, the number of connections between the layer and the next upper fully-connected layer $FC(4096)$ except the bias is $3920 \times 4096 = 16056320$ as shown in Table 4.4. On the other hand, a convolutional layer has tight weights that share the trainable parameters which connect to a channel in the lower feature map, so the number of parameters in the convolutional layer is $80 \times 4096 = 327680$. As shown in the paper [Long et al., 2015], although the fully-convolutional networks only has convolutional layers as its components, this can perform well on the semantic segmentation task. This type of architecture uses the strong spatial contiguity of inputs in semantic segmentation task, and this doesn't need to perform spatial smoothing such as *model averaging with spatial displacement*. Therefore, this can improve not only the accuracy but also the space and time complexity of computational cost. In this thesis, we aims at showing the effectiveness of our new activation function that can treat the *background* class effectively, but another modification on the architecture of the networks should be considered for future work.

We also tested some other resolutions of aerial imagery. For example, the aerial imagery taken in Japan, which have the resolution of 0.5 $\mathrm{m}^2$/pixel, was also used for training and evaluation. Then, we found that the completely same architecture even in terms of all the sizes of filters can be successfully trained on a different dataset which has different resolution, although the input and output patch sizes should be changed. Because, the model in the original setting sees a $64 \times 64$ $\mathrm{m}^2$ area as input, and this is the important perspective for successful training of a CNN for this task. Therefore, in the case of 0.5 $\mathrm{m}^2$/pixel, $128 \times 128$-sized aerial image patch should be input, and $32 \times 32$-sized label images that indicate the center region of the input patch should be given as the label patches.

# Chapter 5

# Model Analysis

As described in the previous chapters, the convolutional neural network (CNN) can perform accurate semantic segmentation for aerial imagery by optimizing the parameters in the network. However, the reason why the trained convolution filters can act as a efficient feature extractor is unclear. Then, some methods to analyze the trained convolution filters and what the network actually represents have recently been developed in computer vision literature [Zeiler and Fergus, 2014, Mahendran and Vedaldi, 2015].

A simple way to analyze the first convolutional layer which directly connects to the input image layer is to show each channel of filters as an image. It is usually used for interpreting the low-level feature extraction part by a CNN. The filters in the first convolutional layer is visualized by tiling to construct an image such as Figure 5-1. Each figure in Figure 5-1 shows 64 $16 \times 16$-sized filters in $C(64, 16 \times 16/4)$ layer in our architecture (Figure 3-3). In this case, each filter has three channels, so that it can be shown as a three-channel RGB image. Then we tile each image which shows one of the filters in $C(64, 16 \times 64/4)$ layer to construct the figure as shown in Figure 5-1.

Figure 5-1a shows the first layer convolutional weights of *Multi-channel* (without CIS), and Figure 5-1b shows the weights of *Multi-channel with CIS*. The first convolutional layer for low-level feature extraction usually has similar filters even in other models trained for different tasks. Also in this case, similar filters appear in the first layers of both models, while a filter to extract a specific pattern appears at difference

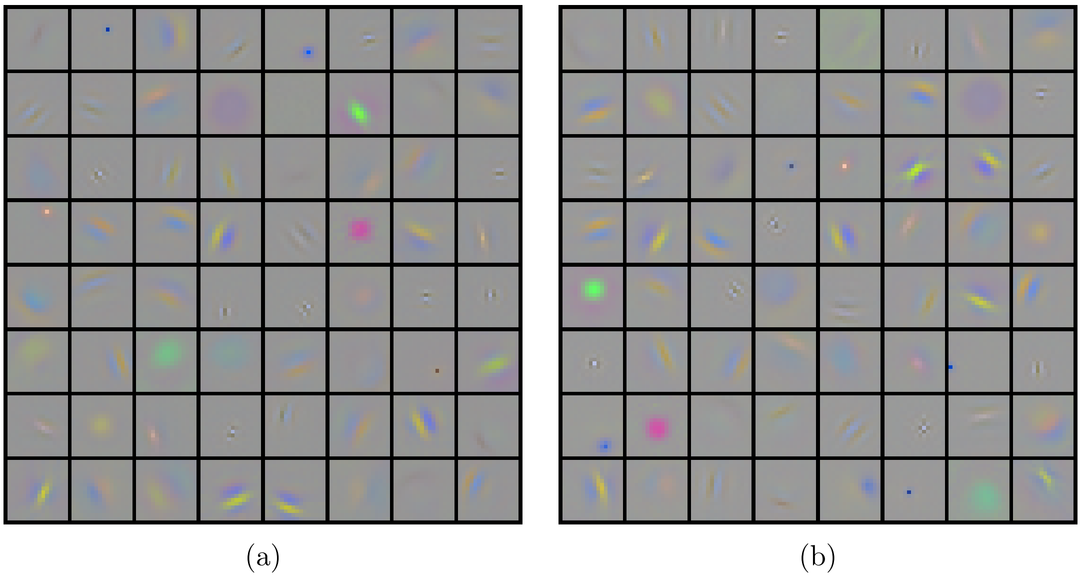


Figure 5-1: Example of the first convolution layer: (a) *Multi-channel* (b) *Multi-channel with CIS*

location in the tiled images. Features to be extracted in the lowest convolutional layer for semantic segmentation task are not so different from the case of object recognition. In many cases, the first layer commonly obtains filters that response colors or edges.

This type of visualization can gain intuition about the function of the lowest layer of a CNN, but it cannot be performed to the upper layers. Then, in this chapter, we visualize the outputs of the middle layers of trained CNNs to investigate what the middle layers' filters obtained through the training stage. Then we discuss about the difference between *Multi-channel* and *Mmulti-channel with CIS*. As described in Chapter 3, the architectures used in both models are the same, but the activation function in the final layer of the CNN is different. Thus, we investigate the effect of the activation function in terms of the obtained feature extractors. We also apply the visualization method to investigate the representations that are obtained in the middle layers of a CNN, which is proposed by Mahendran et al. [Mahendran and Vedaldi, 2015] to those models.

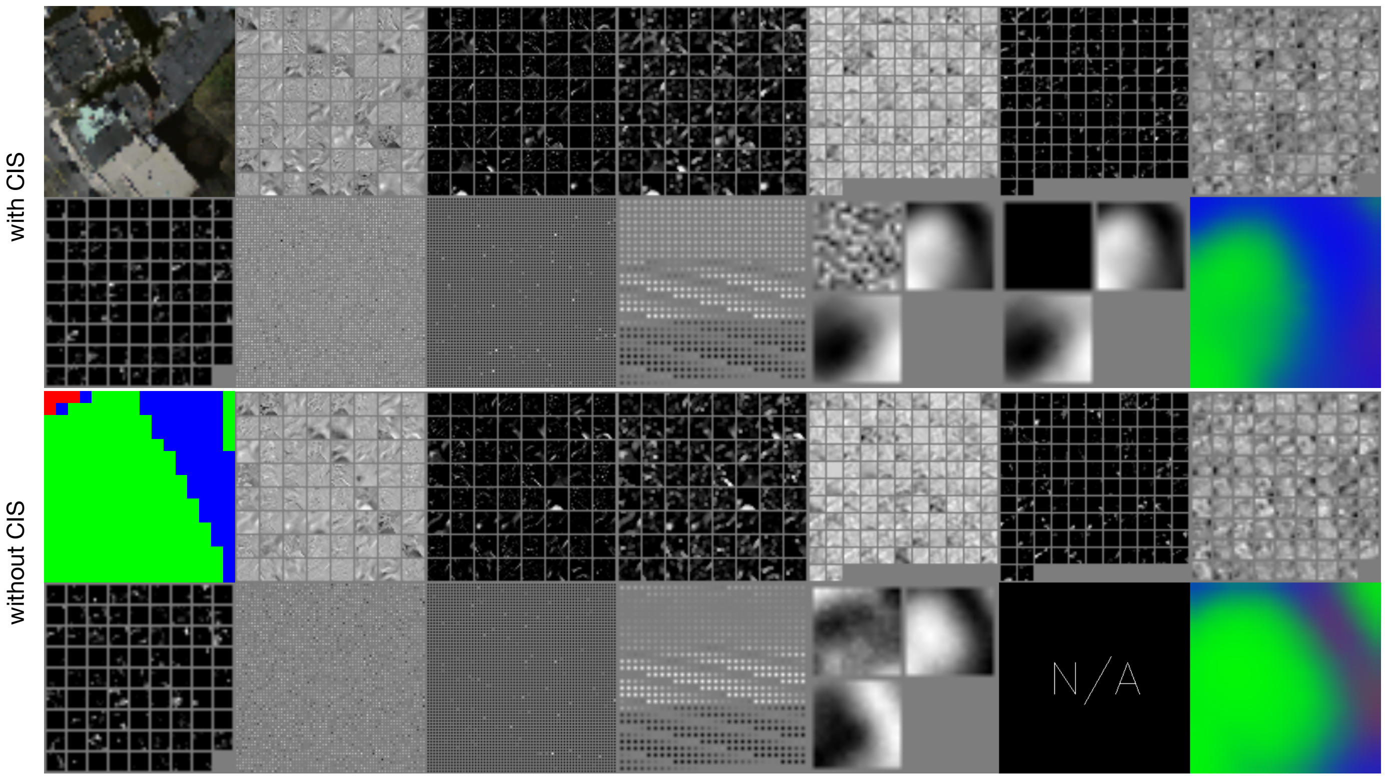
## 5.1 Outputs of The Middle Layers

We extract the intermediate outputs of middle layers from trained CNNs. Figure 5-2 shows the tiled outputs of middle layers. As described in Chapter 3, our CNN architecture is $C(64, 16\times16/4)$ - $P(2/1)$ - $C(112, 4\times4/1)$ - $C(80, 3\times3/1)$ - $FC(4096)$ - $FC(768)$. In this figure, we describe the outputs of these layers as conv1, mpool1, conv2, conv3, fc4, and fc5, respectively. Then, these layers except mpool1 have ReLU activation functions, so that the activated values of conv1, conv2, conv3, and fc4 are shown as relu1, relu2, relu3, and relu4, respectively, in this figure. The output of the fc5 layer is reshaped into $16 \times 16$-sized three-channel image patch, so that the reshaped output is described as reshape in this figure. Then, the CIS operation that gives zero to the background channel is applied to the reshaped output, then the result of CIS operation is described as cis in this figure.

