2020-01-01

Deep Learning For Overhead Imagery: Algorithms And Applications

Anthony Manuel Ortiz Cepeda
University of Texas at El Paso

DEEP LEARNING FOR OVERHEAD IMAGERY:
ALGORITHMS AND APPLICATIONS

ANTHONY ORTIZ

Doctoral Program in Computer Science

APPROVED:

Olac Fuentes, Ph.D., Chair

Christopher Kiekintveld, Ph.D., Co-Chair

Miguel Velez-Reyes, Ph.D.

Nebojsa Jojic, Ph.D.

Yoshua Bengio, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

DEEP LEARNING FOR OVERHEAD IMAGERY: ALGORITHMS AND APPLICATIONS

by

ANTHONY ORTIZ

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

THE UNIVERSITY OF TEXAS AT EL PASO

May 2020

Acknowledgements

I would like to express my deep gratitude to Dr. Olac Fuentes for believing in me and providing me his advice and constant support. Dr. Fuentes was available every single time I needed his guidance since day one. I should also thank Dr. Christopher Kiekinkveld for all his support during this journey. Thank you for giving me the freedom to develop my ideas and wise advice to shape them.

I am very grateful to Luis Francisco for his mentorship throughout my undergraduate studies and to Dr. Miguel Velez-Reyes who was the first person from who I learned about the possibility of pursuing graduate studies at UTEP and for his support since I was back in the Dominican Republic. I am grateful to Dalton Rosario for his great support and mentorship from the early stages of my Ph.D. I thank Prof. Yoshua Bengio for allowing me to join Mila and for serving as a member of my supervisory committee. Those months I spent at Mila shaped my research work. I am also grateful to Nebojsa Jojic and Dan Morris for the opportunity they provided me to work at Microsoft Research and being great mentors.

I thank my friends and labmates at VLL, IASRL, MSR, and Mila for many interesting discussions. I have learned a great deal from working with Kris Sankaran, Caleb Robinson, Kolya Malkin, Blake Elias, Sujan Miah, Moinol Porag, Vincent Michalski, Samira Kahou, Alex Lamb, Alonso Granados, Diego Aguirre, and others. Additionally, I want to thank my family for always believing in me, especially my parents, and my grandmother Elisa who passed away back home. Finally, I must thank my lovely fiancee Gaby for putting up with me during all these years with loving support.

Abstract

Remote sensing using overhead imagery has critical impact to the way we understand our environment and offers crucial information for scene understanding, climate change research, disaster response, urban planning, forest management, and many other applications. At present, deep learning is increasingly used in remote sensing, but mostly borrowing algorithms developed for natural images in the computer vision community. Specific challenges arise while applying deep learning to remote sensing. These challenges include issues related to the high dimensionality and limited labeled data, security and robustness to adversarial attacks, and model generalization. In this thesis we focus on tackling these key challenges.

We present an end-to-end framework to effectively integrate input feature subset selection into the training procedure of a deep neural network for dimensionality reduction. We show that our framework significantly improves performance on multispectral imagery applications. We evaluate quantitatively the robustness of multispectral and hyperspectral image-based deep learning models to adversarial examples. Our experiments show that methods for generating adversarial examples designed for natural images are also effective for remote sensing imagery. We also introduce a framework that integrates dimensionality reduction, adversarial training, and a detector network that greatly improves models' robustness without sacrificing performance.

We then present a novel network architecture which exploits conditional information to improve generalization of deep learning models. Finally, we propose a new normalization layer which facilitates transfer learning and improves performance across a great variety of tasks. Local context normalization is a very efficient generalization of previous ones, it is invariant to batch size, and it is well-suited for transfer learning and interactive systems. This novel normalization layer provides state-of-the-art performance for the tasks of object detection, semantic segmentation, instance segmentation, and aerial image labeling.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Overhead imagery refers to remote sensing imagery acquired by special sensors hosted in a satellite, aircraft, or UAV located above the Earth surface. These remote sensors collect data by detecting the energy that is reflected from Earth. The collected images cover large areas on the Earth's surface, allowing us to see much more than we can standing on the ground. Overhead imagery offers a rich and structured source of information at multiple spatial and spectral resolutions. Overhead images, sensors, and processing systems have applications in agriculture, geosciences, physics, disaster response, climate change studies, environmental studies, and urban planing, among others. The development of new large scale machine learning algorithms designed specifically for overhead imagery applications has great potential to generate positive impact in people's life.

In recent years, deep learning, which is a sub-field of machine learning that uses multiple layers of nonlinear processing units to learn data representations, is transforming almost every industry. During the past few years, deep learning has produced groundbreaking improvements in nearly every field that involves images, audio, or text as the raw data due to both major algorithmic and hardware improvements. Deep learning is extensively pushed by large Internet companies, such as Google, Amazon, Microsoft, and Facebook and it is the first family of methods within machine learning that is truly domain agnostic and does not require feature engineering.

In the wake of this success and thanks to the increased availability of high-quality remote sensing data (e.g. the SpaceNet datasets and Planet labs imagery) and computational resources, the use of deep learning is finally taking off for remote sensing image analysis. In this context, deep learning has been used for some application cases, like land cover/use

classification [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]. However, most of the work has been limited to borrowing algorithms and network architectures developed for other vision tasks to apply them to remote sensing data. Innovative techniques and new learning algorithms designed having the nature of this type of data in mind could have the potential to accelerate progress in the field.

Better leaning algorithms for remote sensing data analysis could have a critical impact in the way we understand our environment and lead to major breakthroughs in global urban planning, climate change adaptation research, mapping, agriculture, disaster response, and many other applications to greatly improve people's lives.

In this thesis we aim to develop new machine learning methods well suited for overhead image analysis. In particular, we focus on tackling the following three challenges that emerge while using machine learning algorithms to solve overhead image-based tasks:

- **High dimensionality issues.** The very high dimensionality of overhead images and the limited size of available labeled datasets limit the exploitation of deep learning algorithms in this context. This is especially true for models trained on hyperspectral and multispectral images. Deep neural networks trained using images with many spectral bands tend to overfit [15]. To overcome high dimensionality issues while still getting full advantage of deep learning technologies, we propose to perform dimensionality reduction jointly with the model training process.

- **Security implications.** We study the security vulnerabilities of overhead image-based machine learning models. Overhead imagery is heavily used for many highly sensitive applications where robustness to adversarial attacks is very important, including applications for mission planning, situational awareness, and target identification systems [16, 17, 18, 19, **?**, 20, 21, 22, 23]. These systems are especially attractive targets for highly skilled and motivated attackers, and the consequences of adversarial examples, which are malicious inputs carefully designed to fool a machine learning system, in these domains could be catastrophic. In Chapter 4 of

this thesis, we evaluate quantitatively the robustness of overhead image-based deep learning models on adversarial settings and develop models more robust to adversarial attacks.

- **Generalization and transferability.** In many cases, overhead imagery is used in remote sensing applications which aim at retrieving information from a huge area (sometimes the whole Earth surface). Solving these tasks successfully requires models that are sufficiently transferable [24]. A major issue with current overhead image-based machine learning models is that models developed for a geographic location do not work well when they are tested in other areas. In Chapters 5 and 6 of this thesis, we work on improving the generalization and transferability of overhead image-based machine learning models. Chapter 5 covers conditional networks as take advantage conditioning information to improve model's generalization. Chapter 6, presents a new normalization approach to improve both performance and transferability of deep learning models.

1.1 Thesis Structure

This thesis is divided into three parts. The first part focus on the high dimensionality issue which arise while learning machine learning models from multispectral and hyperspectral imagery. After surveying related work and providing necessary background in Chapter 2, we focus on methods to overcome overhead imagery high-dimensionality issues while training deep learning models. We present Integrated Learning and Feature Selection as a method to overcome high-dimensionality issues while applying deep learning for multispectral/hyperspectral overhead image analysis in Chapter 3.

The second part centers around the robustness of overhead image-based deep learning models. Chapter 4 shows an analysis of the vulnerability of deep learning remote sensing image-based models to adversarial examples. We also present methods to make those models less vulnerable to white-box attacks and methods to detect adversarial examples in

this domain.

The last part of this thesis focuses on improving generalization and transferability of remote sensing image-based models as well as its application to climate change and land-use monitoring. Chapter 5 covers conditional networks, a family of neural networks which exploit conditioning information to improve model's generalization. In chapter 6, we present Local Context Normalization as a new normalization approach to improve performance and transferability of deep learning models for semantic and instance segmentation, land cover mapping, and iterative segmentation tools. Chapter 7 presents conclusions, discussion and future research directions.

1.2 Thesis Contributions

The thesis includes algorithmic contributions to the fields of machine learning and remote sensing. These contributions are:

1. We propose algorithmic solutions to integrate dimensionality reduction into traditional deep learning approaches to overcome high dimensionality issues in a more suitable way for deep learning models [25].

2. We evaluate quantitatively the robustness of overhead image-based deep learning models on adversarial settings and develop models more robust to adversarial attacks [25, 26].

3. We present scalable algorithms and novel neural network architectures to improve generalization and transferability of overhead image-based deep learning models [27, 28].

Chapter 2

Deep Learning for Overhead Imagery

In this chapter we provide a general overview of previous work solving overhead image-based remote sensing tasks and provide necessary background for following chapters. We focus on approaches that use machine learning (and mostly deep learning) for the broad task of pixel-wise overhead image classification. Other terms like semantic segmentation/labeling, overhead image labeling, and land cover/use mapping refer to similar tasks in nature and would be used indistinctly thorough this thesis.

More formally if X represents an overhead image, overhead image labeling consist of assigning a class label y from the set $y = \{y_1, y_2, ..., y_k\}$ to each pixel $x_{i,j}$, where k represents the total number of classes and i, j represent spatial coordinates. One example of this task is land cover mapping where each pixel $x_{i,j}$ from a high resolution overhead image is required to be mapped to one of k land cover classes as shown in figure 2.1.

By overhead imagery we refer to any type of image acquired by sensors hosted in a satellite, aircraft, or UAV located above the Earth surface. This includes multispectral and hyperspectral images (see figure 2.2) and extends beyond visible light to any region of the electromagnetic spectrum including Near (NIR), Short (SWIR), Medium (MWIR), and Long Wave Infrared (LWIR).

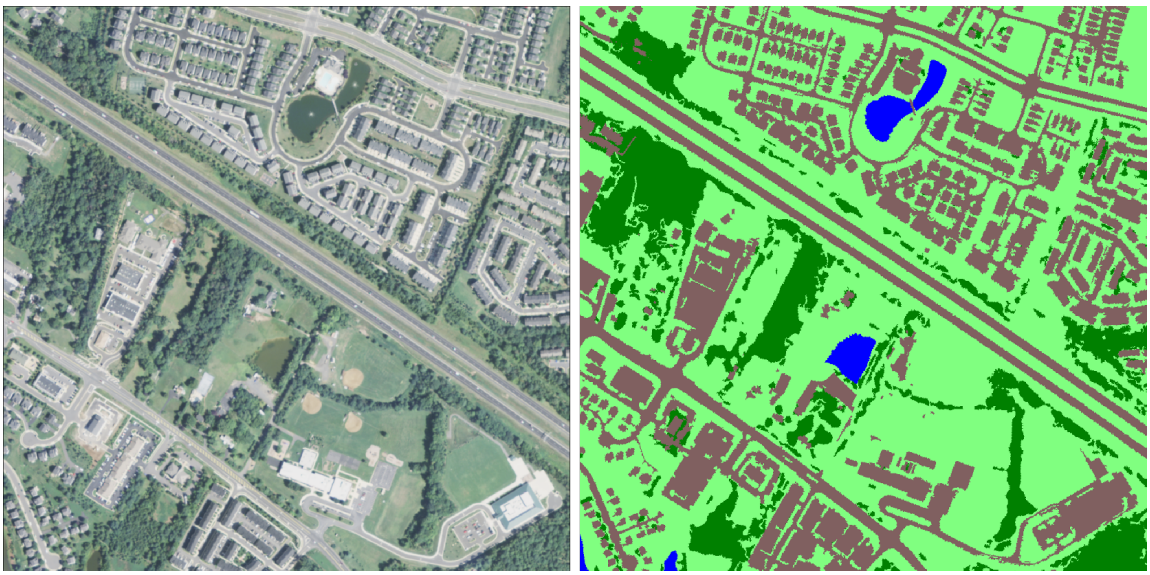


Figure 2.1: Landcover mapping consists of assigning to each pixel $x_{i,j}$ from input image X (left) a land cover class (right). In this example k is four, where blue represents water, dark green forest, light green light vegetation, and brown represents man-made structures.

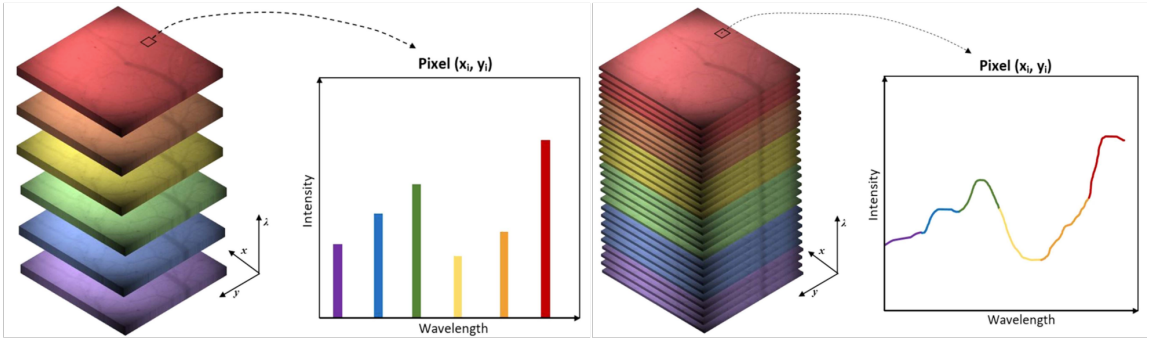


Figure 2.2: Multispectral vs hyperspectral images. Multispectral images are built by several image channels at discrete and somewhat narrow bands while hyperspectral images are built from contiguous ranges of wavelengths and often much more spectral bands [1]. Figure modified from [2].

2.1 Convolutional Neural Networks: Architectures and Key Components

In this section, we provide an overview of Convolutional Neural Networks (CNN) as the main approach used while applying deep learning on overhead imagery.

Convolutional Neural Networks (CNNs) are a class of neural networks which involve the use of convolutions, instead of the matrix multiplications, in several layers, and are commonly used for analyzing imagery. A standard CNN architecture for image analysis tasks is composed of the following key layers types: convolutional layers, nonlinear activations, normalization layers, pooling, and fully connected layers. Figure 2.3 shows an example of a CNN architecture and the configuration of these different layers [1].

2.1.1 CNN Layers

Convolution (conv) layers. For the case of 2D imagery, the convolution operation applied on input features $X(h, x, i)$, where (h, w) indicate the spatial location and i the input channel, is defined in equation 2.1.

$$h^l = X^i * K^{(l,i)} = \sum_m \sum_n \sum_i X(m, n; i) K(h - m, w - n; l, i) \tag{2.1}$$

where $K(l, i)$ is the convolution kernel of size $m \times n$, associated with input channel i and output channel l. In typical setups, multiple kernels are learned for each convolutional layer. The key motivation of using convolutional layers is their ability to provide translation invariance with respect to the detected features. Furthermore, the much smaller size of the kernels compared to the input also promotes parameter sharing and sparsity in terms of connectivity, which also has a positive impact in terms of memory requirements. Two additional parameters in the design of convolutional layers, besides the size of the kernel, are the step size of its application (stride) and the method of handling the boundaries.

[1] Figure modified from `shorturl.at/oEJXY`

Activation functions. Each feature generated by convolving the inputs from the previous layer with a kernel is then passed through a nonlinear function called activation function. The majority of modern approaches use the Rectified linear unit, ReLU, which essentially imposes a non-negatively constraint, $ReLU(X) = max(0, x)$, and other ReLU variants. Other popular activation functions include hyperbolic tangent (tanh), sigmoid, softmax, and linear.

Dilated (atrous) Convolutions. Filters on convolution layers are contiguous. However, it's possible to have filters that have spaces between each cell, called dilation [29]. This can be very useful in some settings to use in conjunction with 0-dilated filters because it allows to merge spatial information across the inputs much more agressively with fewer layers.

Pooling. Pooling layers perform a downsampling operation along the spatial dimensions (width, height). It is common to insert a Pooling layer in-between successive Conv layers in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and to also control overfitting. The pooling layer operates independently on every depth slice of the input and resizes it spatially.

Fully connected. This layer basically takes an input volume (whatever the output is of the conv or ReLU or pool layer preceding it) and outputs an $n\_classes$ dimensional vector where $n\_classes$ is the number of classes that the program has to choose from. It is possible to replace fully connected layers with convolutional layers [30].

Transposed convolution (deconvolution) layers. A requirement of CNN when applied in problems such as land cover mapping is the increase in spatial resolution of a previous layer, which has been tackled through a backward fractional dilated convolution layers [3].

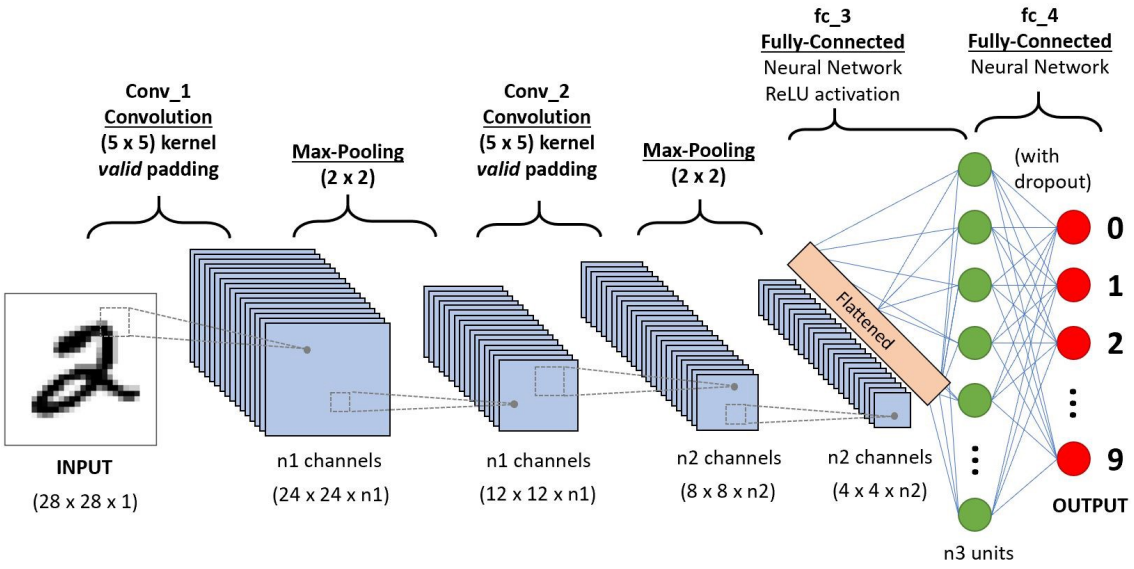


Figure 2.3: Example of a CNN architecture for image classification and how different layers are configure together.

Normalization. Many types of normalization layers have been proposed for use in CNN architectures. We expand on this in Chapters 5 and 6.

2.1.2 Training and Optimization

The process of training a CNN consists on identifying the values for the kernel weights in the convolution and potentially other layers. This involves two major design choices, the selection of the appropriate loss function and the selection of the optimization process. The fundamental algorithm for training deep neural networks is Stochastic Gradient Descend (SGD) [31]; however, variations of SGD such as the Adam [32] often offer similar or better performance and lower computational cost.

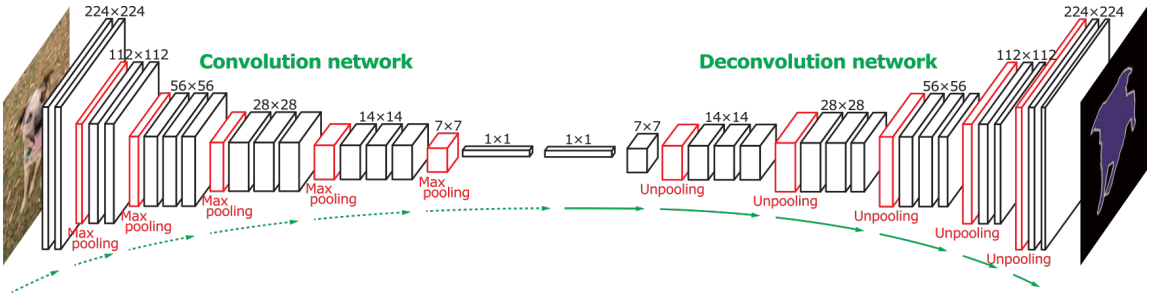


Figure 2.4: Example of a FCN variant for image segmentation using deconvolution layers. Figure from [3].

2.1.3 Popular CNN Architecture used for Overhead Image Labeling

U-Net, Fully Convolutional Networks (FCN) and their most recent variants are the most common CNN architectures for overhead image labeling.

FCN [30]. FCN is built from locally connected layers, such as convolution, pooling and upsampling. No dense layer is used in this kind of architecture which reduces the number of parameters and computation time. Also, the network can work regardless of the original image size, without requiring any fixed number of units at any stage, given that all connections are local. FCN have a downsampling path to capture semantic/contextual information and a upsampling path to recover spatial information where the upsampling path is also learnt. We make use of FCN in chapters 3 and 4. Figure 2.4 shows an FCN extension where transposed convolutions are used in the upsampling [3].

U-Net architectures. U-Net is a very popular CNN architecture for overhead image labeling. It was initially proposed for the problem of semantic segmentation in medical images [33]. In the U-Net architecture, a symmetric design is followed where progressively convolutional and max pooling layers are introduced leading to a very compact representation of the input image (contracting path), which in sequence is expanded back to the dimensions of the input by convolution and upsampling operators. We revisit U-Net in Chapter 5.

2.2 Overhead Imagery Availability and Large Scale Datasets

Recently, very large and ever-growing data volumes of overhead images have been becoming available, often on a global scale.

2.2.1 Datasets

To train deep learning models with good generalization abilities, one needs large datasets. We now review some standard datasets currently available to study overhead imagery.

DSTL Satellite Imagery Dataset [34]. The Defense Science and Technology Laboratory (DSTL) released a dataset of 1km $\times$ 1km satellite images for detection and classification of the types of objects found in these regions at the pixel level. There are two types of spectral imagery content provided in this dataset: 3-band images with RGB natural color and 16-band images containing spectral information captured by wider wavelength channels. This multi-band imagery is taken from the Visible and Near Infrared (VNIR) (400 - 1040 nm) and short-wave infrared (SWIR) (1195 - 2365 nm) range collected using the DigitalGlobe's WorldView-3 satellite system. DSTL labeled 10 different classes:

1. **Buildings**: large building, residential, non-residential, fuel storage facility, fortified building.
2. **Misc**: manmade structures.
3. **Road**
4. **Track**: poor/dirt/cart track, footpath, trail.
5. **Trees**: woodland, hedgerows, groups of trees, standalone trees.
6. **Crops**: contour ploughing/cropland, grain crops (wheat, corn), row crops (potatoes, turnips).
7. **Waterway**
8. **Standing Water**
9. **Large Vehicle**: large vehicle (e.g. lorry, truck, bus), logistics vehicle.

10. **Small Vehicle**: small vehicle (car, van), motorbike.

Inria Aerial Image Labeling Dataset [35]. This dataset was introduced to test generalization of remote-sensing segmentation models [35]. It includes imagery from 10 dissimilar urban areas in North America and Europe. Instead of splitting adjacent portions of the same images into training and test sets, the splitting was done city wise. All tiles of five cities were included in the training set and the remaining ones are used as test set. In this work we refer to the test set as *transfer set*. The transfer set also has variation in illumination, landscape, and time which makes it well suited to evaluate distribution-shift generalization. The provided imagery comes orthorectified [35] and has a spatial resolution of 0.3 m per pixel covering 810 km^2 (405 km^2 for training and 405 km^2 for the transfer set) on an evenly spaced grid. Images were labeled for the semantic classes of *building* and *not building*.

Zeebruges / Data Fusion Contest 2015 dataset [36]. In 2015, the Image Analysis and Data Fusion Technical Committee of the IEEE Geoscience and Remote Sensing Society (GRSS) organized a data processing competition aimed at 5 cm resolution land cover mapping. To do so, the organizers provided both an RGB aerial image and a dense (65 pixels/m^2) lidar point cloud over the harbor of Zeebruges (Belgium). The data are organized on seven $10,000 \times 10,00$ pixels tiles. All the tiles have been labeled densely in eight land classes, including land use (building, roads) and objects (vehicles, boats) [37].

ISPRS 2-D semantic labeling challenge. [38]. The working group II/4 of the ISPRS 3-D Scene Reconstruction and Analysis provided a subdecimeter resolution data set over the two cities of Vaihingen and Potsdam (Germany). The data are similar to those of the Zeebruges data, with the difference that the height information is provided as a Digital Surface Model (DSM) at the same resolution of the image data. Moreover, images are provided with an infrared channel. The data set is also fully labeled into six classes, including land classes (roads, meadows) and objects (cars). It also comes with a clutter

class gathering all unknown objects. The Vaihingen data set comes with 33 tiles having an average size of $2,000 \times 3,000$ pixels. Half the tiles come with labels. The other 17 tiles come with no labels, and participants must upload classification maps for evaluation. The Potsdam data set comes with 24 labeled tiles ($6,000 \times 6,000$ pixels) and 14 unlabeled ones. Both data sets can be obtained from [38].

DeepGlobe [39]. The DeepGlobe dataset was created from DigitalGlobe Vivid+ satellite imagery (2018) containing roads, buildings and landcover labels at resolution of 50 cm/pixel.

Spacenet [40]. Spacenet is a corpus of commercial satellite imagery and labeled training data which consists of building footprints for various cities around the world at resolutions ranging from 30-50 cm/pixel.

Chesapeake Land Cover [12]. This dataset contains high-resolution aerial imagery from the USDA NAIP program [41], high-resolution land cover labels from the Chesapeake Conservancy [42], low-resolution land cover labels from the USGS NLCD 2011 dataset [43], low-resolution multi-spectral imagery from Landsat 8 [44], and high-resolution building footprint masks from Microsoft Bing [45], formatted to felicitate machine learning research into land cover mapping.

Other sources of data includes cost free and for cost images from several satellites. Sentinel satellites have already acquired about 25 PB of data freely available for everyone since 2014. Planet Labs provides for cost daily overhead imagery from almost every place in the world [46]. Noisy labels for a variety of classes can be freely obtained from Open Street Maps (OSM) [47].

2.3 Deep Learning for Remote Sensing

In the remote sensing community deep learning has been used for some application cases like land cover/use classification with some promising results been achieved. In this section we provide a review of previous work for hyperspectral and multispectral image analyis and high resolution overhead images.

2.3.1 Deep Learning for Multispectral and Hyperspectral Image Analysis and Current Challenges

The remote sensing community started using deep neural networks for multispectral and hyperspectral image analysis in 2014 mainly as feature extractors for the tasks of land cover/use classification. In [10] authors used a stacked Auto-encoders (SAEs) to extract hierarchical features in the spectral domain. Subsequently, in [11], the authors employed Deep Belief Networks (DBN) with the same purpose. Similarly, Tao et al. [48] used sparse autoencoders (SAEs) to learn an effective feature representation from the raw data; then, the learned features were used as input to a support vector machine (SVM) for pixel-wise image classification.

Convolutional Neural Networks (CNNs). Makantasis et al. [9] exploited a two-dimensional (2-D) CNN to encode spectral and spatial information, followed by a multilayer perceptron (MLP) performing the actual classification. In [49], the authors attempted to carry out the classification of crop types using 1-D CNN and 2-D CNN. They concluded that the 2-D CNNs can outperform the 1-D CNNs, but some small objects in the final classification map provided by 2-D CNNs are smoothed and misclassified. Following the developments in 3-D CNNs [8], in which the third dimension usually refers to the time axis, such architecture has also been employed in hyperspectral classification. In [50], to allow a CNN to be appropriately trained using limited labeled data, the authors present a novel pixel-pair CNN to significantly augment the number of training samples. In a 3-D CNN, convolution

operations are performed along spatial and spectral dimensions, while in 2-D CNNs, they are done only spatially. The authors in [7] use a supervised 3-D CNN-based model for spatial-spectral classification.

Unsupervised learning. To allow less dependence on the existence of large annotated collections of labeled data, unsupervised feature extraction is of great interest. The authors of [6] proposed an unsupervised convolutional network for learning spectral-spatial features using sparse learning to estimate the network weights in a greedy layer-wise fashion instead of end-to-end learning. Mou et al. [51, 5] presented a network architecture called a fully residual network for unsupervised spectral-spatial feature learning of hyperspectral images.

Curse of dimensionality. Deep neural networks trained with many spectral bands tend to overfit [15]. This is one the main issues preventing deep learning from wide adoption for multispectral and hyperspectral image analysis. Researchers started to look into it using deep learning approaches. In [52], the authors proposed a self-improving CNN model that combines a 2-D CNN with area fractional order Darwinian particle swarm optimization algorithm to iteratively select the most informative bands suitable for training the designed CNN. Santara et al. [53] discussed an end-to-end, band-adaptive spectral-spatial-feature learning network to address this problem. We address this problem in chapter 3 by developing a novel framework that performs band subset selection and overhead image labeling simultaneously and significantly outperforms previous approaches.

2.3.2 Deep Learning for High Spatial Resolution Overhead Images and Current Challenges

Since early 2010, Mnih et al. saw the advantages of using deep neural networks for very high resolution overhead images [54, 55, 56]. They were able to greatly improve performance for the task of road extraction/segmentation by using deep convolutional neural

networks for structured prediction. One of the main issues back then while learning based on overhead imagery was the fact that the labels available at large scale where very noisy. Mnih and Hinton proposed a loss function that is robust to label noise and found that training a classifier on large amounts of this type of noisy data with a robust loss function can potentially produce a much better detector than by using a much smaller set of accurate labels [54, 56].

Thanks to the enormous progress in the development of better neural network architectures for structure prediction [30, 33, 57], and the availability of new datasets [34, 35, 39, 40, 42], we have seen a big improvement in the quality of the predictions. In mid 2015, Robinson et al. where able to create the first land cover mapping of the United States [12] using label super resolution [13]. They point out that deep learning models are unable to generalize across different types of geography or even across seasonality. This is the main issue today while applying machine learning to overhead image analysis instead of label noise. We tackle this problem of generalization in chapter 5 and 6 of this thesis.

Chapter 3

Integrated Learning and Feature Selection

The very high dimensionality of multispectral and hyperspectral images and the limited size of available data sets for training limit the exploitation of this data source. Deep neural networks trained with many spectral bands tend to overfit, which leads to weak generalization capability [15, 58].

A common approach to reduce high dimensionality is to transform the data into a lower dimensional space using methods like Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) [59, 60]. However, these methods always change the meaning of the original data as the features in the low-dimensional space are linear combinations of the original bands. A second approach to dimensionality reduction for hyperspectral and multispectral images is band selection, which selects a subset of the original bands. Band selection techniques can be divided into two categories: supervised and unsupervised [61]. Supervised methods aim to preserve the object information related to a target function, which is known a priori [61, 62, 63, 64], while unsupervised methods do not assume any object information [65, 66, 67]. Most previous band selection methods follow a two-step procedure where an independent method is used initially to select a subset of bands and then a learning algorithm is run using these bands as input.