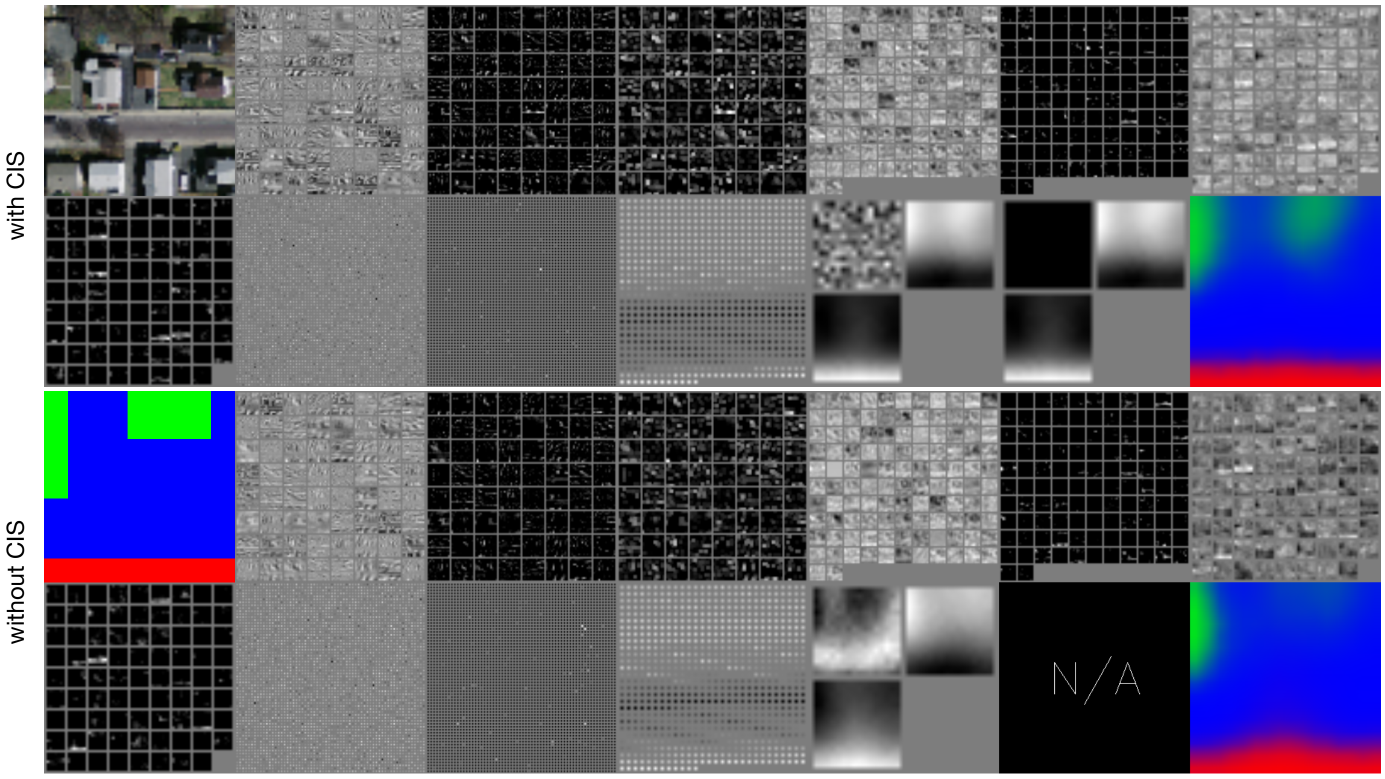
As described in Subsec. 2.4.1, when $K$-channel $W \times W$-sized feature maps are given to a convolutional layer that has $M$ $H \times H$-sized filters, the output is $M$ $(W - H + 1)/s \times (W - H + 1)/s$-sized feature maps, where $s$ is a stride parameter of the convolution. Thus, a single convolution layer that has $M$ filters has $M$ feature maps as its output. In our model, the input image is 3-channel $64 \times 64$-sized image and the first convolutional layer has 64 $16 \times 16$-sized filters and the stride parameter is 4, so that the output is 64 $(64 - 16 + 1)/4 \times (64 - 16 + 1)/4$-sized feature maps. Then, we tile those 64 feature maps to construct a grayscale image. Figure 5-2 shows the tiled feature maps.

Figure 5-2 includes the outputs from two different models for two example inputs. In each example, the square blocks which tile feature maps are, from the left top block to the right bottom block, an input image (in the *with CIS* row) or the true label image (in the *without CIS* row), conv1, relu1, mpool1, conv2, relu2, conv3, relu3, fc4, relu4, fc5, reshape, cis, and predicted label image, respectively, in horizontal writing order.

Focusing on the output of reshape layer in Figure 5-2a (the third block from the last), the left-top patch in this block is very noisy in the case of *with CIS*, because



(a)



(b)

Figure 5-2: The outputs of middle layers. For each example, the upper two rows show the results of *ours (multi-channel with CIS)* and the lower two rows show the results of *ours (multi-channel)*.

no gradients are given to all of the weights connected to this channel because of CIS operation. This channel corresponds to the background channel, and no weight updates are performed to weights which connect to this background channel. On the other hand, in the case of *without CIS*, the gradients are also given to the weights which connect to the background channel, so that the left-top patch in the reshape block of this model shows the prediction of the background class. This is the obvious difference between *with CIS* and *without CIS)*, which can be seen from these example aerial image patches. From these differences, we found that the model *with CIS* never try to predict the background. Because, if there are any errors in the prediction of the background channel, the loss values calculated by using CIS will never be changed. It causes that the model *with CIS* never learn anything about the background class, so that the representation which should be complicated compared to the other simpler classes such as road and building, will also never be obtained through the training of the CNN with CIS.

### 5.1.1 Difference in The Final Prediction Results

We further explore the difference between the final prediction outputs of these different models. Figure 5-3 shows seven examples that consist of pairs of input aerial imagery patch and the corresponding label patch with the outputs of the reshape layer and the final prediction results. The left-top matrix in this figure shows which block means what. The prediction results have continuous values ranging from 0 to 1 in each channel, and the each channel shows each object class, namely, roads, buildings, or the background. Those color patches shows the result of stacking the object label prediction results as different channels.

In all the examples shown in Figure 5-3, the predictions of *Multi-channel with CIS* seem to be better than the results of *Multi-channel* (no-CIS). In the case of *Multi-channel* (no-CIS), the reshape layer output shows that it predicts the background class as well as the other two classes, buildings and roads. To predict the background class, the model should be able to extract the good features to represent the background, so that the lower layers should perform the feature extraction not only for the target

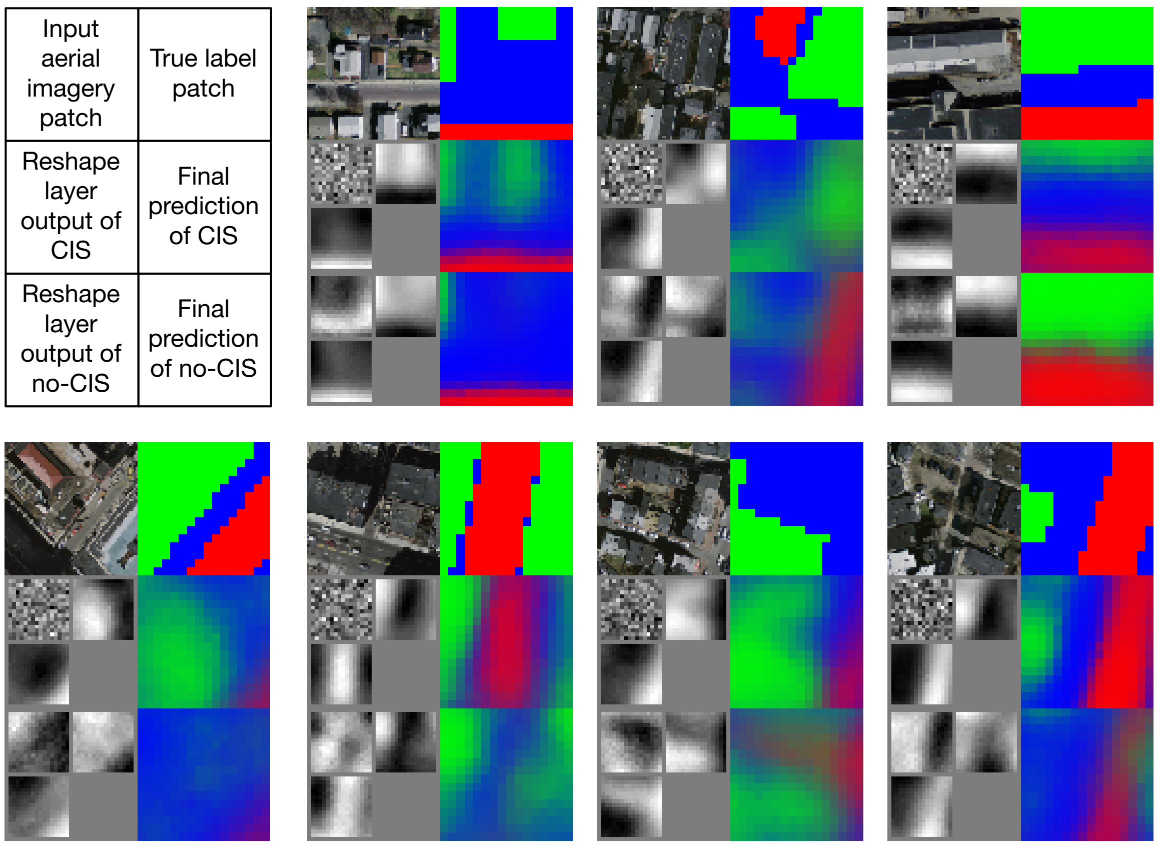

Input aerial imagery patch
True label patch
Reshape layer output of CIS
Final prediction of CIS
Reshape layer output of no-CIS
Final prediction of no-CIS

Figure 5-3: Comparison of the reshape layer outputs and the final prediction between ours (multi-channel) and ours (multi-channel with CIS)

classes, buildings and roads, but also for the background class. It might cause the increase of the problem complexity during training of a CNN model. Indeed, in all the results shown in Figure 5-3, focusing on the right columns in all blocks, the final predictions of CIS always seem to be better than the results of no-CIS. The model without CIS often lacks either buildings or roads in these example patches.

## 5.1.2 Inverting Feature Maps to Image

Finally, we perform the method for image reconstruction from the feature maps proposed by Mahendran et al. [Mahendran and Vedaldi, 2015] to a trained *Multi-channel with CIS* model. This method optimizes a image which is initialized with random values sampled from a carefully designed distribution with some parameters which are calculated based on the training dataset. The objective of the optimization is

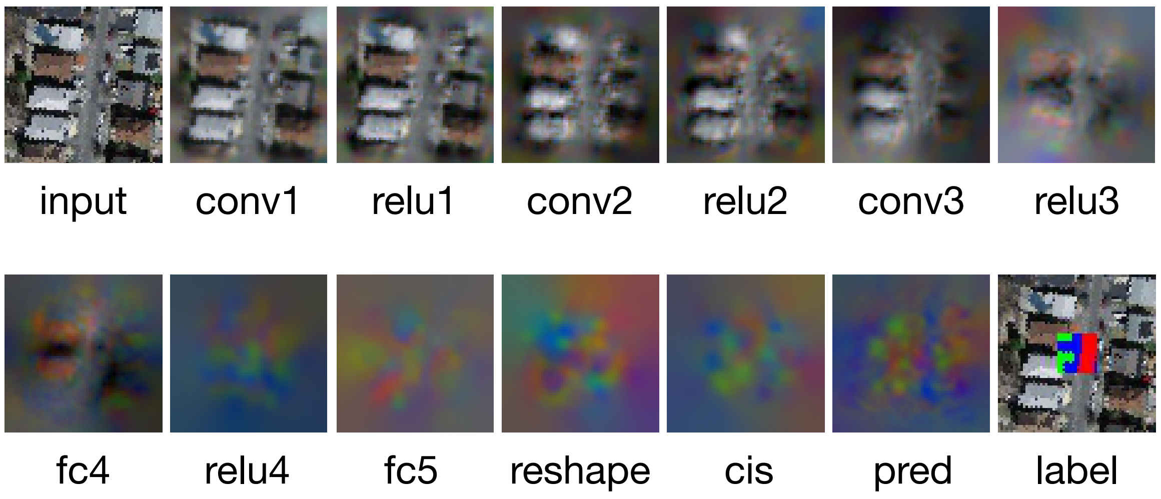


Figure 5-4: Comparison of the reshape layer outputs and the final prediction between ours (multi-channel) and ours (multi-channel with CIS)

to minimize the difference between the middle layer outputs which is obtained by feed-forward of the image to be optimized and the middle layer outputs which is preliminary calculated by feed-forward of an original aerial image patch. Is is actually to reconstruct an aerial image patch from a set of feature maps or a feature vector which is obtained from a middle layer of a trained CNN. The resulting reconstructed image and the original aerial image patch have similar meanings for the network, because both of the resulting image and the original image have the similar middle layer values after feed-forwarding. This method can be used to investigate how the network see the input image.