In this chapter, we propose a new approach for dimensionality reduction that directly integrates supervised feature selection with a deep learning classifier. We call this method Integrated Learning and Feature Selection (ILFS). ILFS allows feature selection to be optimized by the learning algorithm, and because information from bands that are not ul-

timately selected is available during the learning process, it can improve performance. We demonstrate that integrating feature selection into an end-to-end deep learning algorithm greatly improves performance in multispectral image classification.

3.1 Integrated Learning and Feature Selection

We propose ILFS as a framework to automatically select the input features that are most useful for the learning task. Dimensionality reduction is done simultaneously with learning a model to solve the learning task. In the case of our semantic segmentation task, this corresponds to choosing bands that will help a deep neural network better discriminate objects in a multispectral image.

Let us consider I as the input space with n features and let Z be the vector encoding the selected features $\{z_1, z_2, \dots\ z_{k-1}, z_k\}$ (e.g., the bands in a multispectral image). We define X to be the image that results from selecting features Z from I, and J to be the cost function of the network. To discover and select features that can discriminate objects more effectively, we include the input feature selection in the learning process. To achieve this, we compute the gradient of the loss with respect to the selected features using the chain rule:

$$\frac{\partial J}{\partial Z} = \frac{\partial J}{\partial X}\frac{\partial X}{\partial Z} \tag{3.1}$$

3.1.1 Obtaining the Derivatives

To compute $\frac{\partial J}{\partial X}$, it is only necessary to backpropagate the loss from the last layer up to the input image. To obtain $\frac{\partial X}{\partial Z}$, we compute $\frac{\partial X}{\partial z_i}$ for each of the features in Z. While the bands in the high dimensional image are discrete, they are densely sampled, thus they can be viewed as a continuous and differentiable space. We compute the values of fractional bands using simple linear interpolation, which leads to piecewise constant derivatives (see Figure 3.1). Thus, if we define $H(I, z)$ to be the resulting image from selecting feature z

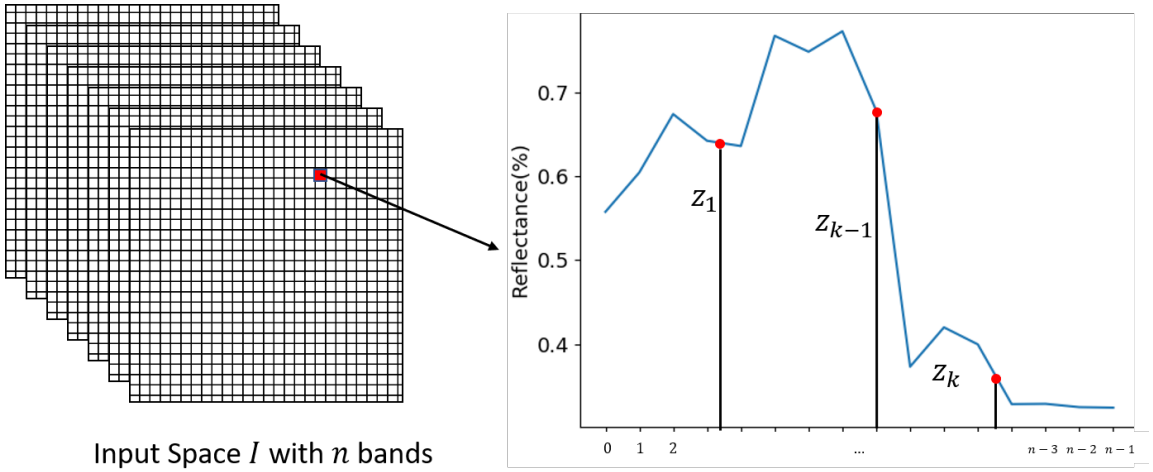


Figure 3.1: $H(I, Z)$ for a pixel by selecting k features

from I, equation 3.2 shows how a change in a particular feature z_i produces a change in X.

$$\frac{\partial X}{\partial z_i} = H(I, \lfloor z_i \rfloor + 1) - H(I, \lfloor z_i \rfloor) \tag{3.2}$$

Finally, we define $\frac{\partial X}{\partial Z}$, as the vector with the values $\left[\frac{\partial X}{\partial z_1}, \frac{\partial X}{\partial z_2}, \ldots \frac{\partial X}{\partial z_{k-1}}, \frac{\partial X}{\partial z_k}\right]$

3.1.2 ILFS Implementation Details

Our implementation of ILFS initially defines Z as a random vector with the bands selected. We use this vector to obtain the image X from the input image I that has all bands. We update the bands 10 times per epoch using gradient descent and an adaptive learning rate. This is necessary to prevent premature convergence in the model. We use an initial learning rate of 0.2 and decrease it exponentially after every epoch using exponential decay. We update X every time Z is updated.

3.2 Experimental Setup

We evaluate our proposed approach using the DSTL Multispectral Satellite Imagery Dataset presented in Chapter 2. ILFS can also be applied without any modification to hyperspectral imagery as well.

3.2.1 Models

In semantic segmentation, we want to assign each pixel in the input image to an object class. Most popular approaches to do semantic segmentation are based on Fully Convolutional Networks (FCN) [30]. FCN are a type of Convolutional Neural Network architecture for dense predictions that do not use any fully connected layers. This allows segmentation maps to be generated for large images. Almost all subsequent state-of-the-art approaches for semantic segmentation have adopted this paradigm [68, 69, 70].

We used Tensorflow to train different models of VGG-19-based FCN-8 for semantic segmentation [30, 71]. We trained models both with and without using ILFS for dimensionality reduction. We trained our deep network on the DSTL Satellite Image Dataset using either RGB, VNIR, SWIR, or VNIR and SWIR channels as input. 10000 randomly selected (without replacement) 224x224 patches were used for training, and 500 224x224 patches were reserved for testing. The models were trained on an NVIDIA Tesla GPU on Amazon Web Services. All the models were trained for the same number of epochs on the training set. A small batch size (4 patches) was necessary to fit the training set in memory.

3.2.2 ILFS Evaluation

We report performance results using mean intersection over union (mean IoU), a standard metric for common semantic segmentation and scene parsing evaluations.

$$meanIoU = (1/n_{cls}) \sum_{i} n_{ii}/(t_i + \sum_{j} n_{ji} - n_{ii}) \tag{3.3}$$

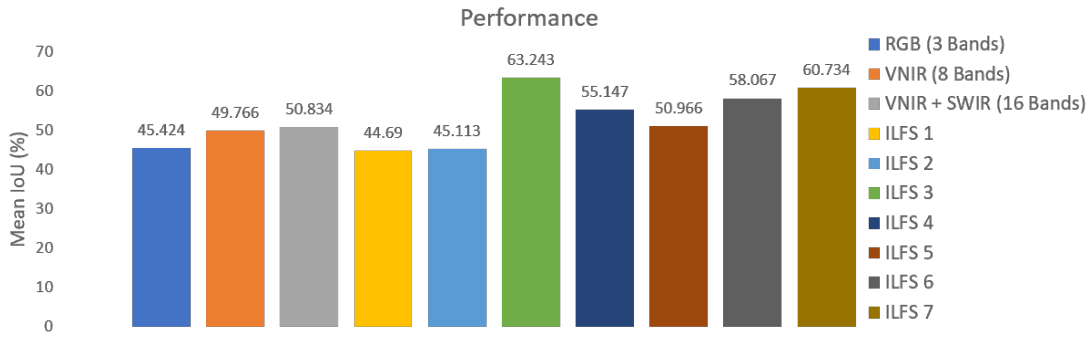


Figure 3.2: Performance Evaluation for Semantic Segmentation on the DSTL dataset. All of the models were trained on the same deep neural network architecture (VGG19-based FCN-8) for the same number of epochs. For RGB we trained using only the three RGB bands as input. For VNIR, we trained with all of the bands from the visible and near-infrared (VNIR) region of the spectrum. For VNIR + SWIR all of the available bands were used as input while training. ILFS 1, 2, ..., 7 show the results when ILFS is used to pick 1, 2, 3, 4, 5, 6, and 7 input features, respectively.

where n_{ij} is the number of pixels of class i predicted to belong to class j, there are n_{cls} different classes, and $t_i = \sum_j n_{ji}$ is the total number of pixels of class i.

3.3 Results

3.3.1 ILFS Significantly Improves Performance

Figure 3.2 presents the results of training models for semantic segmentation using different inputs, including ILFS used to select different numbers of input features. Performance is measured using the mean IoU. All of the models used the same network architecture (VGG19-based FCN-8) and the same number of training epochs. The results show that ILFS improves performance by up to around 25% over the best model without ILFS. All ILFS models with more than 2 features beat all of the non-ILFS models. Hyper-parameters were fine-tuned for ILFS 3 and then used for the other experiments. Improvement in

performance could be obtained by fine-tuning hyper-parameters for every model. The intuition for the improved accuracy is that ILFS allows the network to find a combination of bands (including interpolated bands) that allow for better discrimination of the objects in the scene while the smaller feature space prevents overfitting of the training data.

Another advantage of ILFS with 3 features is the feasibility of doing transfer learning. Training a deep neural network from scratch may not be feasible for various reasons: a dataset of sufficient size may not be available, or reaching convergence can take too long, or the memory requirement may be too high for the available hardware. Even if training a network is feasible, it is often helpful to start with pre-trained weights instead of randomly initialized weights. Here we can use a model previously trained on RGB for a similar task to initialize the weights and selected features and then continue the training using ILFS 3. Yosinski et al. showed that transferring features even from distant tasks can be better than using random initialization, taking into account that the transferability of features decreases as the difference between the pre-trained task and the target one increases [72].

3.3.2 ILFS Visualization

For ILFS 3 it is possible to display a false color image of the selected bands. This allows the user to visualize what the network has learned and determine if it is a good set of discriminative channels or bands. Figure 3.3 shows the true color image of one portion of the data cube and the corresponding false-color image of the features obtained from ILFS 3. This output is very similar to what is known as spectral indices in remote sensing. We can visually discriminate classes (e.g., vegetation is bright white), which may be useful for human analysts as well.

3.4 Chapter Summary and Conclusions

We introduced Integrated Learning and Feature Selection (ILFS) as a framework for dimensionality reduction of high dimensional imagery through supervised feature subset se-

(a) RGB Visualization (b) False Color ILFS 3 Visualization

Figure 3.3: ILFS 3 Visualization. We see similar results to what is known as spectral indices in the remote sensing community, and can visually discriminate classes.

lection using gradient descent on the input space. ILFS not only reduces data dimensionality but also improves performance on deep neural networks for high dimensional imagery applications. While we demonstrated ILFS here for band selection, this technique could be adapted to any learning task where the bands or channels of the input image are densely sampled. We show results on multispectral pixel-wise image classification, however the propose approach should be applicable to hyperspectral images as well.

In the following chapter we will start exploring the robustness of this machine learning models on adversarial settings.

Chapter 4

Building Robust Overhead Image-based Models

It has been shown that machine learning models based on RGB images are often vulnerable to adversarial examples [73, 74, 75, 76, 77]. Adversarial examples are inputs maliciously constructed to induce errors by machine learning models at test time. This represents a new attack vector against systems that rely on machine learning models for critical functions (e.g., facial recognition to establish identity, among many others). Researchers are working towards developing defenses against attacks on RGB-based classifiers and have proposed numerous strategies to train models that are more robust to adversarial examples or to detect adversarial examples [78, 79, 80, 81, 82]. However, they have frequently been found to be vulnerable to other types of attacks, or come at the cost of decreased performance on clean inputs [83, 84, 85, 86, 87, 82].

Many highly sensitive applications of remote sensing and image analysis rely on more sophisticated imaging technologies that go beyond RGB. Such applications include screening systems in airport security, military applications of satellite imagery for mission planning, situational awareness, surveillance, night vision systems, thermal sensors, and target identification systems [16, 17, 18, 19, 20, 21, 22, 88]. These systems are especially attractive targets for highly skilled and motivated attackers, and the consequences of adversarial attacks on learning algorithms in these domains could be catastrophic. However, we are not aware of any previous work that focuses on adversarial examples for machine learning with non-RGB images.

We present the first rigorous study of the robustness of non-RGB image-based sys-

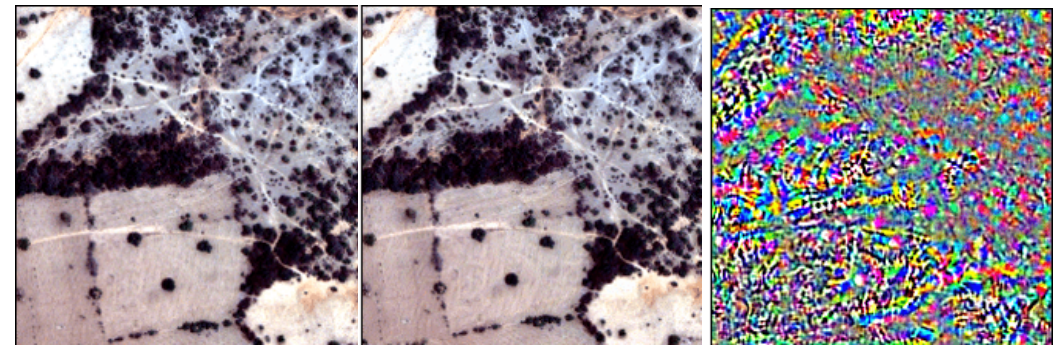
tems (VNIR, SWIR, Panchromatic) against adversarial examples for the task of semantic segmentation. We show that it is not only feasible to generate deceptive examples for machine learning models based on non-RGB images, it is easier in a sense than attacking models for RGB images. However, we show that performing band selection using ILFS can substantially improve the robustness of machine learning models against these adversarial examples. ILFS limits the attack surface by reducing the number of features that can be modified by an attacker to induce errors, and does so in an unpredictable way. ILFS also forces the model to perform well when there is uncertainty in the input space because the input changes throughout the learning process.

We also introduce a detection network that uses domain knowledge information to effectively detect adversarial examples. Finally, we propose a framework that integrates input subset feature selection, adversarial training, and a detector network to drastically improve the robustness of the models without sacrificing performance.

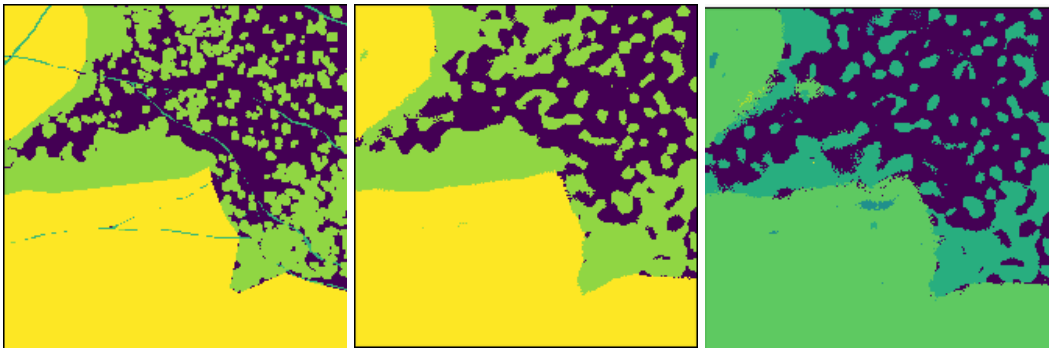
4.1 Adversarial Examples Beyond RGB

The objective of adversarial learning is to find a perturbation ξ that when added to an input X changes the output of the model in a desired way. The attacker tries to keep ξ small enough such that when it is added to X to produce $\mathrm{X}^{\mathrm{Adv}}$ = X + ξ the difference between $\mathrm{X}^{\mathrm{Adv}}$ and X is almost imperceptible.

We denote by the function f_θ a deep neural network with parameters θ. $f_\theta(X)$ is the output of f_θ, and y^{true} is the corresponding ground-truth label. In this work, X is an image, $f_\theta(X)$ is the conditional probability $p(y|X;\theta)$ encoded as a class probability vector, and y^{true} is a one-hot encoding representation of the class. $J(f_\theta(X), y^{true})$ is the classification loss function. We assume that J is differentiable with respect to θ and with respect to X.

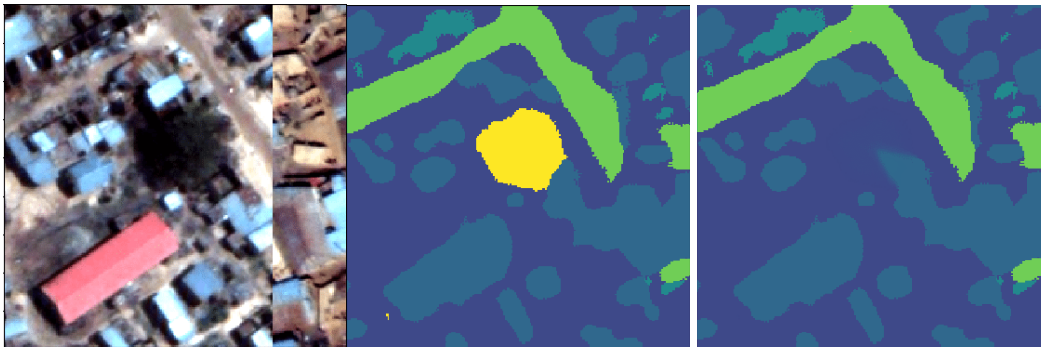


(a) Clean Input (b) Adv. Example (c) Amplified Perturbation



(d) Ground-Truth (e) Pred. Clean (f) Pred. Adversarial

Figure 4.1: An adversarial example generated with l_∞-norm of 4. (a) RGB of the original image (b) RGB representation of the adversarial examples obtained using the Iterative FGSM ll method (c) the amplified noise added to the original image (d) the ground-truth image (e) the model prediction when the original image is the input (f) the model prediction when the adversarial example is the input.



(a) True Color Input (b) Pred. Clean (c) Pred. Adversarial

Figure 4.2: Dynamic Adversarial Perturbation for HSI Semantic Segmentation attack.

4.1.1 Methods to Generate Adversarial Examples

We tested the following attack methods:

Fast Gradient Sign Method (FGSM): Goodfellow et al. [73] proposed a fast single-step method for computing untargeted adversarial perturbations. This method defines an adversarial perturbation as the direction in image space that yields the greatest increase in the linearized cost function under L_∞ norm with the perturbation bounded by the parameter ϵ. This can be achieved by performing one step in the gradient sign's direction with step-width ϵ:

$$X^{\text{Adv}} = X + \epsilon \, sign(\Delta_x J(f_\theta(X), y^{true})) \tag{4.1}$$

This method is simple and computationally efficient compared to more complex methods but it usually has a lower success rate [74].

One-step Target Class Method (FGSM ll): Kurakin et al. [89] proposed an alternative approach to FGSM that maximizes the conditional probability $p(y^{target}|X)$ of a specific target class y^{target} which is unlikely to be the real class for the input image X.

$$X^{\text{Adv}} = X - \epsilon \, sign(\Delta_x J(f_\theta(X), y^{target})) \tag{4.2}$$

As proposed in [89], we choose the least likely class predicted by the model as the

target class y^{target}.

Basic Iterative Method (Iterative FGSM): [74, 82] This is an extension of FGSM in which FGSM is applied multiple times with a small step size, , and clip pixel values of intermediate results after each step to ensure that they are in an ϵ-neighbourhood of the original input:

$$X_0^{Adv} = X,$$

$$X_{i+1}^{Adv} = Clip_X, \epsilon \{X_i^{Adv} + \alpha \, sign(\Delta_x J(f_\theta(X_i^{Adv}), y^{true}))\} \tag{4.3}$$

This increases the chance of fooling the original network. In this work, as in [89], we used $\alpha = 1$, which means that we changed the value of each pixel by 1 on each step. We set the number of iterations to be $min(\epsilon + 4, 1.25 * \epsilon)$.

$Clip_X, \epsilon(A)$ refers to the element-wise clipping of A, with $A_{i,j}$ clipped to the range $[X_{i,j} - \epsilon, X_{i,j} + \epsilon]$. This guarantees that the max l_∞-norm of the perturbation is never greater than ϵ.

Iterative Least-Likely Class (Iterative FGSM ll) [74] is a stronger version of FGSM ll. In this case the target class is set to be the least-likely class (y^{ll}) predicted by the network to fool:

$$X_0^{Adv} = X,$$

$$X_{i+1}^{Adv} = Clip_X, \epsilon\{X_i^{Adv} - \alpha \, sign(\Delta_x J(f_\theta(X_i^{Adv}), y^{ll}))\} \tag{4.4}$$

we used $\alpha = 1$ and the number of iterations was set to $min(\epsilon + 4, 1.25 * \epsilon)$.

These attacks were all originally proposed in the context of RGB image classification, but they have been adapted to semantic segmentation [90, 91, 92, 93], object detection [93], and other tasks.

Dynamic Adversarial Perturbations for Semantic Segmentation: For semantic segmentation, the loss function is a sum over the spatial dimensions of the ground-truth.

$$J_s(f_\theta(X), y) = 1/nmPix \sum_{(i,j)\in X} J_{cls}(f_\theta(X)_{ij}, y_{ij}) \tag{4.5}$$

Metzen et al. [92] describes an adversarial example for semantic segmentation as an input x_{adv} for f_θ such that $J_s(f_\theta(X), y^{tgt})$ is minimal without making perceptible changes to the input. In the context of multispectral and hyperspectral images in addition to keeping the spatial information almost identical to the input, the spectral signature of every pixel should be preserved. Otherwise, experts could identify the perturbations by just looking at the spectral information. Real world scenarios in remote sensing may consist of an adversary trying to hide certain kind of object. We assume that the adversary has access to the model f_θ and can use $y^{pred} = f_\theta(X)$ as an initial step, and he would like to keep y^{tgt} as similar as possible to y^{pred} to avoid attracting the attention of humans monitoring the system. To accomplish this Metzen et al. proposed assigning to the target class the predicted output for all the pixels in the background (X_{bg}) (the ones you are not looking to hide) and filling the gaps of the objects trying to hide (X_o) by interpolating pixels in the background using a nearest-neighbor heuristic. We follow the same idea for y^{tgt} in this work.

Given y^{tgt}, adversarial examples to hide objects while making the spatial and spectral changes imperceptible can be obtained using the following formulation:

$$\begin{aligned} J_s(f_\theta(X), y) = 1/nmPix\{w \sum_{(i,j)\in X_o} J_{cls}(f_\theta(X)_{ij}, y_{ij}^{tgt}) \\ +(1-w) \sum_{(i,j)\in X_{bg}} J_{cls}(f_\theta(X)_{ij}, y_{ij}^{tgt}\} \end{aligned} \tag{4.6}$$

$$\begin{aligned} X_0^{Adv} = X, \\ X_{i+1}^{Adv} = Clip_X, \epsilon \{X_i^{Adv} - \alpha\, sign(\Delta_x Js(f_\theta(X_i^{Adv}), y^{tgt}))\} \end{aligned} \tag{4.7}$$

4.1.2 Attacks Used

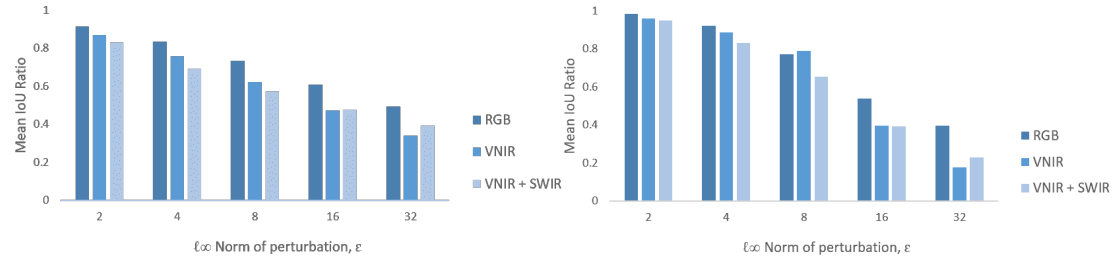
We used the FGSM, FGSM ll, Iterative FGSM, Iterative FGSM ll, and Dynamic Adversarial perturbations for Semantic Segmentation attacks. The attacks were generated with l_∞ norms of 2, 4, 8, 16, and 32, which corresponds to allowing increasingly more perceptible changes to the original image.

4.1.3 Robustness Evaluation

The mean Intersection over Union (mean IoU) is the primary metric used for evaluating semantic segmentation. However, as the accuracy of each model varies, we adopt the relative metric used in [94] and measure adversarial robustness using the mean IoU Ratio. The mean IoU Ratio is the ratio of the network's IoU on adversarial examples to that for clean images computed over the entire dataset. A higher mean IoU Ratio implies more robustness.

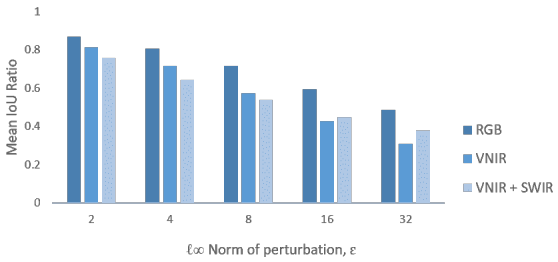
4.1.4 Non-RGB Image-Based Models are Vulnerable to Adversarial Examples

Figure 4.1 shows an example of an attack on a multispectral model using the Iterative FGSM ll with an l_∞-norm of perturbation of 4. To visualize the results we show the true color composition for the multispectral clean and adversarial input. The difference between the clean and the adversarial input is visually imperceptible, but the predictions of the model are totally different. FGSM, Iterative FGSM, FGSM ll, and Iterative FGSM ll attacks try to cause the model to make as many mistakes as possible. The main issue with these attacks is that in real life scenarios totally disassociated with the real classes will make the attacks obvious. Attackers will likely prefer attacks like the one shown in Figure 4.2. This attack can be used to hide specific classes and/or objects from the scene while giving as output a prediction that is as close as possible to the real prediction. Figure 4.2 shows how we successfully hid a tree for the adversarial prediction.

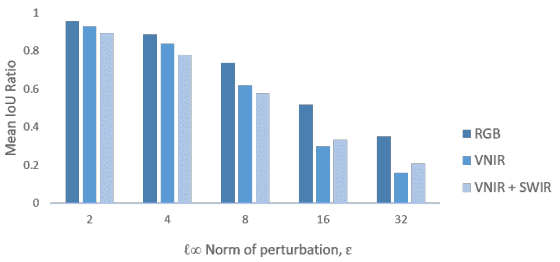


(a) FGSM Attack

(b) FGSM ll Attack



(c) Iterative FGSM Attack



(d) Iterative FGSM ll Attack

Figure 4.3: Robustness of multispectral image-based models to adversarial examples. We observe that models trained on high dimensional images are even more vulnerable to adversarial examples than RGB image-based models for white box settings for all four tested attacks.

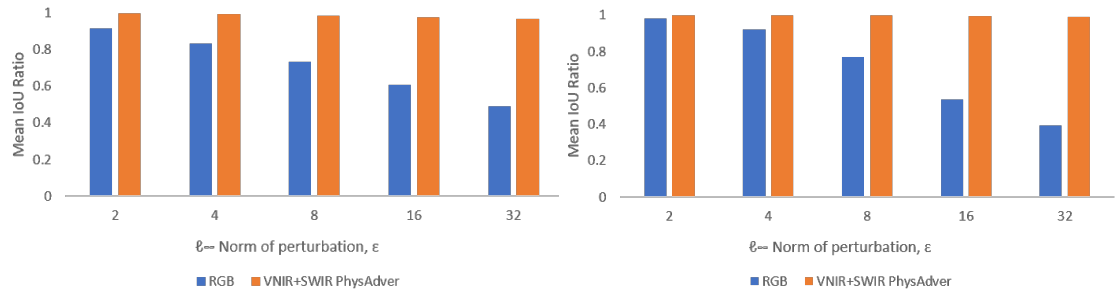
Figure 4.3 shows the mean IoU ratio as a measure of the robustness of the trained models to adversarial examples obtained in a white box setting with different l_∞-norm of perturbation (2, 4, 8, 16, 32). From Figure 4.3 we can see that multispectral image-based models are vulnerable to adversarial examples. In fact, it is even easier to fool those models in a white box setting, as they produce lower mean IoU ratio than RGB models for the same amount of perturbation. The intuition behind this result is that with high dimensional images an attacker has more information to manipulate.

4.1.5 Non-RGB Adversarial Examples in the Physical World

Adversarial attacks have proven to be successful in the physical world as well [77, 95]. Adversarial examples in the physical world are normally accomplished by printing the color image of the adversarial examples. To test this, we generated adversarial examples against an RGB image-based semantic segmentation model and used those adversarial examples to modify the RGB part of the input images sent to the model trained on both Visible Near Infrared (VNIR) and Short Wave Infrared (SWIR) images. This is an attempt to study attacks in settings where the attacker is constrained in the information that can be manipulated. In the physical world, modifying RGB is trivial, but modifications in other regions of the spectrum like near infrared are more difficult because other physical conditions like the temperature of the objects need to be manipulated as well. Figure 4.4 provides evidence that this type of attack will not be successful when attacking models that include more spectral information.

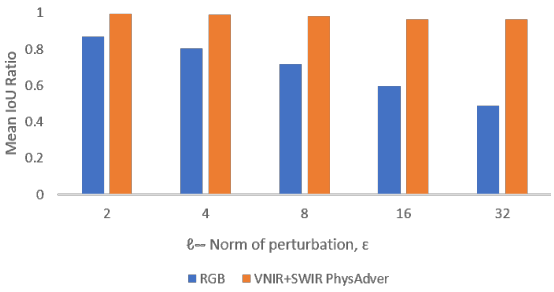
4.1.6 Spectral Signature of Adversarial Examples

It is possible to obtain the spectral signature of the different materials from high dimensional imagery. Figure 4.5 shows the mean reflectance for the pixels belonging to the class "standing water" for the clean input images and adversarial examples crafted using different l_∞-norm of perturbation for two different attacks. Figure 4.5 offers an intuition for

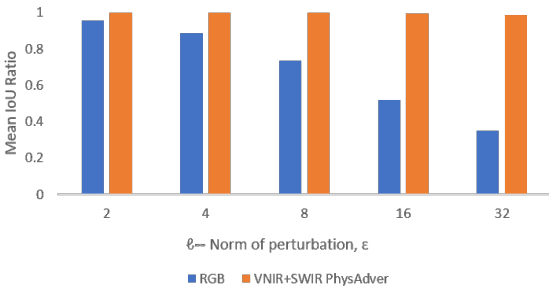


(a) FGSM Attack

(b) FGSM ll Attack

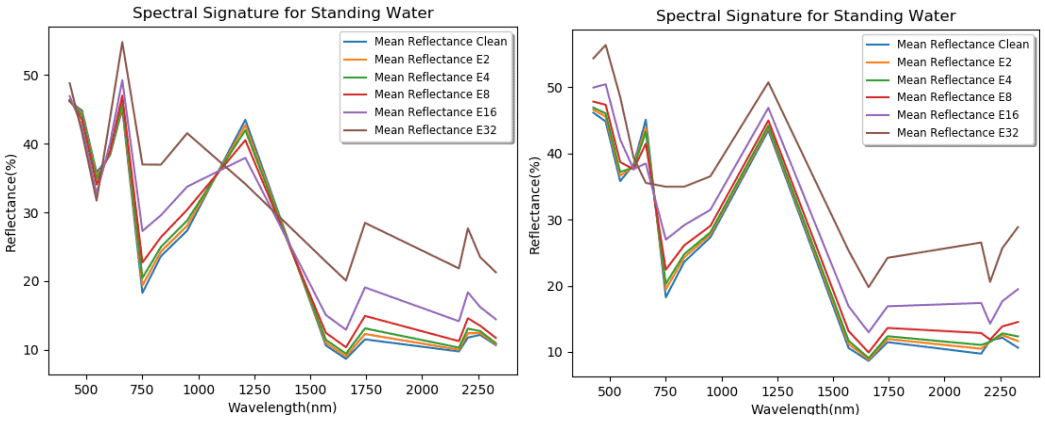


(c) Iterative FGSM Attack



(d) Iterative FGSM ll Attack

Figure 4.4: Attacks in the Physical World. For these set of attacks, the attacker can only manipulate the visible region of the spectrum.