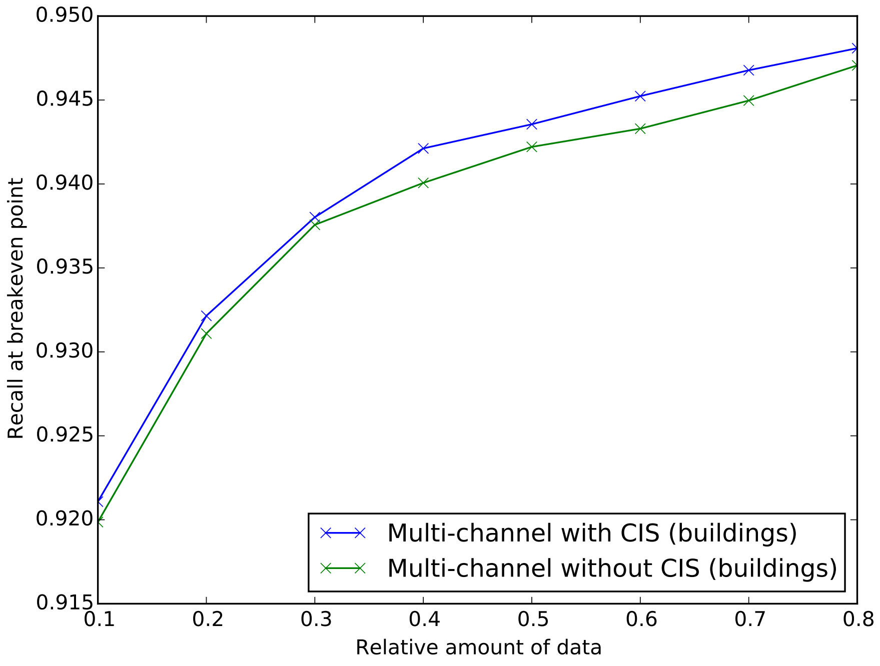
Figure 5-4 shows a reconstruction result with an example aerial image patch chosen from the test set of *Mass. BR*. From this figure, we found that the receptive field is getting narrower and narrower as the target layer gets higher. Because, the convolution operation without padding makes the size of input feature maps smaller, so that the information around the border is lost as an input image goes through convolutional layers. Additionally, our models output the smaller sized prediction ($16 \times 16$) than the input aerial image patch ($64 \times 64$), so that the network focuses more on the center region of the input image field. This might also causes that the areas around the border is blurred in the higher layers. From the lower layer results under relu3, we see that there are still much information of the input aerial image patch. It might mean that the filters in conv1, conv2, and conv3 keep as much information as possible during performing feature extractions to transform the aerial image patch to a predicted label image patch as shown in the center region of the right bottom block in Figure 5-4. Taking into account the activation functions, ReLU, the network perform non-linear transformations repeatedly and gradually transform data through many layers in the network.

## 5.2 The Effect of The Amount of Data

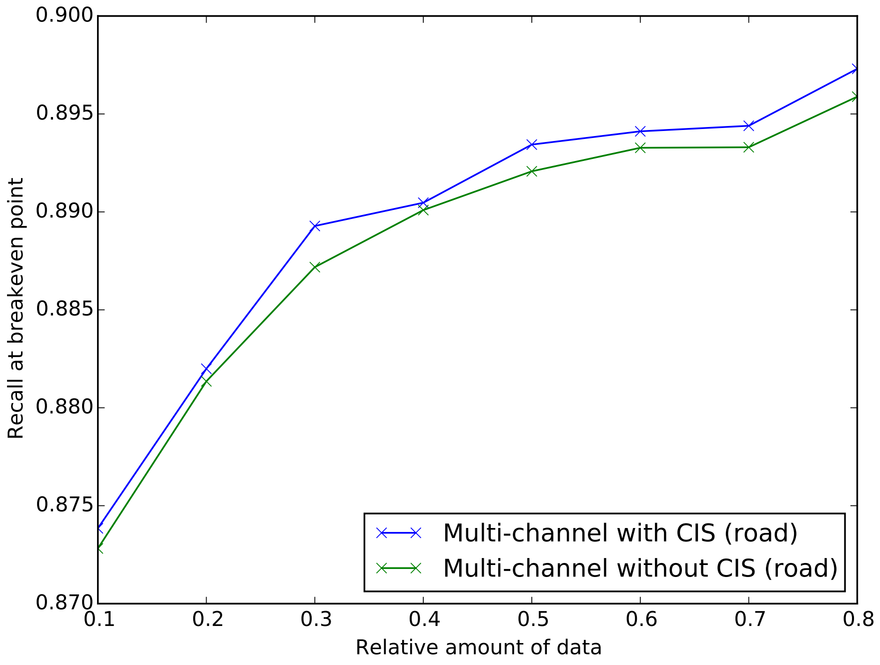
Convolutional neural network such as our proposed architecture essentially needs large amount of data to be used for training. However, in general, collecting many labels that are obtained by complex attentional task for human experts is very difficult, so that if the CNN can be trained well with less amount of data, it is more effective in practical situations. Therefore, we investigate the relationship between the final accuracy of a model and the size of training dataset.

We perform the same experiments that are described in Chapter 4 with different size of the training dataset. First, we randomly extracted aerial image patches from the training set of *Mass. BR* util the total number of extracted patches reaches 10 % of the number of whole patches in the training set. Then, we trained two models, *Multi-channel* and *Multi-channel with CIS*, with the extracted patches and tested on the same test data which is used in Chapter 4. We also perform this experiment with 20 %   80 % of the whole patches.

Figure 5-5 shows the results. In this figure, the results of *Multi-channel with CIS* are almost always better than the case of *Multi-channel without CIS*. The largest difference between the two models occurs at 0.3 of the relative amount of data in the building prediction result, and its difference is 0.21 %. This is quite large difference compared to the difference of the results shown in Table 4.2 (0.05 %). Therefore, we found that the effectivity of the CIS is significant when the available data is scarce, because the amount of improvements in the performance is larger when the size of dataset is smaller than the case of the result using whole data in the dataset. It is very strong advantage on many practical situations.



(a)



(b)

Figure 5-5: Comparison of the accuracy of models when the dataset size is changed. (a) The results of building extraction. (b) The results of road extraction.

# Chapter 6

# Aerial Imagery Mosaicking

In this chapter, we propose a new cost map calculation method based on the results of semantic segmentation that considers the meaning of pixels. Our method uses a CNN to generate a building probability map that is used as a cost map. The CNN is trained on a large-scale dataset consists of many aerial orthoimages and corresponding building mask images. Once the CNN is trained, we never need to select any parameters to perform semantic segmentation on an aerial image when using the trained CNN. We then find the minimum cost path on the cost map as in the conventional method [Pan et al., 2014] by applying Dijkstra's algorithm [Dijkstra, 1959]. Finally, we perform our proposed method and the conventional method on 15 sets of aerial images that have overlapping regions and cover larger areas compared to the images used for the evaluation of the conventional method in Pan et al. [Pan et al., 2014]. Then we show that our seamlines never cross any buildings, while the seamlines produced by the conventional method pass through many buildings, because the optimality of seamlines can be defined by how many buildings are passed through by the resulting seamlines.

## 6.1 Semantic Segmentation for Image Mosaicking

The process flow for our method is shown in Figure 6-1. Our goal is to determine the optimal seamline in an overlapping region between two aerial images. As shown in

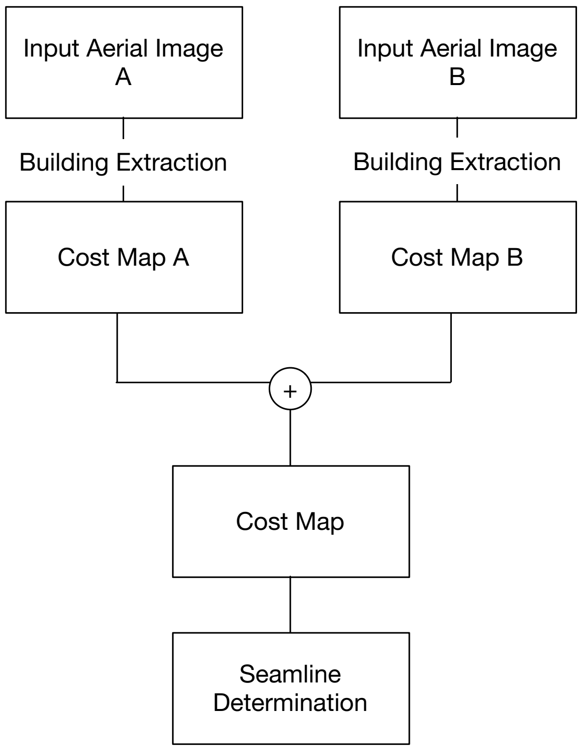


Figure 6-1: The process flow for our proposed method.

Figure 6-2, once two aerial images are given, an overlapping region is uniquely found and the start and end points of a seamline are also determined as the crossing points of these images. Therefore, the objective is to find the minimum cost path between the start point and the end point inside the overlapping region. Thus, the cost map matters. We assume that the optimal seamline never crosses any buildings, so we design the cost map as a building probability map.