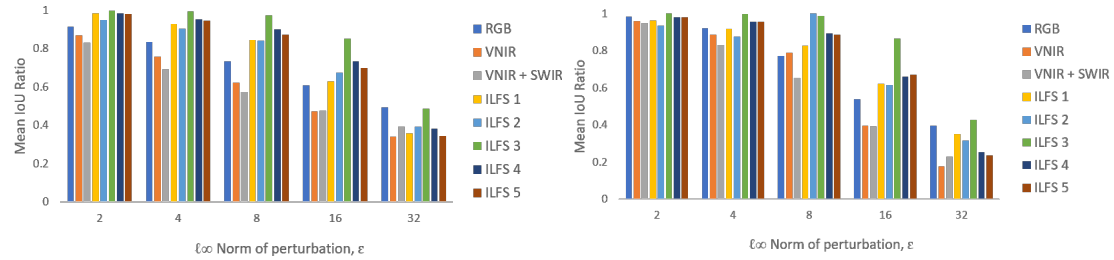
(a) FGSM (b) Iterative FGSM ll

Figure 4.5: Spectral Signature of Adversarial Examples. Larger l_∞-norm produces more drastic changes to the spectral signature of the class.

what the attack is doing on the input image. Larger l_∞-norm perturbations produce more drastic changes in the spectral signature of the class. This could be exploited to actually detect these adversarial perturbations.

4.1.7 ILFS as a Defense to Adversarial Examples

To test the robustness of ILFS to adversarial examples, we attacked ILFS models obtained selecting different number of features with four attack methods and different l_∞-norm of perturbation. We use the mean IoU Ratio as a metric to compare robustness between ILFS models and models trained in the traditional way on RGB, VNIR, VNIR + SWIR regions of the spectrum. Figure 4.6 shows that, in general, ILFS models not only achieve better performance on clean inputs but are also more robust to adversarial examples as their mean IoU ratio is consistently higher. This is particularly the case for all l_∞-norm of perturbation smaller than 32. When the l_∞-norm of perturbation is 32 the margin between the most and least robust model is smaller as none of them perform well. For l_∞-norm of 32 the perturbations are visually obvious and the spectral signature of the different classes is drastically modified (see Figure 4.5) which will not represent acceptable adversarial examples. Moreover, as expected, the Iterative FGSM ll attack is more powerful at fooling



(a) FGSM Attack

(b) FGSM ll Attack

(c) Iterative FGSM Attack

(d) Iterative FGSM ll Attack

Figure 4.6: ILFS as a Defense to Adversarial Examples

networks than single-step FGSM for non-RGB image-based models.

We trained a model using the output bands from ILFS 3 as a fixed input and compared its robustness with the model trained using Integrated Learning and Feature Selection. Figure 4.7 shows that the model trained on fixed inputs is less robust. This supports our hypothesis that as ILFS keeps changing the input space during training, ILFS forces the network to perform well even when the input is maliciously modified.

4.2 Detecting Adversarial Examples

Soil moisture information is one indicator of drought. Remote sensing techniques are able to record accurately the conditions of soil moisture in a large area by using a wetness index. We define a wetness index as follows:

$wetness = \frac{b_{swir2}-b_{swir4}}{b_{swir2}+b_{swir4}}$, where b_{swir2} is the image channel corresponding to the wavelength 1550-1590nm and b_{swir4} is image channel corresponding to the wavelength 1710-1750nm

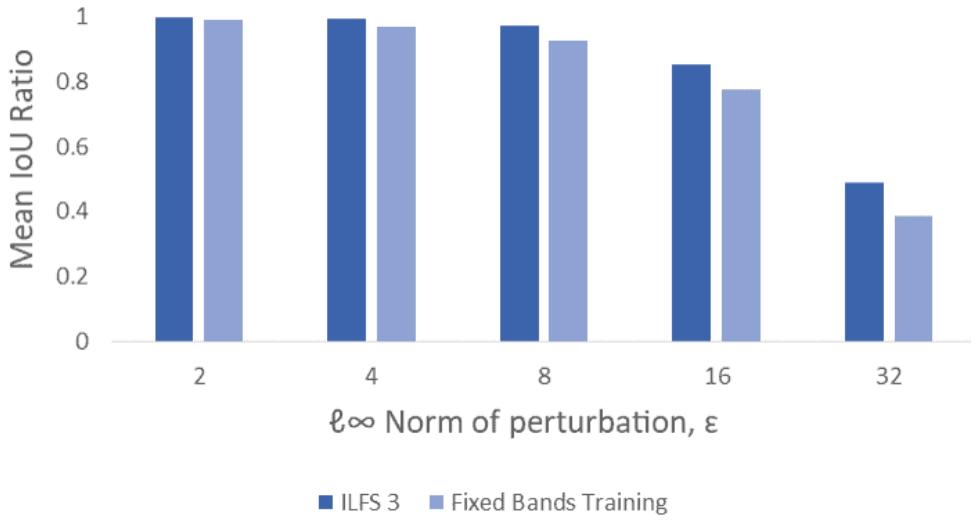


Figure 4.7: Robustness of ILFS 3 vs training on fixed input.

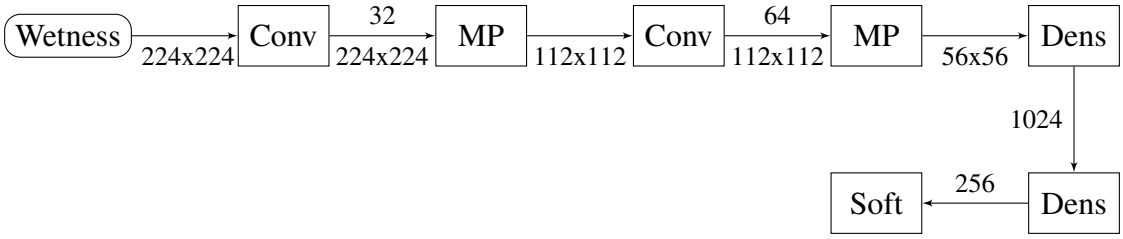


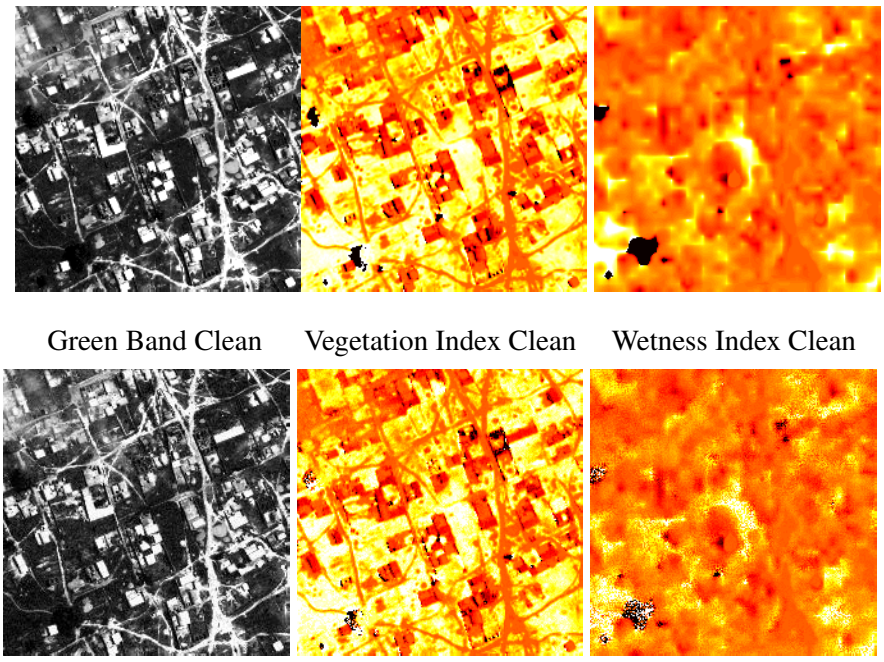
Figure 4.8: Detector Network Architecture. Numbers on top of arrows denote the number of feature maps (neurons in case of dense layers) and numbers below arrows denote spatial resolutions. Conv denotes a convolutional layer, MP denotes a max pooling layer, Soft denotes softmax and Dens a fully-connected layer. Spatial resolutions are decreased by MP. All convolutional layers have 3x3 receptive fields and are followed by batch normalization and rectified linear units.

Table 4.1: Detection Performance

Wetness-based Detector Network Accuracy					
Attack	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 8$	$\varepsilon = 16$	$\varepsilon = 32$
FGSM	0.84	0.99	1.00	1.00	1.00
FGSM ITER	0.94	0.99	1.00	1.00	1.00
FGSM ll	0.83	0.99	1.00	1.00	1.00
FGSM ll ITER	0.95	0.99	1.00	1.00	1.00

The wetness index tends to be uniform among the objects in the scene so even small perturbations on different directions are easy to distinguish. Figure 4.9 shows how the "noise" is definitely noticeable to human eyes in the SWIR wetness index. To exploit this fact we augment the semantic segmentation network by adding a detector subnetwork, which branch off the main network after the input layer and produce an output $p_{adv} \in [0, 1]$ which is interpreted as the probability of the input being adversarial. We call this subnetwork detector network and train it to classify the network inputs into being regular examples or examples generated by a specific adversary. For this, we first train the segmentation networks on the regular (non-adversarial) dataset as usual and subsequently generate adversarial examples for each data point of the train set using the FGSM method discussed previously with $\varepsilon = 8$. We thus obtain a balanced, binary classification dataset of twice the size of the original dataset consisting of the original data (label zero) and the corresponding adversarial examples (label one). From the resulting dataset we obtain the images corresponding to the wetness index of both clean and adversarial data. Then, we freeze the weights of the segmentation network and train the detector such that it minimizes the cross-entropy of p_{adv} and the labels. Figure 4.8 shows the architecture used for the detector network.

Table 4.1 shows the detectability of different adversaries for the proposed detector network. Even though the detector was trained using adversarial examples from a specific



Green Band Clean Vegetation Index Clean Wetness Index Clean

Green Band Adv. Vegetation Index Adv. Wetness Index Adv.

Figure 4.9: Top images are clean. Bottom Images are obtained from an adversarial example. Adversarial noise is noticeable on images obtained using the wetness index

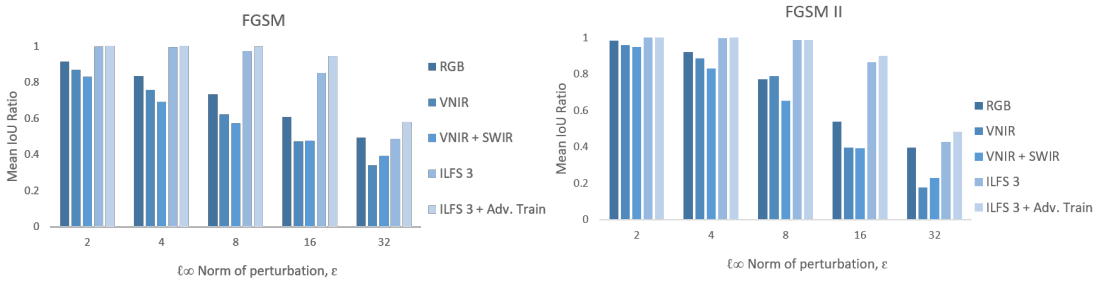
attack and a specific norm of perturbation it is able to generalize to other attacks. For a perturbation greater or equal to 8, the detector networks always detect the attacks. It is important to mention that for our experiment we assume a static adversary, which means that the adversary only has access to the classification network but not to the detector.

4.3 Improving ILFS Robustness through Adversarial Training

Adversarial training increases robustness by augmenting training data with adversarial examples. Madry et al. [82] showed that adversarially trained models can be made robust to white box attacks if the perturbation computed during training maximize the model's loss. We used the obtained adversarial examples from the previous section to augment the original DSTL training set. Then, we retrained the FCN-8 model for semantic segmentation using ILFS and the augmented dataset. This forces ILFS to choose not only the bands that will help a deep neural network to better discriminate objects in a multispectral image, but to choose also bands less sensible to adversarial perturbations. Figure 4.10 shows the mean IoU ratio as a measure of the robustness of the trained models to adversarial examples obtained in a white box setting with different l_∞-norm of perturbation (2, 4, 8, 16, 32). From Figure 4.10 we can see that multispectral image-based models trained using ILFS and adversarial training are more robust to adversarial examples.

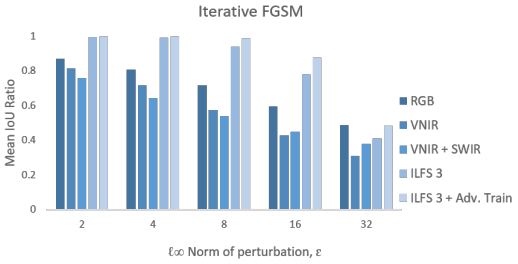
4.4 Proposed Framework

As final framework for semantic segmentation of multispectral images we propose to combine the proposed adversarial detector network with a network trained using ILFS and trained with a training set augmented using adversarial examples. Figure 4.10 shows that most of the mistakes made by ILFS occur when the perturbation is bigger than 8. The adversarial detector network always detects adversaries with a perturbation bigger than 8.

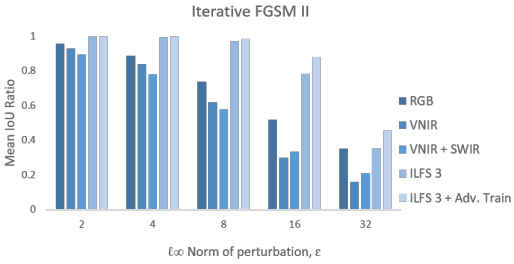


FGSM Attack

FGSM ll Attack



Iterative FGSM Attack



Iterative FGSM ll Attack

Figure 4.10: Robustness of multispectral image-based models to adversarial examples. We observe that models trained on high dimensional images using ILFS along with adversarial training are more robust to adversarial examples.

Because of that, both defense mechanisms complement each other producing a very robust model.

4.5 Chapter Summary and Conclusions

We have shown what, to the best of our knowledge, is the first rigorous evaluation of the robustness of non-RGB image-based machine learning models to adversarial attacks. We showed that known methods to produce adversarial attacks for RGB images generalize to fool non-RGB image-based models with very little or no modifications. In fact, it is even easier to fool this type of systems because more information can be modified.

We showed that applying IFLS increases robustness to adversarial examples in the high dimensional semantic segmentation problem, considering four state-of-the-art attack algorithms. We have also proposed an adversarial detection network that successfully detect attacks coming from static adversaries using four attack algorithms. Detecting adversarial examples works surprisingly well given that the detectability is above 83%. We showed that ILFS defense could be improved by doing adversarial training while selecting the bands. We proposed a framework that integrates the detector network and ILFS trained using adversarial training to obtain more robust models.

In the following chapters we tackle one the generalization and transferability issues which raise when we start applying machine learning models to overhead imagery.

Chapter 5

Conditional Networks: Leveraging Conditioning Information to Improve Generalization

In this chapter we tackle the geographic generalization issue which often happens on overhead image-based machine learning models in real world applications.

As mentioned in the first chapter of this thesis, deep learning has achieved great success in many real-world applications in the past few years. However, training supervised deep learning models requires large-scale datasets [96]. Collecting and annotating datasets which represent a sufficient diversity of real-world test scenarios for every task or domain is costly and time-consuming. Hence, sufficient training data may not always be available. This is particularly true for overhead imagery; we have data from almost everywhere on earth, however, labeled data is scarce. Due to many factors of variation (e.g., weather, season, daytime, illumination, view angle, sensor, and image quality), there is often a distributional change or domain shift that can degrade the performance in real-world applications [97, 98, 99]. Applications on remote sensing and Earth observation, such as land cover mapping, constantly suffer from distributional shifts because of atmospheric changes, seasonality, weather and other variations which translate to unexpected behavior at test time [24, 12].

Different methods to address the problem of covariate shift in transfer learning settings [100, 97, 101] have been proposed in the literature. They consider the case where only the input distribution $p(x)$ differs across domains, while the output (given input) conditional

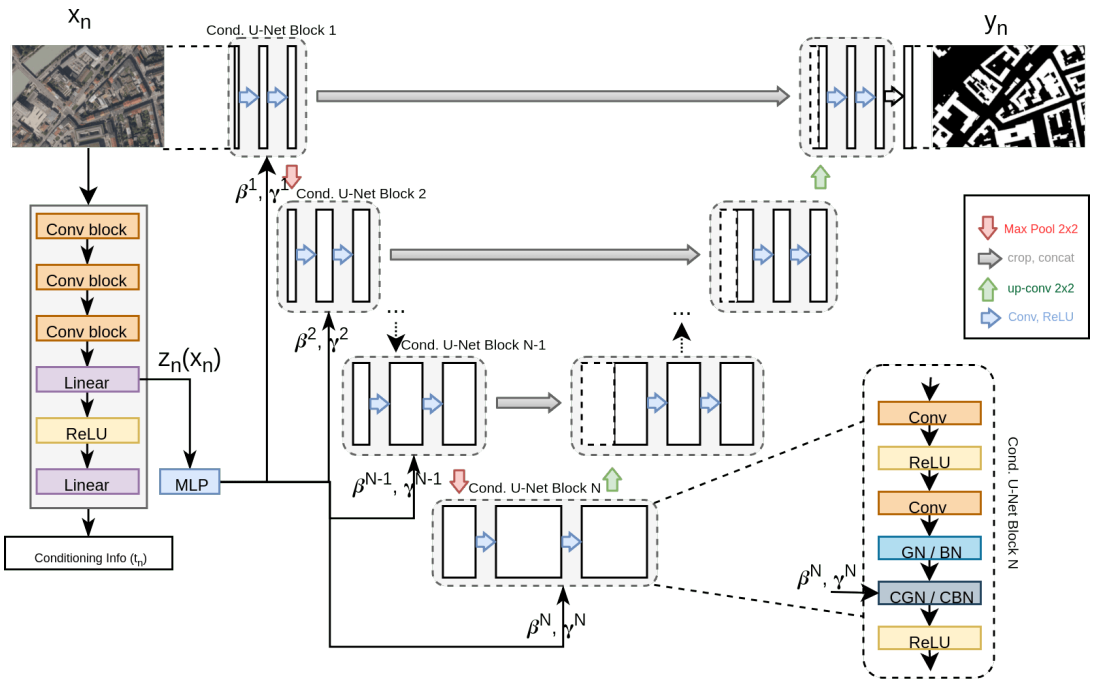


Figure 5.1: Conditional U-Net Network Architecture

distribution $p\,(y|x)$ stays unchanged. A major issue with this approach is that the conditional distribution $p\,(y|x)$ assumption might not be true under many real-world cases.

In this work, we present a novel neural network architecture to increase robustness to distributional changes. Our framework combines conditional computation [102, 103, 104, 105] with a task-specific neural architecture for better domain shift generalization.

One key feature of this architecture is the ability to exploit extra information, often available but seldom used by current models, through a conditioning network. This results in models with better generalization, better performance in both independent and identically distributed (i.i.d.) and non- i.i.d. settings, and in some cases faster convergence. We demonstrate these methodological innovations on an aerial building segmentation task, where test images are from different geographic areas than the ones seen during training [35]. We also found conditional networks to be a generic framework that allows better generalization in other tasks and domains including classification and segmentation

of histopathology images as well, but we would rather focus on the overhead imagery application in this chapter.

We summarize our main contributions from this chapter as follows:

- We propose a novel architecture to effectively incorporate conditioning information, such as metadata in the form of latitude and longitude, season, or any other source of information available.
- We show empirically that our conditional network improves performance in the task of semantic segmentation and sometimes improves convergence.
- We study how conditional networks improve distribution shift generalization and allow fast adaptation.

5.1 Background

Self-supervised learning. Self-supervised learning extract and use available relevant context and embedded metadata as supervisory signals. It is a representation learning approach that allow us to exploit a variety of labels that come with the data for free. To make use of this much larger amount of unlabeled data, one way is to set the learning objectives properly so as to get supervision from the data itself. The self-supervised task, also known as pretext task, guides us to a supervised loss function [106, 107, 108, 109]. However, in self-supervised learning we usually do not put emphasis on the performance on this auxiliary task. Rather we focus in the learned intermediate representation with the expectation that this representation can carry good semantic or structural meanings and can be beneficial to a variety of practical downstream tasks. Conditional networks can be seen as a self-supervision approach in which the pretext task is jointly learn along the downstream task.

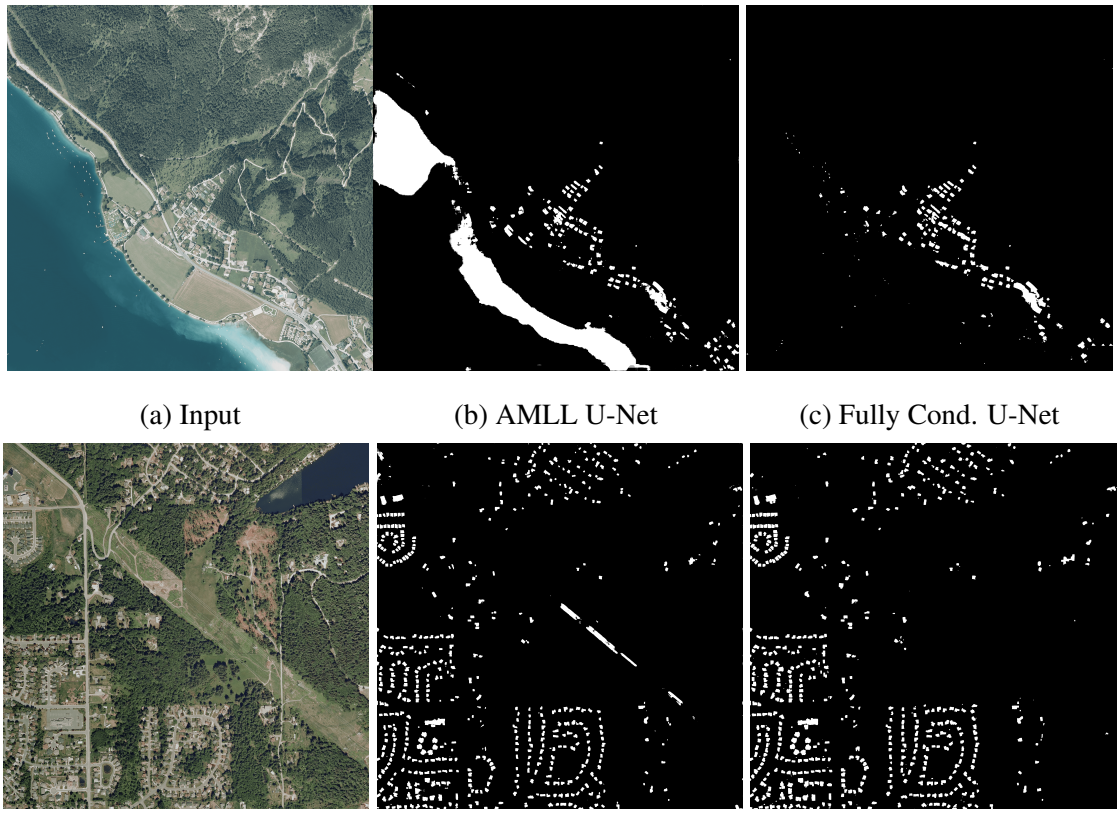
Our proposed modulation of a segmentation architecture based on an intermediate representation of an auxiliary network can also be seen as an instance of *knowledge trans-*

fer [110, 111, 112]. Because the auxiliary network has an additional task signal – predicting metadata – information about this task can be transferred to the segmentation network.

Conditional Computation. Ioffe and Szegedy originally designed Batch Normalization (BN) as a technique to accelerate the training of deep neural networks by reducing the internal co-variate shift [113]. BN normalizes the feature maps of a given mini-batch $B = \{F_{i,.,.,.}\}_{n=1}^{N}$ of N training samples as described in Equation 5.1. γ_c and β_c are trainable scale and shift parameters, introduced to keep the representational power of the original network, and ϵ is a constant factor for numerical stability.

$$BN(F_{n,c,h,w}|\gamma_c, \beta_c) = \gamma_c \frac{F_{n,c,h,w} - \mathbb{E}_B[F_{.,c,.,.}]}{\sqrt{\mathrm{Var}_B[F_{.,c,.,.}] + \epsilon}} + \beta_c \tag{5.1}$$

De Vries et al. introduced Conditional Batch Normalization (CBN) as a method for language-vision tasks in [104, 105]. Instead of learning γ_c and β_c in Equation 5.1 directly, CBN defines them as learned functions $\beta_{n,c} = \beta_c(q_n)$ and $\gamma_{n,c} = \gamma_c(q_n)$ of a conditioning input q_n. Note, that this results in a different scale and shift for each sample in a mini-batch. Scale ($\gamma_{n,c}$) and shift ($\beta_{n,c}$) parameters for each convolutional feature are generated and applied to each feature via an affine transformation. Feature-wise transformations frequently have enough capacity to model complex phenomena in various settings [102]. For instance, they have been successfully applied to neural style transfer [103] and visual question answering [105]. This kind of conditional computation scheme is not tied to the type of normalization used. Wu and He recently proposed Group Normalization (GN [114]). GN divides feature maps into groups and normalizes the features within each group. GN only uses the layer dimensions and its computation is independent of batch size. In this work, we also study the use of conditional computation on Group Normalization and refer to it as Conditional Group Normalization (CGN).



(a) Input (b) AMLL U-Net (c) Fully Cond. U-Net

(d) Input (e) AMLL U-Net (f) Fully Cond. U-Net

Figure 5.2: Qualitative Results. (a) Input from Tyrol East city, (b) AMLL U-Net segmentation results for (a), (c) Fully Conditional U-Net CGN segmentation results for (a), (d) Input from Bellingham city, (e) Fully Conditional U-Net CGN segmentation results for (d), (f) Fully Conditional U-Net CGN segmentation results for (d).

5.2 Formulation and Network Architecture

5.2.1 Problem Abstraction

We first establish notation. Let $(x_n)_{n=1}^N$ represent the training images. Each image x_n is associated with a corresponding label for the task of interest, y_n. From now on we refer to the task of interest as main task.

Available extra annotation is denoted by t_n. This is commonly present for overhead imagery where abundant metadata is available. An auxiliary conditioning network learns an embedding z_n of the input image, which is used to predict t_n and to modulate layers of the main task network. There is flexibility in the choice of t_n. For example, in the building segmentation problem – revisited in Section 5.3.2 – t_n could be anything from the name of the neighborhood, its geographic coordinates or a flag indicating whether it is urban or rural.

Given a particular input image x_n, we adapt our segmentation mask y_n depending on the structure of z_n. Effectively, this allows the model to learn a family of regressors, parameterized by z_n's. Each member of this family is then free to adapt to its contextual z_n. The representations z_n must be learned and we propose to guide the learning by the auxiliary task of predicting the metadata t_n. To encourage this, we train an auxiliary network to predict the metadata t_n given x_n and use intermediate high-level features $\hat{z}(x_n)$ as a proxy for z_n. This way of leveraging the additional information indirectly is an important piece of the proposed architecture, since using the raw t_n is not useful for generalization. Consider the example of building segmentation discussed in Section 5.3.2, where we use geographical coordinates as extra information. Imagine the model is presented with an image from a new city, which might have features that are visually similar to cities from the training set, but has very different coordinates. If we conditioned on the location alone, we would not be able to generalize to this new city. Another important property of leveraging metadata like this is that we do not need access to t_n at test time.

Learning $\hat{z}(x_n)$ in this way can give useful hints to the main task network. For exam-

ple, if we know we are in a city (target label t_n) with a certain visual characteristic such as predominantly smaller buildings, we can assume that this will be captured by $\hat{z}\,(x_n)$. This yields a useful prior for the segmentation task as the labeled regions (in the segmentation mask y_n) are probably smaller than usual. Of course, if the image x_n is clear, the model has enough capacity and we have enough training data, such a hint may not be necessary. However, as stated in the introduction, this is usually not feasible in practice, as any reasonably sized dataset cannot densely cover all possible variations. We hypothesize, that while both the main task network and the auxiliary network only receive the image as input, the prior introduced by this architecture and the provided training metadata can help to learn an efficient representation that improves generalization.

5.2.2 Network Architecture

Our proposed architecture transforms any standard convolutional neural network to incorporate conditioning information t with the goal of improving out-of-distribution generalization. After convolutional blocks in a neural network, we add a linear operation to scale by $\beta_{n,c}(\hat{z}\,(x_n))$ and shift by $\gamma_{n,c}(\hat{z}\,(x_n))$ every feature map c. This affine transformation is applied after the common normalization operations according to (sub-)batch or feature map statistics. We refer to our family of networks as Conditional Networks. Figure 5.1 shows this extension applied to the popular U-Net [33] architecture. U-Net is an encoder-decoder network architecture with skip connections. The right side of Figure 5.1 shows the modified U-Net. The left side of Figure 5.1 shows the conditioning network.

We obtain $\beta_{n,c}(\hat{z}\,(x_n))$ and $\gamma_{n,c}(\hat{z}\,(x_n))$ from the conditioning network. The conditioning network is a convolutional architecture [115] followed by a fully-connected layer that predicts metadata t_n as a function of the input image x_n. The pre-activation features before the output layer are used as $\hat{z}\,(x_n)$. The functions $\beta_{n,c}(\hat{z}\,(x_n))$ and $\gamma_{n,c}(\hat{z}\,(x_n))$ mapping $\hat{z}\,(x_n)$ to the scale and shift parameters are implemented with a MLP. Using the latent representations instead of directly using t_n allows us to leverage combinations of features

that were useful in localizing images from previously seen cities, potentially improving generalization.

Because all its parts are differentiable, conditional networks can be trained end-to-end using gradient-based optimization. Our full objective is described in Equation 5.2, where α is a hyperparameter. L_{main_task} represents a standard main task loss. L_{main_task} depends on the task, such as Jaccard, crossentropy, and dice for semantic segmentation. $L_{conditioning}$ ensures the conditioning networks correctly predicts t_n.

$$L_{cond.\_net} = L_{main\_task} + \alpha * L_{conditioning} \tag{5.2}$$

5.3 Experiments

We test conditional networks on the Inria Aerial Image Labeling Dataset. We use geographical coordinates of the images as target data for the auxiliary network (see Section 5.3.2).

5.3.1 Hypotheses

We study the following hypotheses:

H1: Generalization through context. Explicit incorporation of conditioning information improves generalization in semantic segmentation and image classification tasks.

H2: Interpretability. The features learned by the conditioning and the segmentation network reflect context-specific and context-invariant information, respectively.