We propose a new cost map that uses the results of semantic segmentation. To perform semantic segmentation, we train a CNN to obtain a mapping from raw pixel values in an aerial image to a building probability map. We first prepare a dataset that includes 127 aerial images and corresponding building label images. The resolution of all these images is 0.5 $m^2$/pixel in common. This dataset covers roughly 625 $km^2$. All of the building labels are binary and have a 1 for a pixel that belongs to building and a 0 for a background pixel. The dataset is divided into two groups, 121 training images and six test images. An example pair of an aerial image and the corresponding label image chosen from the test dataset is shown in Figure 6-3a and Figure 6-3c. We

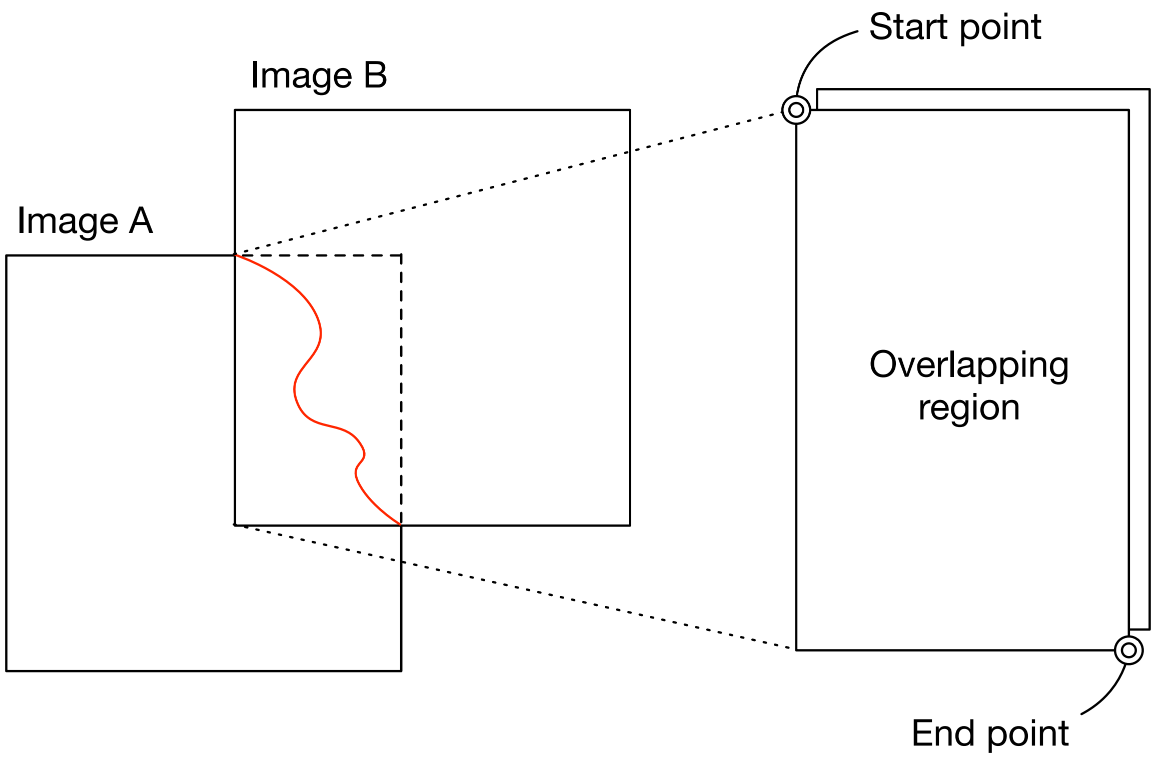


Figure 6-2: An overlapping region between two aerial images and start and end points for finding an optimal seamline

train a CNN called VGG-16 that has the architecture proposed by Simonyan et al. [Simonyan and Zisserman, 2014] Table 6.1 shows the network configuration. This table follows the method of summarizing the network configuration as in Parkhi et al. [Parkhi et al., 2015]. In this table, the only layers that have trainable parameters are underlined.

We formulate the segmentation task for a large aerial image by using a patch-based approach as described in Chapter 3, so that the input image is a $w_s \times w_s$-sized three channel color patch, and the output from the CNN is a $w_m \times w_m$-sized single-channel patch. Each pixel $x_i$ at location $i$ in the output patch is converted into a probability using a sigmoid function $\hat{m}_i = 1/(1 + \exp(-x_i))$, where $\hat{m}_i$ denotes a predicted label probability at pixel location $i$. Once a $w_s \times w_s$-sized aerial image patch $\mathbf{s}$ is given, the CNN estimates the building probability $p(\tilde{\mathbf{m}} = 1|\mathbf{s})$, where $\tilde{\mathbf{m}}$ denotes a correct label patch. Here, we assume that all pixels in the estimated patch are independent

Table 6.1: Network configuration

| Layer | Type | Number of filters | Filter size | Stride | Pad | Activation |
|---|---|---|---|---|---|---|
| 0 | input | - | - | - | - | - |
| $\underline{1}$ | conv | 64 | 3 | 1 | 1 | ReLU |
| $\underline{2}$ | conv | 64 | 3 | 1 | 1 | ReLU |
| 3 | max pooling | - | - | 2 | 0 | - |
| $\underline{4}$ | conv | 128 | 3 | 1 | 1 | ReLU |
| $\underline{5}$ | conv | 128 | 3 | 1 | 1 | ReLU |
| 6 | max pooling | - | - | 2 | 0 | - |
| $\underline{7}$ | conv | 256 | 3 | 1 | 1 | ReLU |
| $\underline{8}$ | conv | 256 | 3 | 1 | 1 | ReLU |
| $\underline{9}$ | conv | 256 | 3 | 1 | 1 | ReLU |
| 10 | max pooling | - | - | 2 | 0 | - |
| $\underline{11}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| $\underline{12}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| $\underline{13}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| 14 | max pooling | - | - | 2 | 0 | - |
| $\underline{15}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| $\underline{16}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| $\underline{17}$ | conv | 512 | 3 | 1 | 1 | ReLU |
| 18 | max pooling | - | - | 2 | 0 | - |
| $\underline{19}$ | fc | 4096 | 1 | - | 0 | ReLU |
| $\underline{20}$ | fc | 4096 | 1 | - | 0 | ReLU |

of each other, so the objective probability is defined as below:

$$p(\tilde{\mathbf{m}}|\mathbf{s}) = \prod_{i=1}^{w_m^2} p(\tilde{m}_i|\mathbf{s}), \tag{6.1}$$

where $p(\tilde{m}_i|\mathbf{s})$ denotes a building probability at pixel $i$, and is defined using a Bernoulli distribution:

$$p(\tilde{m}_i|\mathbf{s}) = \hat{m}_i^{\tilde{m}_i}(1 - \hat{m}_i)^{1-\tilde{m}_i}. \tag{6.2}$$

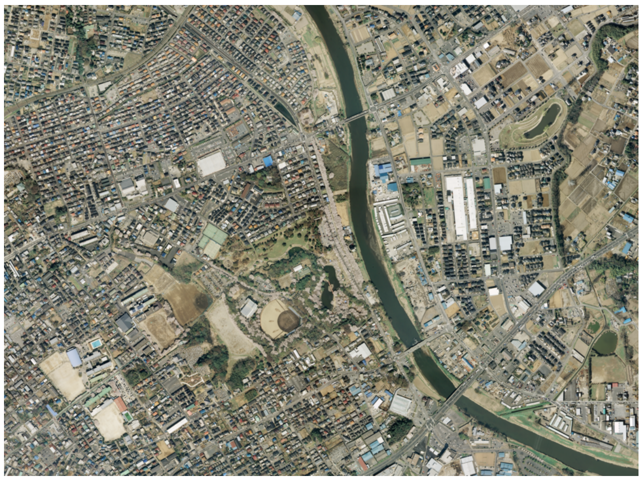
Therefore, to maximize the probability described in Eq. 6.1, we minimize the negative log-likelihood defined below:

$$\mathcal{L} = -\sum_{i=1}^{w_m^2} \left( \tilde{m}_i \ln \hat{m}_i + (1 - \tilde{m}_i) \ln(1 - \hat{m}_i) \right), \tag{6.3}$$

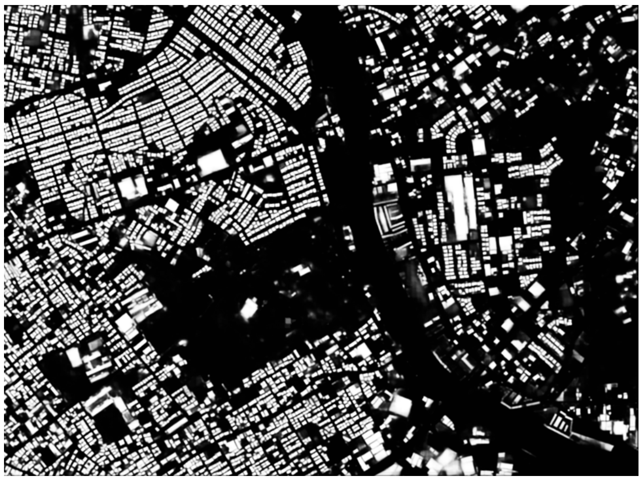
where $\tilde{m}_i$ is the correct label at pixel $i$. In this chapter, all aerial images in the training set are divided into $128 \times 128$-sized patches, so that $w_s = 128$, and we set the output patch size to be $w_m = 32$. Setting the size of an output patch to be smaller than that of an input patch improves the performance because of context utilization [Mnih, 2013]. The original form of the output layer of VGG-16 is a 1000-dimensional vector for image classification, so we modify the output layer with a fully-connected layer that has 1024 units. The output layer is always reshaped to a $32 \times 32$-sized single channel patch immediately after feed-forwarding.

We train all parameters in the CNN end-to-end by minimizing the negative log-likelihood (Eq. 6.3) using mini-batch stochastic gradient descent with momentum. During training, we reduce the learning rate by multiplying by a fixed reduction rate every $\tau$ iterations. Furthermore, we regularize the network using L2 weight decay. Therefore, the hyper-parameters in the learning stage are the mini-batch size, the learning rate (LR) $\eta$, the LR reduction rate $\gamma$, the LR reduction frequency $\tau$, the weight of the momentum term $\alpha$, and the weight of the L2 weight decay $\beta$. The learning rate $\eta$ starts from $\eta_0$, and we use these values for all experiments in this chapter: $\eta_0 = 0.0005$, $\tau = 10^4$, $\gamma = 0.1$, $\alpha = 0.9$, $\beta = 0.0005$, and the mini-batch size is 64. All of these values except the mini-batch size are completely the same as those used in Saito et al. [Saito and Aoki, 2015, Saito et al., 2015] and Mnih [Mnih, 2013]. The mini-batch size is modified because of the limitation of GPU memory to train the deep model, VGG-16.