H3: Sample Complexity. Conditioning information can reduce the number of training-samples needed in order to successfully learn the target task.

5.3.2 Experimental Setup

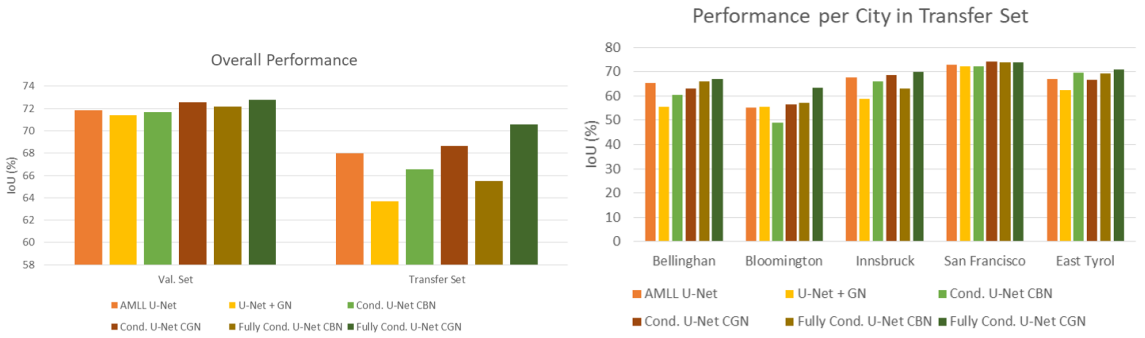
To study hypotheses H1, H2, and H3 we focus on the Inria Aerial Image Labeling Dataset. For H1, we compare model performances using the standard benchmark training-transfer split. For H2, we perform an exploratory visualization of the feature-activation maps for the different models. For H3, we evaluate performance of the different models with respect to the number of image tiles used for training.

Generalization via conditioning. We used the Inria standard train-transfer split to see whether conditioning information helps distribution-shift generalization. From the training set we reserved five images of each city for the validation set as suggested in [35].

For this set of experiments we trained our conditional U-Net presented in Figure 5.1 end-to-end from scratch. We used as segmentation network the Duke AMLL U-Net as described in [116], which is a version of U-Net with fewer filters designed for the Inria dataset. The Duke AMLL U-Net was the winning entry and top of the Inria leaderboard and we use it as baseline for comparison in this section.

The Conditional U-Net was trained exactly as Duke AMLL U-Net. We used the Adam optimizer [32] with cross-entropy as segmentation loss L_{main_task}. We used an initial learning rate of 0.001. We trained our network for 100 epochs, where each epoch consisted of 8000 patches of size 572 x 572 obtained from the image tiles, and the learning rate was reduced to 0.0001 after 60 epochs exactly as for Duke AMLL U-Net, but without any data augmentation technique. We used the standardized latitude and longitude of the center pixel of each patch as the conditioning information to be predicted by the conditioning network and the mean-squared error (MSE) as the conditioning loss $L_{cond.}$.

Binary Cross-Entropy was used as the segmentation loss function. For the conditioning part of the conditional network we used the latitude and longitude of every patch center pixel as t. Latitude and longitude were standardised to be from -1 to +1. The dimension of z is 4374 (9x9x54). z is the input of the MLP predicting for γ and β. We used 1024



(a) Histogram of overall performance

(b) Histogram of per city performance

Figure 5.3: (a) Histogram of overall performance for both validation and transfer set (b) Histogram of per city performance

Table 5.1: Inria Transfer Set Performance.

Method	Bellingham		Bloomington		Innsbruck		San Francisco		East Tyrol		Overall	
	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.
AMLL U-Net	65.37	**96.53**	55.07	95.83	67.62	96.08	72.80	91.00	67.00	96.91	67.98	95.27
U-Net + GN	55.48	93.38	55.47	94.41	58.93	93.77	72.12	89.56	62.27	95.73	63.71	93.45
Cond. U-Net CBN	60.38	95.91	48.84	95.20	65.99	95.97	72.74	90.88	69.70	97.24	66.54	95.04
Cond. U-Net CGN	63.10	96.15	56.43	95.84	68.70	96.23	**74.31**	**91.28**	66.71	97.03	68.66	95.31
Fully Cond. U-Net CBN	65.91	96.49	57.24	95.95	63.14	95.71	73.89	90.83	69.24	97.21	68.50	95.24
Fully Cond. U-Net CGN	**66.98**	96.52	**63.27**	**96.33**	**69.80**	**96.24**	73.88	90.75	**70.77**	**97.35**	**70.55**	**95.44**
$nont_n$ Cond. U-Net	62.97	96.10	53.08	95.03	65.61	95.73	71.87	90.32	65.66	97.75	66.33	94.78

hidden units in the MLP. [1]

Figure 5.3 shows the overall and per-city performance of the models. The fully-conditional U-Net variant using CGN consistently outperforms the other models on every city in the transfer set. The CGN variant where only the encoder is conditioned has a stronger overall performance and generalizes significantly better than the conditional U-Net variants using BN. This is consistent with the observation of [114] that (regular) GN

[1]The source code with details explaining how to run our experiments is available here: `https://github.com/anthonymlortiz/conditional_networks`

Table 5.2: Distribution Shift Generalization Gap.

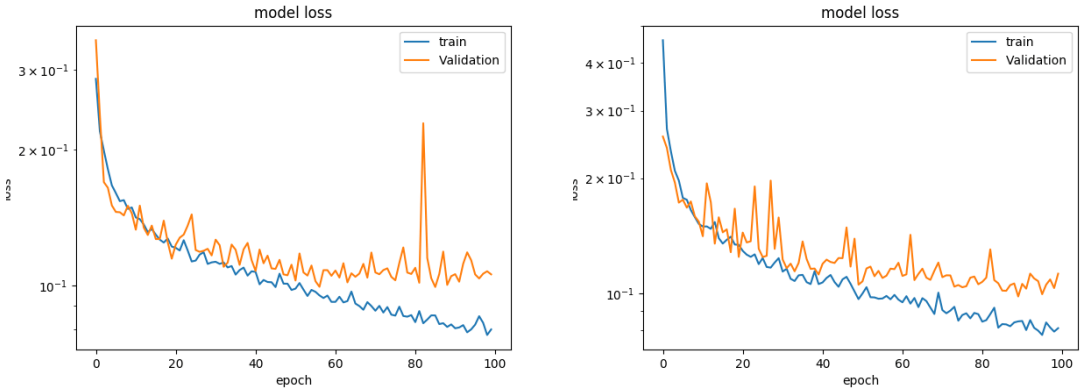
Method	Val. set	Transfer set	Gen. Gap
	IoU (%)	IoU (%)	IoU (%)
AMLL U-Net	71.87	67.98	3.89
U-Net + GN	71.38	63.79	7.59
Conditional U-Net CBN	71.69	66.54	5.15
Conditional U-Net CGN	72.54	68.66	3.88
Fully Conditional U-Net CBN	72.15	68.50	3.65
Fully Conditional U-Net CGN	**72.77**	**70.55**	**2.22**

outperforms BN in segmentation tasks.

Table 5.1 shows the performance of the AMLL U-Net baselines and different variations of our proposed architecture on the transfer set. Conditioning uniformly improved segmentation performance by a big margin over the same models not using conditioning. This empirically validates our hypothesis H1 about generalization through context. We identify as *Conditional U-Net* those models in which only the encoder part of the network is modulated. In the models identified as *Fully Conditional U-Net* both encoder and decoder are modulated, which yielded a small gain in performance. We recommend to condition all blocks since the extra computational cost is very small. Modulating using CGN consistently outperforms CBN.

t_n **Matters.** An experiment using the conditioning network without the auxiliary task of predicting t_n shows a performance that is slightly lower than the baseline U-Net as shown in Table 5.1. We see this as evidence for the importance of guiding the learning process of the latent conditioning representation $\hat{z}\,(x_n)$.

Table 5.2 shows the generalization gap as the difference between the validation set (i.i.d.) and the transfer set performance. Notice how the models' performance consistently



(a) Train loss AMLL U-Net

(b) Train loss Cond. U-Net

Figure 5.4: (a) Training loss AMLL U-Net (b)Training loss AMLL U-Net Conditional U-Net

degrades when we evaluate it on cities not seen during training. Conditioning substantially reduces the generalization gap induced by the distribution-shift between the training and transfer sets, yielding evidence for hypothesis H1.

Figure 5.4 shows the learning curves of the baseline U-Net and the fully conditional U-Net (CGN). The AMLL U-Net seems more prone to overfitting than our conditional U-Net.

Figure 5.2 shows a qualitative comparison in the performance of the proposed network. The baseline labels a beach (5.2a) and power lines (5.2d) as buildings while the Conditional U-Net does not.

Interpretation of conditioning features. To evaluate hypothesis H2, we analyze patterns of activations across these experiments. This hypothesis is interesting for several reasons,

- It serves as a sanity check for the proposed architecture, ensuring that supervision from conditioning information leads to features that do indeed distinguish between cities.

- It sheds light on the potential of using conditioning information to facilitate learning of generalizable features by intentionally learning context-dependent features.

As our first approach towards characterizing feature context-dependence, we associate activations with underlying conditioning information. We apply the following procedure for a few models,

- Compute all activations at a prespecified layer in the network for all patches within an epoch.
- Compute the norm of activations for each feature map in that layer.
- Arrange these values into a patch $\times$ feature matrix, and visualize using either a heatmap or t-SNE.

For U-Net models, we focus on the "bottom" of the U, which has a large number of filters with small spatial extent. In the conditional U-Net, we additionally compute the activations from the last layer before prediction of patch coordinates[2] If we notice separation of patch activations according to conditioning information, we deduce that the learned feature maps are not invariant to that conditioning context.

The learned activations are displayed in Figure 5.5. We find that individual features that activate for a patch in one city tend to activate in a large fraction of patches across all cities. Further, across similar cities, patterns of activation are similar, for both conditioned and unconditioned models. Consider relative similarity between Chicago and San Francisco, for example. For the conditioned model, the large majority of features are zero, across all patches. However, when they are nonzero, their values tend to be larger than the typical activation in unconditioned models. While not obvious from the heatmap, these projections suggest that patterns of activation can be used to distinguish cities, for both conditioned and unconditioned models.

[2]Since these features have no spatial extent, we do not take any norms.

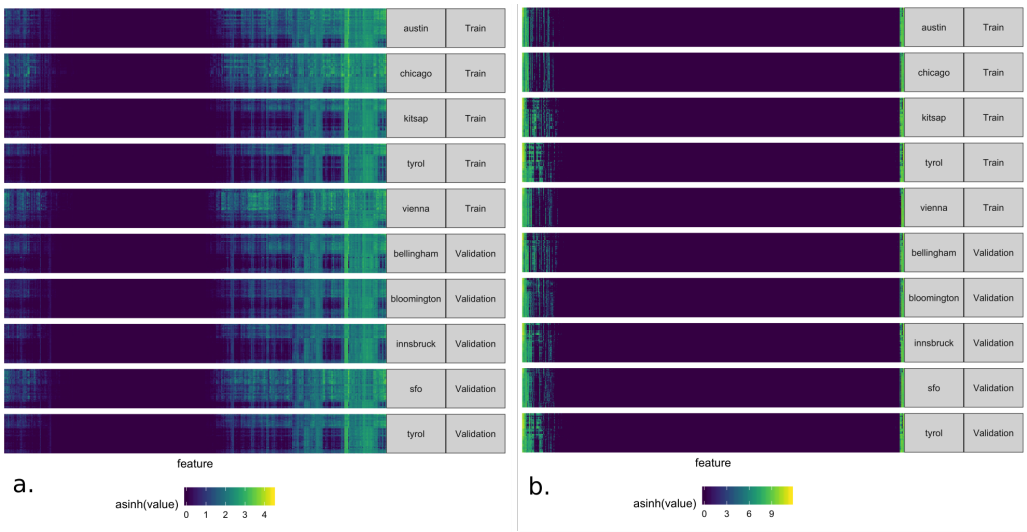
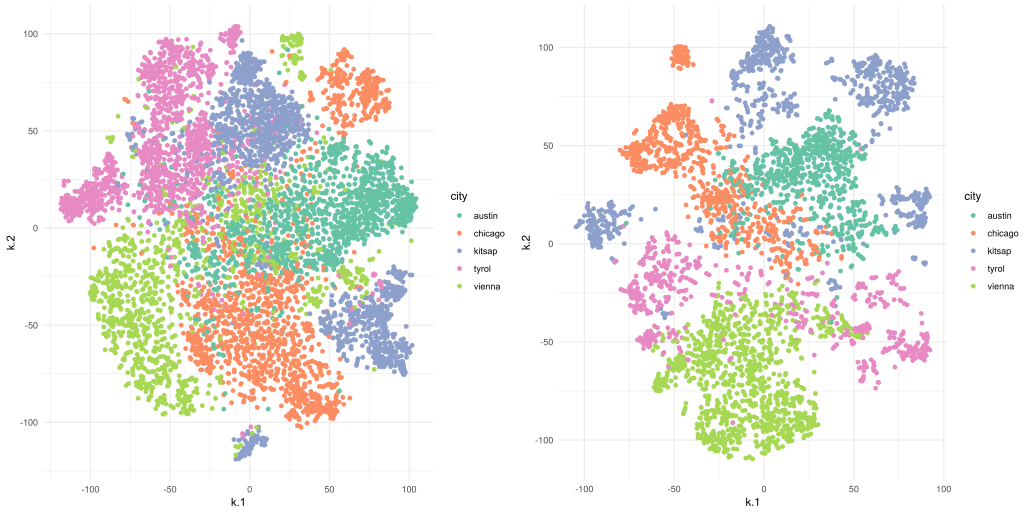


Figure 5.5: Heatmap of activations for (a) AMLL U-Net and (b) conditional U-Net. Cell ij in row i and column j gives the activation of features j on patch i. Rows and columns are sorted so that those that are more similar to one another appear side-by-side. Note that the color legends have different scales.

Figures 5.6 and 5.7 are the t-SNE figures obtained from inspecting the activations of the bottom of the U in the U-Net for both the AMLL U-Net and Conditional U-Net CGN. Figure 5.6 shows the t-SNE associated to sample images from the same cities in the training set. Figure 5.6 shows the t-SNE associated to sample images from both training and transfer set.

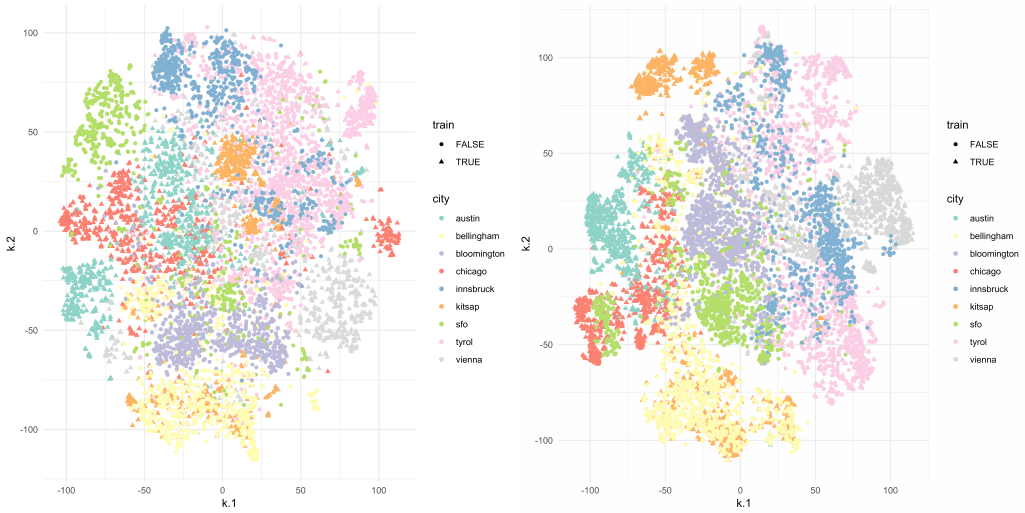
From these observations, we conclude that the improved generalization ability of the conditional U-Net is not due to any ability to learn features that are more invariant to the identity of the corresponding city. Instead, it appears that the conditional U-Net learns a smaller collection of features that are ultimately more useful in the downstream segmentation task. We speculate that having fewer active features for any prediction allows for sharper predictions, preventing "blurring" that could result from averaging across feature maps.