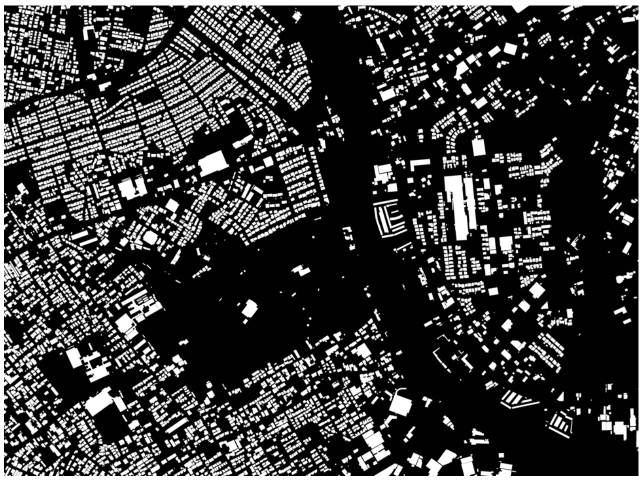
Following this patch-based formulation, this CNN can produce a predicted building probability map from a raw aerial image input. Therefore, there is no need to perform pre-processing on input images. Figure 6-3a shows an example input image. Figure 6-3b shows the predicted building probability map. Figure 6-3c shows the correct binary label map. The output building probability map is created by tiling



(a) An aerial image



(b) An example result of semantic segmentation (grayscale)



(c) The corresponding label image (binary)

Figure 6-3: Images used for training of the CNN and the result example

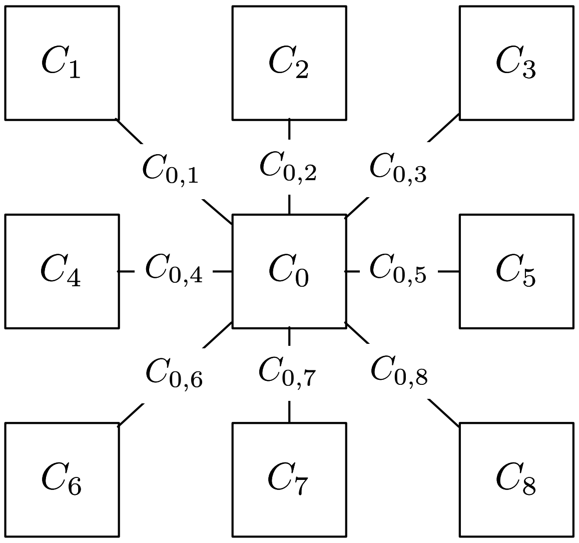


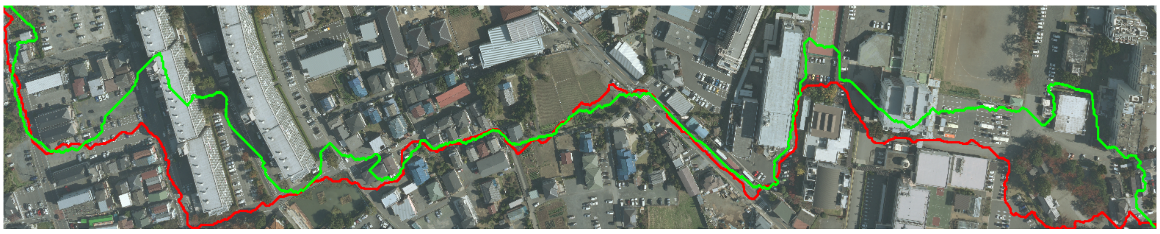
Figure 6-4: A node and connected neighbor nodes

the output patches of the CNN.

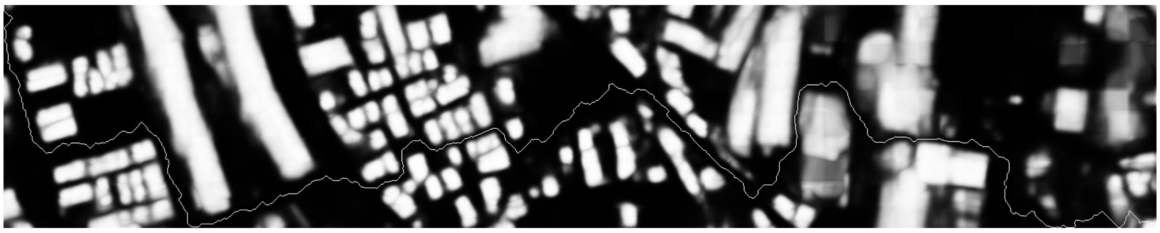
We create two building probability maps from both overlapping regions from Image A and Image B, as shown in Figure 6-2, using the trained CNN. We then combine the two resulting maps by adding them and create a single cost map as shown in Figure 6-5b. A pixel in this integrated cost map would have a large value (the maximum value is 2) when its location is considered as belonging to a building from the viewpoints of both input images A and B. We then convert this integrated cost map into a graph that treats each pixel as a node. All eight neighbor nodes are connected as shown in Figure 6-4. A weight between two nodes $C_{i,j} (i \neq j)$ is defined as below:

$$C_{i,j} = |C_i - C_j|. \tag{6.4}$$

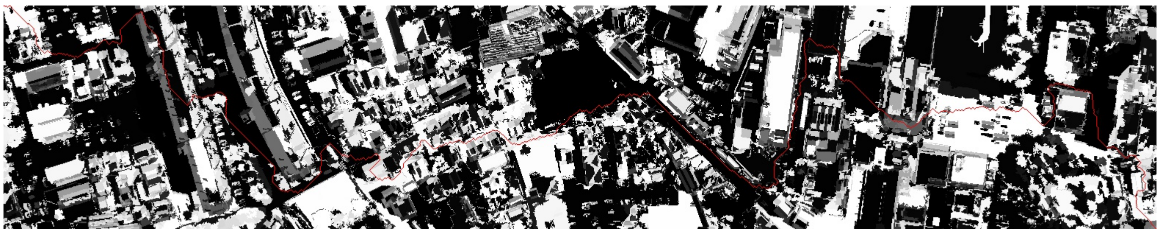
In the integrated cost map, it should be noted that all pixel values are positive. Hence, we can finally calculate the shortest path on this graph using Dijkstra's algorithm [Dijkstra, 1959]. The start point and end point are fixed beforehand according to the crossing points as shown in Figure 6-2.



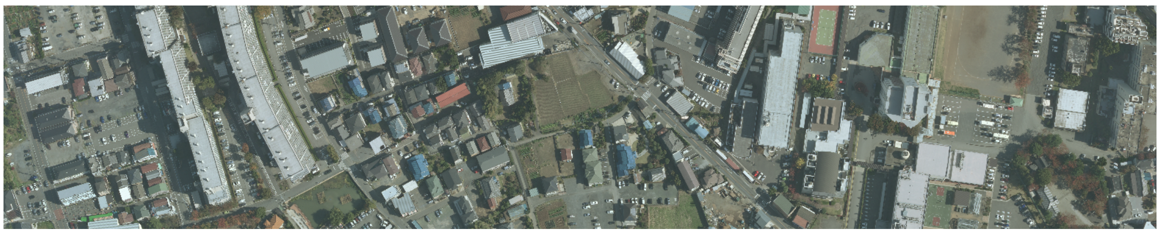
(a) The green line and the red line show the seamline found by the conventional method [Pan et al., 2014] and the seamline found by our method, respectively. Both seamlines are drawn on an input aerial image.



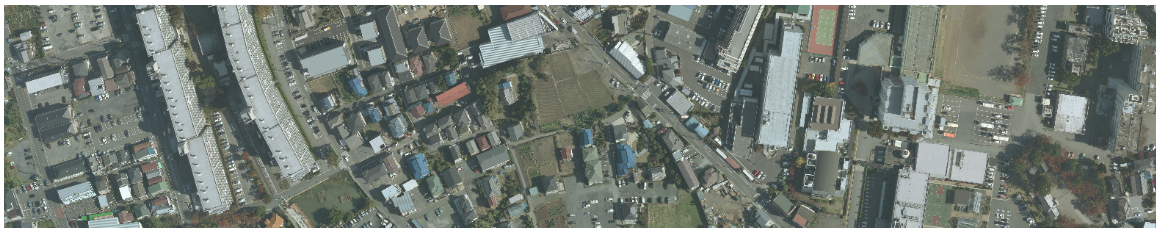
(b) The grey line shows the seamline calculated on a predicted building probability map (grayscale).



(c) The red line shows the seamline determined using the conventional method [Pan et al., 2014]. The seamline is drawn on this cost map (grayscale) calculated by the method.



(d) The final mosaic image that used the seamline found by the conventional method [Pan et al., 2014]



(e) The final mosaic image that used the seamline found by our method

Figure 6-5: Experimental results. All images are 1570-sized and the resolution is 0.5 $m^2$/pixel.

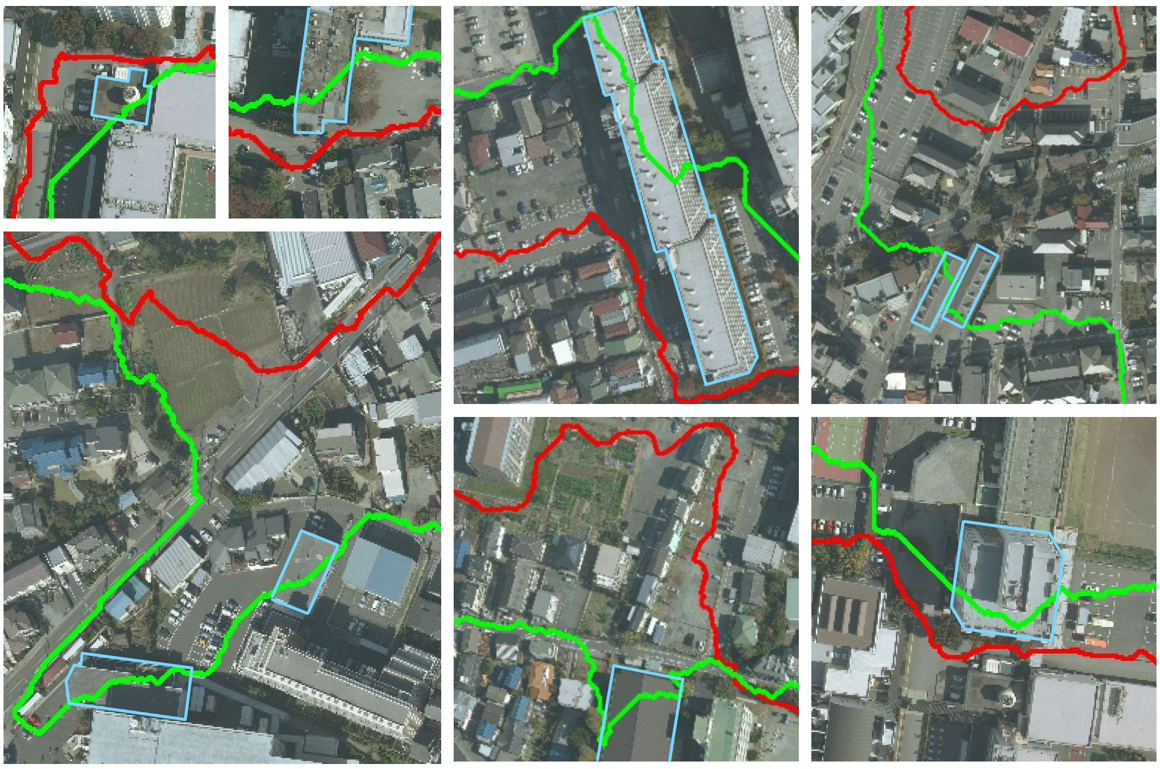


Figure 6-6: Red lines show the seamlines found by our method. Green lines show the seamlines found by the conventional method[Pan et al., 2014]. The light blue lines denote buildings that are passed through by seamlines.

## 6.2 Experimental Results

We first evaluate our semantic segmentation method for building extraction on the aerial images and building label dataset. The evaluation metric is relaxed precision and recall as described in Section. 4.1 in Chapter 4. In all experiments in the previous chapter, the slack parameter $\rho$ is set to 3, but we use $\rho = 6$ because the resolution of our input aerial images is twice the resolution of the images used in the previous experiments. This makes it comparable in terms of the ground measurement because in both errors within 3 m are allowed. We then summarize the relaxed precision and recall values over 256 different thresholds with a single recall at the breakeven point.

The recall at the breakeven point on the test dataset was 0.9984 with $\rho = 6$. Additionally, the values were 0.9969 for $\rho = 3$ and 0.8680 for $\rho = 0$. $\rho = 0$ indicates the exact precision and recall. These results show that our CNN can predict building pixels very accurately. An example result is shown in Figure 6-3b. A pixel in a

Table 6.2: The number of buildings passed through by seamlines.

| Method | Number of buildings passed through | Elapsed time |
|---|---|---|
| Pan et al.[Pan et al., 2014] | 54 | 1589.11 sec |
| Ours | 0 | 1762.66 sec |

predicted label image that is the result of semantic segmentation performed by our trained CNN shows the probability of building existence at the pixel. Therefore, all pixels have values ranging from 0 to 1, which are always positive.

To evaluate our seamline determination method and compare it with the conventional method in Pan et al. [Pan et al., 2014], we prepared 15 aerial image pairs that have overlapping regions. All of these images have a resolution of 0.5 $m^2$/pix, and the overlapping regions in the aerial image pairs cover roughly 6.6 $km^2$. This is larger area than the test image used in the paper where the conventional method was proposed [Pan et al., 2014].

We perform semantic segmentation for all overlapping region pairs. We obtain cost maps of both overlapping regions for an aerial image pair and combine them to generate an integrated cost map as shown in Figure 6-5b. We then calculate the shortest path on the integrated cost map. The start and end nodes are selected as the intersection points of image boundaries for two input aerial images as shown in Figure 6-2. All pixels in the cost map are converted into connected nodes, that is, a graph (see Figure 6-4). We perform Dijkstra's algorithm [Dijkstra, 1959] to find the shortest path between the start node and the end node. The resulting path is considered to be a seamline. An example result is shown as the red line in Figure 6-5a. We performed this seamline determination method for all 15 integrated cost maps.

To evaluate the quality of seamlines, we projected the resulting seamlines onto the original aerial images such as Figure 6-5a and counted the number of buildings that are passed through by the resulting seamline manually. Then, as shown in Table 6.2, we found that none of the seamlines determined by our method pass through any buildings in this case.

Finally, we apply the conventional method proposed by Pan et al. [Pan et al., 2014] to compare the results with ours. They used images that have the same resolution (0.5 $m^2$/pix) as our experiments, so we used the same mean-shift parameters in their paper. We first used the Edge Detection and Image SegmentatiON (EDISON) library [Christoudias et al., 2002] to perform mean-shift segmentation over all aerial images. The parameters for EDISON were chosen to be the same as in the paper, namely, $(h_s, h_r, M) = (6, 5.5, 15)$, where $(h_s, h_r)$ are bandwidth parameters and $M$ is the least significant feature size used in the library. We then extracted the preferred regions, defined as segments that have the size larger than $s_T$, appearing in the results for EDISON. The parameter $s_T$ is defined as *width* $\times$ *height* for the bounding box of the largest object segment. However, in the paper that proposed the conventional method [Pan et al., 2014], the determination of $s_T$ was unclear. Pan et al. [Pan et al., 2014] mentioned that $s_T$ can be estimated from the actual size of the largest object in the ground coverage, but the actual size of the largest object varies depending on the input aerial image. Therefore, we looked for the largest building in all 15 overlapping region pairs. We found that the bounding box for the largest building was $167 \times 248$ pix$^2$, so we used $s_T = 167 \times 248 = 41416$.

We converted all the pairs of aerial images into hue-saturation-value (HSV) color space and extracted the value (V) channel representing the brightness of each pixel in the images. Then a cost map was calculated as the absolute difference between the two aerial images across the value (V) channels. According to [Pan et al., 2014], all difference values in the preferred regions of a cost map are multiplied by 0.01, so that segments larger than $s_T$ have smaller costs in the resulting cost map. We then perform Dijkstra's algorithm in the same way as [Pan et al., 2014] on the calculated cost maps. An example result is shown as the green line in Figure 6-5a. We also counted the number of buildings that are passed through by the seamlines found by the conventional method. As shown in Table. 6.2, the seamlines passed through 54 buildings within the 15 overlapping regions.

Furthermore, to investigate the effect of the value of $s_T$, we tested 10 different values ranging from 1000 to 200000. The average number of buildings that are passed through by the seamlines found by the conventional method with the different $s_T$

values was 54.5 and the standard deviation was 2.33. The minimum and maximum were 51 and 57, respectively. The small variance indicates that the actual value of $s_T$ does not have a big impact on the performance.

The third column of Table. 6.2 shows the elapsed time for solving the shortest path problem over all 15 cost maps generated by each method. The computational complexity for finding the shortest path does not differ much between the two methods in terms of worst-time complexity, because both methods use Dijkstra's algorithm to solve the problem. The measurement environment for the elapsed time shown in Table 6.2 was an Intel Core i7-5960X (3.00 GHz) CPU with 64 GB memory.

## 6.3 Discussion

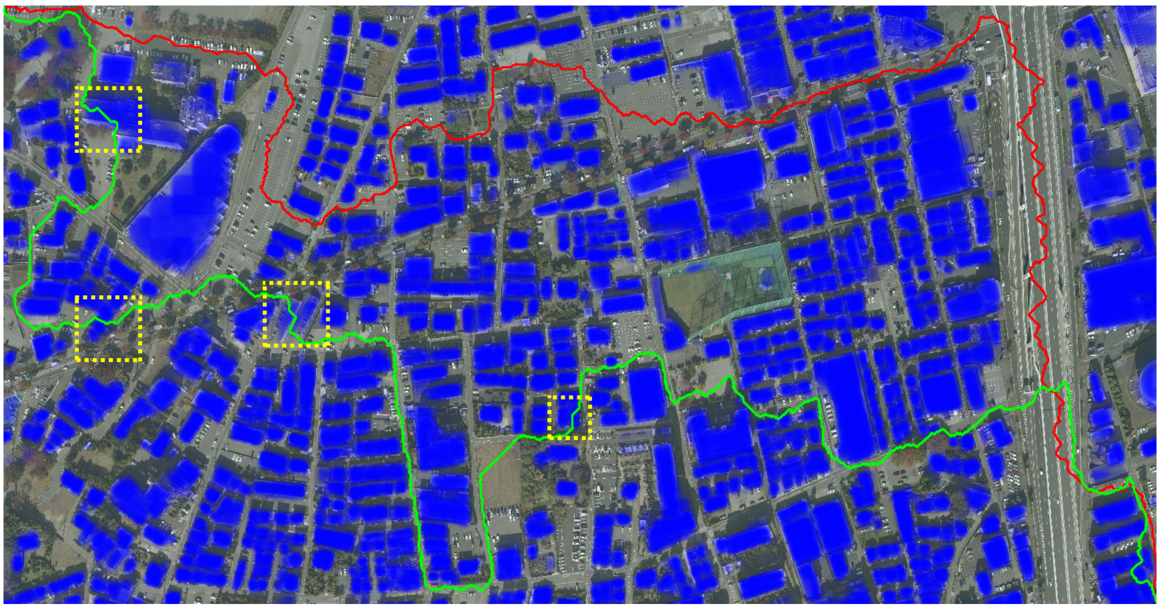
Figure 6-6 shows some of the resulting seamlines. The green lines are found by the conventional method, while the red lines are found by our method. As shown in these figures, some buildings have similar colors as the surrounding ground or span larger areas than neighboring ground that looks like parking areas. Also, all such buildings are larger than the houses in the residential area. These observations may be the reason why those regions are not considered to be preferred regions and are passed through by the seamlines as a result. However, our semantic segmentation method correctly extracted all the buildings enclosed by light blue lines, so that all red lines in Figure 6-6 avoid passing through any buildings. The preferred regions based on mean-shift segmentation might treat all of the large regions filled with similar colors to be non-object even if they are actually large buildings, because mean-shift segmentation does not consider the meaning of pixels.

An example of the final mosaicking results are shown in Figure 6-5d and Figure 6-5e. The former is the result obtained by using a seamline found by the conventional method, while the latter is generated using the seamline found by our method. In Figure 6-5d, there are two unusual appearance around two large buildings. Both are caused by the seamline that passes through those buildings, while the result of our method shows natural appearance around them.

Figure 6-7a and Figure 6-7b show some further examples. Buildings detected by our method are filled with blue. The green lines and the red lines have the same meaning as in Figure 6-6. The yellow rectangles show the areas that have buildings that are passed through by the green seamline.

As shown in Figure 6-7a, the red line crosses many parking areas and roads appeared in the upper area of this figure, while the lower area of this figure is covered by many houses and buildings. In this aerial image, reaching the wide road located on the right side is crucial when choosing the path for a seamline. The red line successfully avoids the residential area and, with the help of accurate building extraction results, passes through the sparse zones that can be seen in the upper half of this figure.

In Figure 6-7b, the green line suddenly goes through a residential area after crossing the bridge located in the centre of this figure. This is because there are many shadows cast on the road in the centre of this figure to the right side of the bridge, so that the cost values in this region of the difference image for the two original aerial images used as the basis for the cost map for the conventional method are large. The yellow rectangle in Figure 6-8 shows the region that has building shadows. The leftmost figure shows the cost map used for the conventional method that is created by calculating the difference between V (value) channels of two overlapping regions. In the leftmost figure, the region that is enclosed by the yellow rectangle has large white areas that have the same shape as the shadows shown in the rightmost picture in Figure 6-8. White pixels have large cost values, so that the red seamline couldn't pass through the road. Because the two aerial images are captured at different time, shadows could appear only in either of the aerial image pair. Therefore, calculating difference of pixel intensities of those images could make shadow regions have high cost. However, our semantic segmentation method succeeded in finding only buildings without being affected by shadows (see Figure 6-8b). Therefore the cost values on the road are small in our method. This is why our seamline succeeded to pass through the wide road located on the right side of the bridge. Furthermore, in Figure 6-7b, there is a large building that appears to be an elementary school beside a school


(a) Red lines denote the seamlines determined by our method. Green lines denote the seamlines determined by the conventional method [Pan et al., 2014]. The red lines pass through many parking areas and roads, while the green lines pass through residential areas. Yellow rectangles indicate the existence of buildings passed through by the green seamlines (the conventional method).

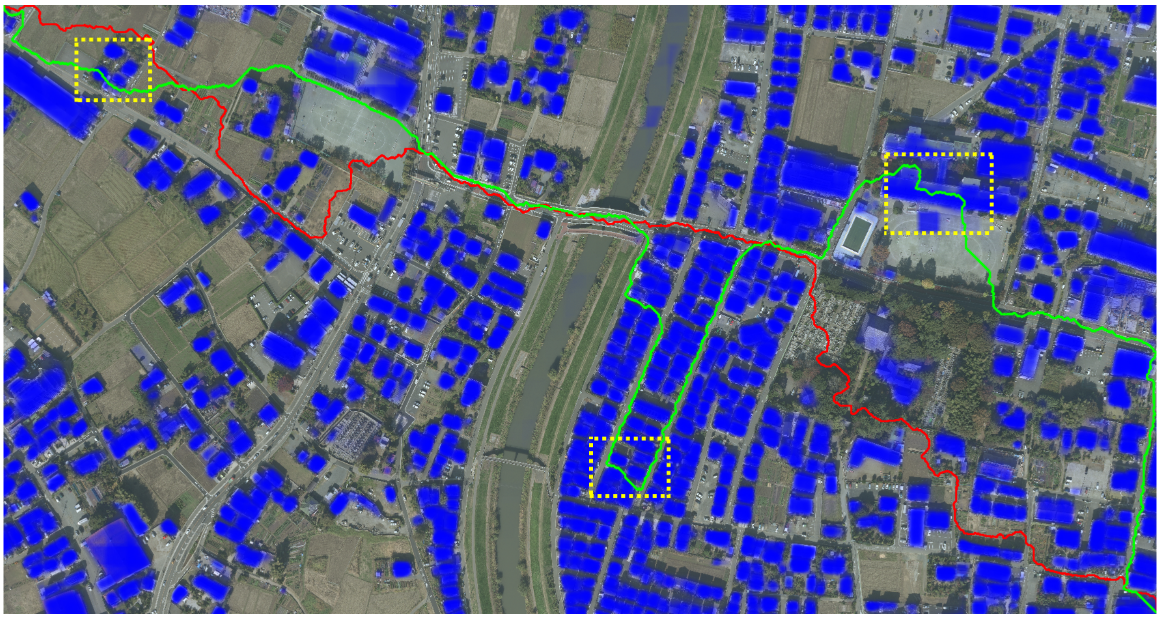

(b) This is another example result. Red and green lines and yellow rectangles have the same meaning as in Figure 6-7a. In the right-most yellow rectangle, the green line drawn by the conventional method passes through a large building even though the ground beside the building is larger.

Figure 6-7: Resulting seamline examples

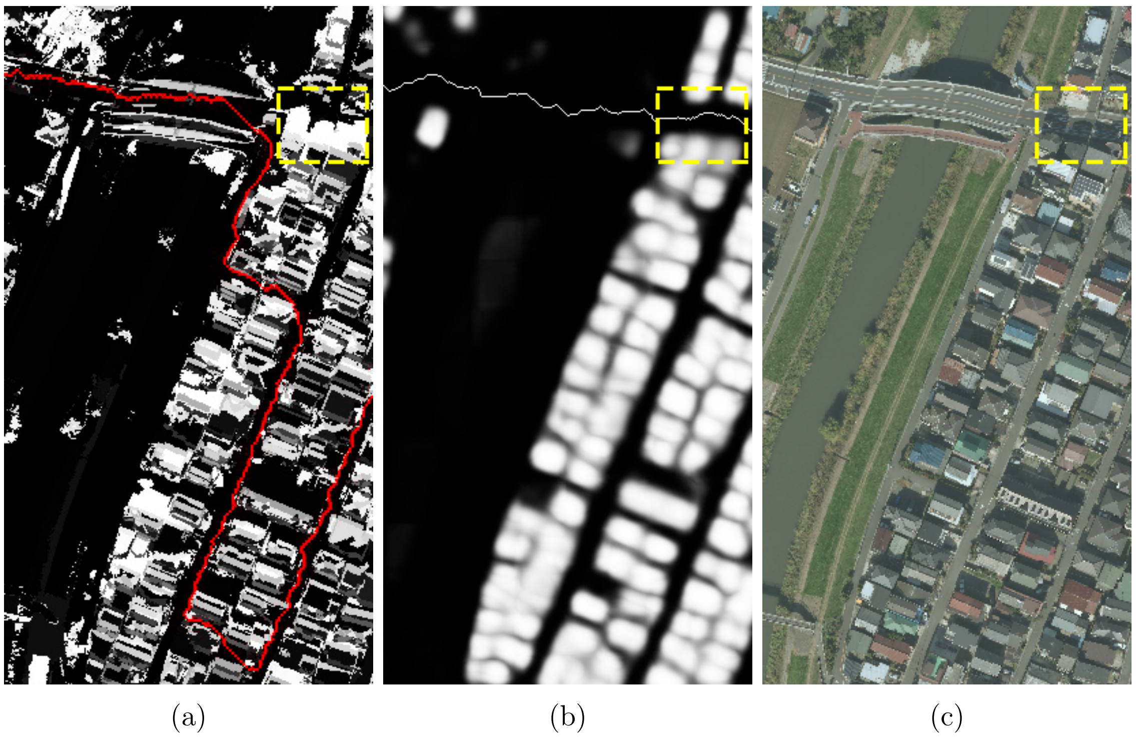


Figure 6-8: The difference of resulting seamlines around the shadow region. The yellow dotted rectangle shows the region that has building shadows. (a) The result of the conventional method [Pan et al., 2014]. (b) The result of our method. (c) The input aerial image.

field on the right side of this figure (enclosed by the rightmost yellow rectangle), but the green line crosses through the building. This is another example caused by the fundamental flaw of the preferred region-based approach.

On the other hand, if an accurate stereo-matching method is available to synthesize disparity maps from a couple of overlapping regions, the resulting disparity can be used to construct a valid cost map to determine seamlines with avoiding buildings. However, once either of a pair of overlapping regions is older than the other, a building appears in one side could not exist in the other side. In such situation, the stereo-matching should be failed, while our method can successfully find the building because of the use of semantic segmentation.

# Chapter 7

# Conclusion

Automatic extraction of ground objects from large aerial imagery is highly demanded in remote sensing field and has significant impact on various applications. Although there have been many attempts at automating this task, simultaneous extraction of multiple objects using only a single model considering computational efficiency but with high accuracy has not been achieved so far. Therefore, in this thesis, we proposed a novel technique to train a single convolutional neural network (CNN) successfully for multiple object extraction from aerial imagery with high accuracy under the effect of regularization derived from the proposed technique.

To learn the parameters of a CNN for semantic segmentation, we should formulate the problem as a pixel labeling task. Then, one of the conventional ways to formulate the task is patch-based approach. It divides a large aerial image into many small patches, and trains a CNN to learn a mapping from raw pixel values to a semantic label image in patch-by-patch manner. This formulation can successfully model the relationship between aerial imagery and the corresponding label map images. However, there have been some problems along with the convenience to define a differentiable objective function to be minimized through optimization of a CNN.

To use the output values of a CNN as a probability vector for such formulation and utilize the exclusiveness between different classes such as road and building for accurate prediction, considering *background* class is inevitable. However, background class is essentially not based on a single object class, because it should include many kinds of objects of no interest. Therefore, the representation for the background class should be more complicated than the other simple object classes. For example, if we consider roads and buildings as classes of interest, the background class could include trees, cars, ocean, rivers, and grass, etc.

Then, in Chapter 3, we proposed a new way to suppress the effect of the background class during training of a CNN. This is the *channel-wise inhibited softmax (CIS)*, and we showed the effectivity for this semantic segmentation task for aerial imagery through some empirical evaluation using large-scale datasets as shown in Chapter 4, and analysis of the middle layer outputs of the trained CNN with CIS as shown in Chapter 5. Furthermore, we proposed an another way to yield non-patchy results without any post-processing networks which need further training processes. That was the *model averaging with spatial displacement (MA)* which performs model averaging explicitly and yield smooth prediction results at the same time. We showed that these two techniques, CIS and MA, can perform more accurate semantic segmentation compared to the state-of-the-art approaches [Mnih and Hinton, 2012, Mnih, 2013] for automatic multiple object extraction in Chapter 4. Those conventional methods also can predict multiple objects, but with independent models for each object of interest. Therefore, our framework has an advantage of compactness.

In Chapter 5, we investigated what actually the CIS performs on a CNN during training by visualizing the resulting middle layer outputs and inverting the resulting feature maps into an image. Then we found that the CIS can suppress learning representations to predict the background class. Furthermore, through a experiment on the difference of amount of data, we found that the improvements derived from the CIS is larger as the size of the dataset is smaller. It might have a significant effectivity on some practical applications. Then, the CIS could be applied to different tasks such as object recognition and object detection, when a background class should be considered as one of classes of interest.

The architecture that used in this thesis is basically fixed on the one shown in Figure 3-3, but there is much room for exploring. For example, one of the simplest way to improve the performance is make the architecture of a CNN deeper. Indeed, we tested the VGG-16 architecture [Simonyan and Zisserman, 2014] on the same dataset, and obtained the best results compared to all the other experimental results shown in this thesis. However, the effectiveness of the CIS is the main focus of this thesis, so that we did not perform further experiments for finding better architecture. As discussed in Chapter 4, the fully-convolutional network (FCN) [Long et al., 2015] is promising also for the semantic segmentation for aerial imagery. It considers three levels of different scales of receptive fields by combining information from feature maps in different depths in the network. It seems to be advantageous in structured prediction. The FCN also has a great advantage in the number of parameters by removing fully-connected layers. It causes not only the compactness of the space complexity but also contributes the speed in the inference stage. While the CIS has more general advantage in any problems which should consider *background* class, the architecture which is adjusted to a specific task could achieve significant improvement.

Some approaches in semi-supervised learning [Kingma and Ba, 2014, Rasmus et al., 2015, Maaløe et al., 2015, Miyato et al., 2015] might be promising also for the pixel labeling task we addressed. Although preparing the large-scale dataset with accurate annotations of building and road masks is obviously costly, we need a large-scale fully-annotated dataset for each domain of input aerial imagery (e.g. country) in the current approaches. However, while the annotation process is highly costly and time-consuming, incredible amount of raw aerial imagery data without annotations are available. Therefore, if we can create a good predictor of semantic labels for each different area or country by using a vast amount of unlabeled data with a very small amount of label data, it will have a great advantage in many practical applications. Thus, considering how to apply some semi-supervised approaches [Rasmus et al., 2015, Miyato et al., 2015] and the work related with generative models [Kingma and Ba, 2014, Maaløe et al., 2015] should be considered for future work.

We implemented our models and the evaluation system with Caffe [Jia et al., 2014] and Chainer [Tokui et al., 2015], which are deep learning frameworks. All of the codes of our proposed methods, the experiments, and the new dataset *Mass. BR* are publicly available [1] [2].

In Chapter 6, we showed an application in aerial image processing to show the practical impact of semantic segmentation for aerial imagery. We proposed a new cost map for determining seamlines based on the semantic segmentation results. It has been believed that extracting ground objects accurately is too difficult [Pan et al., 2014], but we have showed that it can be achieved by using a CNN that has been successfully trained on a large dataset consisting of many aerial images and corresponding object label images. The trained CNN can extract buildings accurately from raw pixel values with no pre-processing. Therefore, there is no need to design hand-crafted features to create cost maps as opposed to the conventional methods. We focused on avoiding the crossing of buildings in this application, and achieved the determination of seamlines that never pass through buildings by applying Dijkstra's algorithm [Dijkstra, 1959] to determine paths on the predicted building probability maps. We showed that if the cost map is designed in a sophisticated manner, then the seamline determination algorithm could be simple. Therefore, the segmentation-based approach for seamline determination is promising as mentioned in Pan et al. [Pan et al., 2014], and the semantic segmentation-based approach is more effective as shown in this thesis.

[1]https://github.com/mitmul/ssai
[2]https://github.com/mitmul/ssai-cnn

# Bibliography

Radhakrishna Achanta, Appu Shaji, Kevin Smith, AurÃľlien Lucchi, Pascal Fua, and Sabine SÃijsstrunk. SLIC Superpixels Compared to State-of-the-art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (11), pages 2274–2282, 2012. doi: 10.1109/TPAMI.2012.120.

Yehuda Afek and Ariel Brand. Mosaicking of orthorectified aerial images. *Photogrammetric Engineering and remote sensing*, 64(2), pages 115–124, 1998.

L Nonboe Andersen, Jan Larsen, Lars Kai Hansen, and Mads Hintz-Madsen. Adaptive regularization of neural classifiers. pages 24–33, 1997.

Ruzena Bajcsy and Mohamad Tavakoli. Computer Recognition of Roads from Satellite Pictures. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(9), pages 623–637, 1976.

Herbert Bay, Tinne Tuytelaars, and Luc Gool. SURF: Speeded Up Robust Features. *The Proceedings of the 9th European Conference on Computer Vision (ECCV)*, pages 404–417, 2006.

Jón Atli Benediktsson, Philip H. Swain, and Okan K. Ersoy. Neural network approaches versus statistical methods in classification of multisource remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4), pages 540–552, 1990.

Horst Bischof, Werner Schneider, and Axel J. Pinz. Multispectral classification of landsat-images using neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 30(3), pages 482–490, 1992.

Julian E. Boggess. Identification of roads in satellite imagery using artificial neural networks: A contextual approach. Technical report, 1993.

Léon Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8), 1991.

Leo Breiman. Random forests. *Machine learning*, 45(1), pages 5–32, 2001.

Christopher M Christoudias, Bogdan Georgescu, and Peter Meer. Synergism in low level vision. *The Proceedings of 16th International Conference on Pattern Recognition (ICPR)*, volume 4, pages 150–155, 2002.

Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. *International conference on artificial intelligence and statistics*, pages 215–223, 2011.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. *The Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

Scott Evan Decatur. Application of neural networks to terrain classification. *The Proceedings of International Joint Conference on Neural Networks*, volume 1, pages 283–288, 1989.

Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1), pages 269–271, 1959.

Piotr Dollar, Zhuowen Tu, and Serge Belongie. Supervised Learning of Edges and Object Boundaries. *The Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1964–1971, 2006.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12, pages 2121–2159, 2011.

Baltsavias Emmanuel. Object extraction and revision by image analysis using existing geodata and knowledge: Current status and steps towards operational systems. *ISPRS Journal of photogrammetry and remote sensing*, 58(3–4), pages 129–151, 2004.

Clement Farabet, Camille Couprie, Laurent Najman, and Yann Lecun. Scene parsing with multiscale feature learning, purity trees, and optimal covers. *The Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 575–582, New York, NY, USA, 2012.

Clément Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), pages 1915–1929, 2013.

Elena Fernandez, Robert Garfinkel, and Roman Arbiol. Mosaicking of aerial photographic maps via seams defined by bottleneck shortest paths. *Operations Research*, 46(3), pages 293–304, 1998.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1), pages 119–139, 1997.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4), pages 193–202, 1980.

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *The Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1319–1327, 2013.

Robert M. Haralick. Automatic remote sensor image processing. *Digital Picture Analysis*, pages 5–63, 1976.

Robert M. Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-3(6), pages 610–621, 1973.

Chris Harris and Mike Stephens. A combined corner and edge detector. *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *The Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015b.

Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. *The Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 695–702, 2004.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786), pages 504–507, 2006.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160 (1), pages 106, 1962.

Jerome M. Idelsohn. A learning system for terrain recognition. *Pattern Recognition*, 2(4), pages 293–301, 1970.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *The Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision (IJCV)*, 1(4), pages 321–331, 1988.

Martin Kerschner. Seamline detection in colour orthoimage mosaicking by use of twin snakes. *ISPRS journal of photogrammetry and remote sensing*, 56(1), pages 53–64, 2001.

Robert Lawrence Kettig and David A. Landgrebe. Classification of multispectral image data by extraction and classification of homogeneous objects. *IEEE Transactions on Geoscience Electronics*, 14, pages 19–26, 1976.

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

Stefan Kluckner and Horst Bischof. Semantic classification by covariance descriptors within a randomized forest. *The Proceedings of IEEE 12th International Conference on Computer Vision Workshop on 3D Representation and Recognition (3dRR)*, pages 665–672, 2009.

Stefan Kluckner, Thomas Mauthner, Peter M. Roth, and Horst Bischof. The proceedings of 9th asian conference on computer vision (accv). pages 477–488, 2009.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

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

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), pages 541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *The Proceedings of the IEEE*, 86(11), pages 2278–2324, 1998.

Jonathan Lee, Ronald C. Weger, Sailes K. Sengupta, and Ronald M. Welch. A neural network approach to cloud classification. *IEEE Transactions on Geoscience and Remote Sensing*, 28(5), pages 846–855, 1990.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *The Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

David G. Lowe. Object recognition from local scale-invariant features. *The Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157, 1999.

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Improving semi-supervised learning with auxiliary deep generative models. *The Proceedings of Advances in Neural Information Processing Systems Workshop on Advances in Approximate Bayesian Inference (NIPS Workshop)*, 2015.

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. *The Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196, 2015.

Helmut Mayer. Object extraction in photogrammetric computer vision. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(2), pages 213–222, 2008.

David M. McKeown, Wilson A. Harvey, and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(5), pages 570–585, 1985.

David L Milgram. Computer methods for creating photomosaics. *IEEE Transactions on Computers*, (11), pages 1113–1119, 1975.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing by virtual adversarial examples. *arXiv preprint arXiv:1507.00677*, 2015.

Volodymyr Mnih. *Machine Learning for Aerial Image Labeling.* PhD thesis, University of Toronto, 2013.

Volodymyr Mnih and Geoffrey Hinton. Learning to Detect Roads in High-Resolution Aerial Images. *The Proceedings of the 11th European Conference on Computer Vision (ECCV)*, 2010.

Volodymyr Mnih and Geoffrey Hinton. Learning to Label Aerial Images from Noisy Data. *The Proceedings of the 29th Annual International Conference on Machine Learning (ICML)*, pages 567–574, 2012.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. *The Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 807–814, 2010.

Nobuyuki Otsu. A threshold selection method from gray-level histograms. 11(285-296), pages 23–27, 1975.

Mete Ozay and Fatos T Yarman Vural. A new fuzzy stacked generalization technique and analysis of its performance. *arXiv preprint arXiv:1204.0171*, 2012.

Jun Pan, Qinghua Zhou, and Mi Wang. Seamline determination based on segmentation for urban image mosaicking. *IEEE Geoscience and Remote Sensing Letters*, 11(8), pages 1335–1339, 2014.

Justin D. Paola and Robert Schowengerdt. A detailed comparison of backpropagation neural network and maximum-likelihood classifiers for urban land use classification. *IEEE Transactions on Geoscience and Remote Sensing*, 33(4), pages 981–996, 1995.

Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. *The Proceedings of the British Machine Vision Conference (BMVC)*, 1(3), pages 6, 2015.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2 (11), pages 559–572, 1901.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. *The Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 3532–3540, 2015.

Brian D. Ripley. *Pattern recognition and neural networks*, page 150. 1996.

Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1), pages 105–119, 2010.

Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3), pages 309–314, 2004.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 1988.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3), pages 211–252, 2015.

Shunta Saito and Yoshimitsu Aoki. Building and road detection from large aerial imagery. *The Proceedings of SPIE Electronic Imaging, Image Processing: Machine Vision Applications VIII*, volume 9405, pages 94050K–1–94050K–12, 2015.

Shunta Saito, Ryota Arai, and Yoshimitsu Aoki. Seamline determination based on semantic segmentation for aerial image mosaicking. *IEEE Access*, 3, pages 2847–2856, 2015.

Shunta Saito, Takayoshi Yamashita, and Yoshimitsu Aoki. Multiple object extraction from aerial imagery with convolutional neural networks. *Journal of Imaging Science and Technology*, 60(1), pages 10402–1–10402–9, 2016.

Caglar Senaras, Mete Ozay, and Fatos T Yarman Vural. Building detection with decision fusion. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3), pages 1295–1304, 2013.

Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic texton forests for image categorization and segmentation. *The Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Beril Sirmacek and Cem Ünsalan. A probabilistic framework to detect buildings in aerial and satellite images. *IEEE Transactions on Geoscience and Remote Sensing*, 49(1), pages 211–221, 2011.

Paul Smolensky. Information Processing in Dynamical Systems: Foundations of Harmony Theory, 1986.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *The Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. Chainer: a next-generation open source framework for deep learning. *The Proceedings of Neural Information Processing Systems Workshop on Machine Learning Systems (NIPS Workshop)*, 2015.

Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *The Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1653–1660, 2014.

Compton J Tucker. Red and photographic infrared linear combinations for monitoring vegetation. *Remote sensing of Environment*, 8(2), pages 127–150, 1979.

Cem Ünsalan. Gradient-magnitude-based support regions in structural land use classification. *IEEE Geoscience and Remote Sensing Letters*, 3(4), pages 546–550, 2006.

Vladimir Vapnik. Pattern recognition using generalized portrait method. *Automation and remote control*, 24, pages 774–780, 1963.

Martin Vetterli and Jelena Kovačević. Wavelets and subband coding. 87, 1995.

Youchuan Wan, Dongliang Wang, Jianhua Xiao, Xiang Wang, Yongsheng Yu, and Jingzhong Xu. Tracking of vector roads for the determination of seams in aerial image mosaics. *IEEE Geoscience and Remote Sensing Letters*, 9(3), pages 328–332, 2012.

Dongliang Wang, Youchuan Wan, Jianhua Xiao, Xudong Lai, Wenli Huang, and Jingzhong Xu. Aerial image mosaicking with the aid of vector roads. *Photogrammetric Engineering & Remote Sensing*, 78(11), pages 1141–1150, 2012.

Thomas P. Weldon, William E. Higgins, and Dennis F. Dunn. Efficient Gabor filter design for texture segmentation. *Pattern Recognition*, 29(12), pages 2005–2015, 1996.

Christian Wiedemann, Christian Heipke, Helmut Mayer, and Olivier Jamet. Empirical Evaluation Of Automatically Extracted Road Axes. *Empirical Evaluation Techniques in Computer Vision*, pages 172–187, 1998.

Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *The Proceedings of the European Conference on Computer Vision (ECCV)*, pages 818–833. 2014.

Xiang Zhang and Yann LeCun. Universum prescription: Regularization using unlabeled data. *arXiv preprint arXiv:1511.03719*, 2015.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. *The Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015.