In light of the differences in scale and sparsity of feature activations in the conditional



(a) t-SNE AMLL U-Net train set (b) t-SNE Cond. U-Net train set

Figure 5.6: t-SNE based on activations obtained using image tiles from the cities in the training set. (a) AMLL U-Net train set (b) Cond. U-Net CGN train set



(a) t-SNE AMLL U-Net train and transfer sets (b) t-SNE Cond. U-Net train and transfer sets

Figure 5.7: t-SNE based on activations obtained using image tiles from cities in both training and transfer sets. (a) AMLL U-Net train and transfer sets (b) Cond. U-Net CGN train and transfer sets

and AMLL U-Nets, we zoom into individual features of interest in Figure 5.8. It is not surprising that the overall y-axis scale differs between the two networks – this is clear by inspecting the legend of Figure 5.5. However, we observe that there are more instances of high variation in activation across patches in the conditional U-Net, compared to the AMLL U-Net, among both low and high variance features. Indeed, in the conditional U-Net, there seem to be features that, while typically active, are often exactly or near zero, and similarly, features that are typically inactive, but occasionally spike. This type of variation is even observed within individual cities.

This suggests a type of specificity in the learned features. Rather than activating slightly more or slightly less across all patches, features seem sensitive to particular features within the patches that they activate.

Sample Complexity Analysis. To test hypothesis H3, we evaluate performance of the different models depending on the number of image tiles available while training. The original Inria Aerial training set holds 180 image tiles from five different cities (36 tiles per city). Twenty five training image tiles (five per city) are reserved for validation. We used the remaining training images to create 10 different training subsets. The first subset holds three image tiles per city. Every new training subset holds the image tiles from the previous subset plus three new tiles per city. We train the different models from scratch for each training subset and evaluate performance on the same validation and transfer set as in the previous experiments. Notice that while training we randomly sample patches from the different tiles, hence we might end up with slightly different models.

Figure 5.9 shows that when using 18 or more tiles per city, the conditional U-Net consistently outperforms the basic U-Net. This suggests that the conditional formulation can leverage the metadata to learn a better representation when given a sufficient amount of training data.

Figure 5.9 shows the overall performance of the models as a function as the number of image tiles per city used for training. One can see that with 15 or more tiles per city

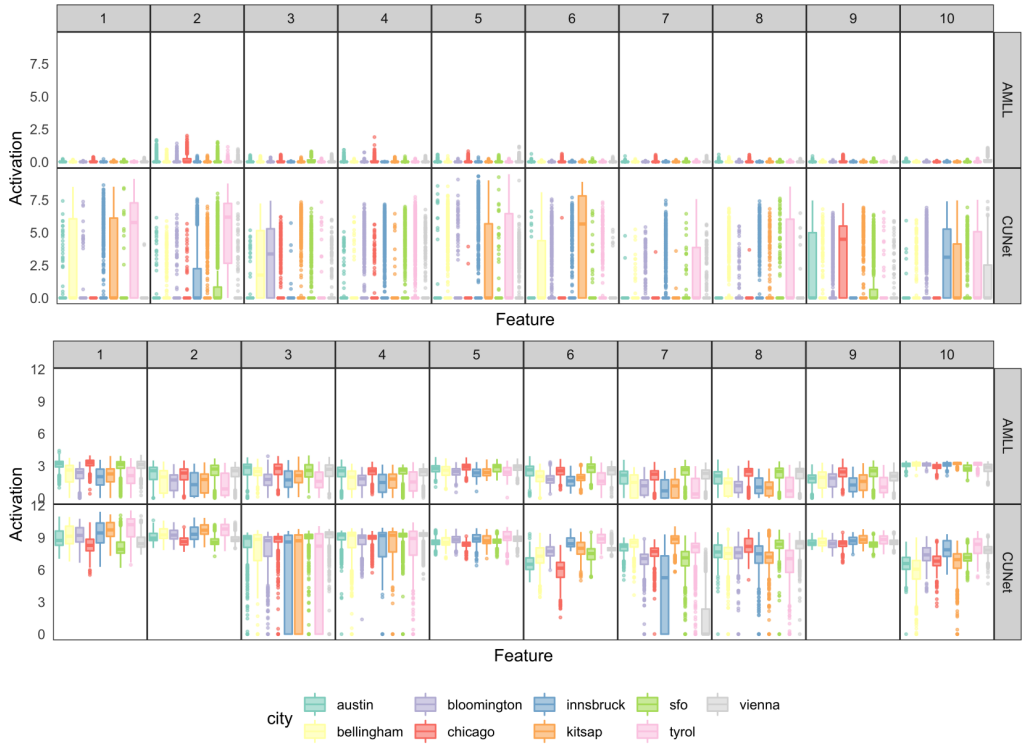


Figure 5.8: The (top) least and (bottom) most variable features, according to interquartile range, in the AMLL and conditional U-Nets, after filtering to those features that are active in at least 10% of patches. Columns 1-10 give the 10 most and least variable features, for least and most variable features, respectively. The y-axis is the activation for each feature. Patches are split into individual cities, and a boxplot of each city's activation values is given.

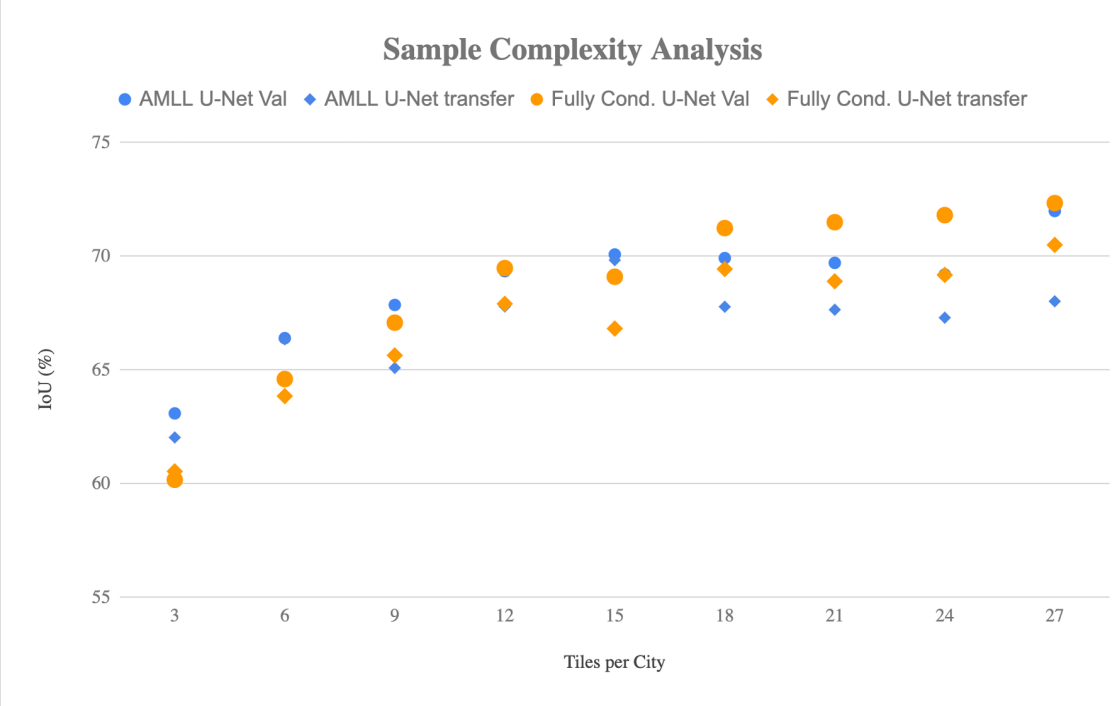


Figure 5.9: Sample Complexity Analysis. The y-axis represents the overall performance across all cities either in the validation or transfer set. Conditional U-Net models appear in orange, while the baseline appears in blue. The circular dots represent results on the validation set. Results in the transfer set are shown using diamonds. (Best seen in color)

the conditional U-Net performs better than the baseline in both i.i.d and non-i.i.d. settings. The generalization gap (difference between circular dots and diamonds of the same color) is smaller as well.

5.4 Chapter Summary Conclusions

In this chapter we have presented a network architecture that leverages conditional information to improve generalization in aerial image labeling. We have shown that conditional networks consistently reduce the loss in performance observed when there is a shift of the underlying distribution at test time. After carefully studying the network feature activations, we found that the improved generalization ability of the proposed network is not due to the ability of learning more invariant features. It appears, instead, that the conditional network learns a smaller collection of features more relevant to the task. One downside of conditional networks is the need for extra annotation t_n. It is not always obvious which choice of t_n helps the most to better estimate z_n. On a side study not included in this thesis, we found conditional networks to be a generic framework for better generalization in other tasks, domains, and images sources including classification and segmentation of histopathology images. Unfortunately, even after applying all these tricks, models are unable to generalize to types of geographies far away from the training data. This enforces the need for transfer learning and techniques to facilitate it. In the following chapter we propose a novel normalization layer which facilitates transfer learning and improves performance on land cover mapping, and semantic segmentation in general.

Chapter 6

Local Context Normalization: Revisiting Local Normalization

As discussed in the previous chapter, recent research has shown that Batch Normalization (BN) [113] facilitates convergence of very deep learning architectures by smoothing the optimization landscape [117]. BN normalizes the features by the mean and variance computed within a mini-batch. Using the batch dimension while calculating the normalization statistics has two main drawbacks:

- Small batch sizes affect model performance because the mean and variance estimates are less accurate.
- Batches might not exist during inference, so the mean and variance are pre-computed from the training set and used during inference. Therefore, changes in the target data distribution lead to issues while performing transfer learning, since the model assumes the statistics of the original training set [118].

To address both of these issues, Group Normalization (GN) was recently proposed by Wu and He [114]. GN divides channels into groups and normalizes the features by using the statistics within each group. GN does not exploit the batch dimension so the computation is independent of batch sizes and model performance does not degrade when the batch size is reduced. GN shows competitive performance with respect to BN when the batch size is small; consequently, GN is being quickly adopted for computer vision tasks like segmentation and video classification, since batch sizes are often restricted for those applications. When the batch size is sufficiently large, BN still outperforms GN.

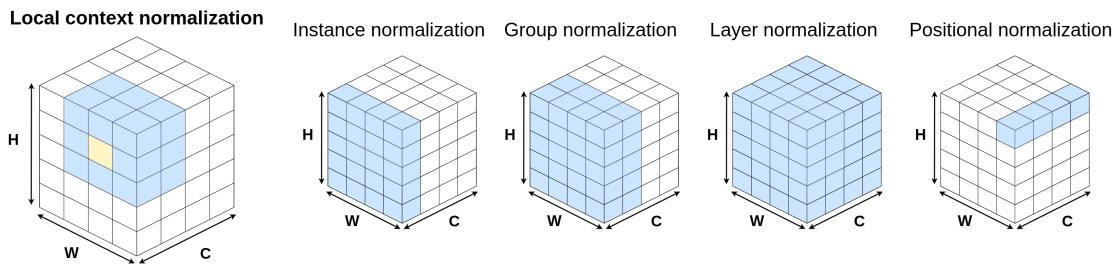


Figure 6.1: **Proposed *Local Context Normalization* (LCN) layer**. LCN normalizes each value in a channel according to the values in its feature group and spatial neighborhood. The figure shows how our proposed method compares to other normalization layers in terms of which features are used in normalization (shown in blue), where **H**, **W**, and **C** are the height, width, and number of channels in the output volume of a convolutional layer.

BN, GN, Instance Normalization (IN), and Layer Normalization (LN) all perform "global" normalization where spatial information is not exploited, and all features are normalized by a common mean and variance value. We argue that for the aforementioned applications, local context matters. To incorporate this intuition we propose *Local Context Normalization* (LCN) as a normalization layer which takes advantage of the context of the data distribution by normalizing each feature based on the statistics of its local neighborhood and corresponding feature group. LCN is in fact inspired by computational neuroscience, specifically the *contrast normalization* approach leveraged by the human vision system [119]. LCN provides a performance boost over all previously-proposed normalization techniques, independent of the batch size, while keeping the advantages of being computationally agnostic to the batch size and suitable for transfer learning. We empirically demonstrate the performance benefit of LCN for object detection as well as semantic and instance segmentation. LCN is a generic normalization layer and it's application is not limited to overhead imagery. LCN can be used on neural networks trained on any type of image source.

Another issue with GN is that because it performs normalization using the entire spatial dimension of the features, when it is used for inference in applications where input images

need to be processed in patches, just shifting the input patch for a few pixels produces different predictions. This is a common scenario in geospatial analytics and remote sensing applications where the input tends to cover an immense area [12, 25].

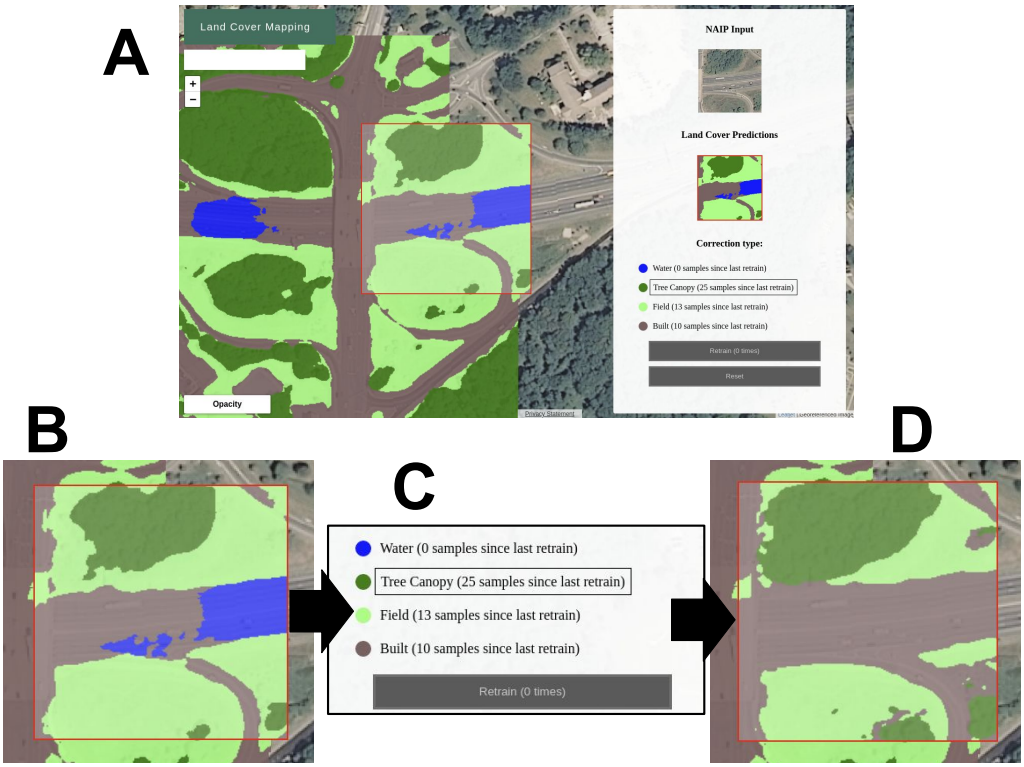


Figure 6.2: Interactive tool for land cover labeling tool. (**A**) Prediction results are overlaid on top of the map. (**B**) The user can easily identify misclassified pixels and (**C**) submit corrections by clicking on the map. (**D**) Pressing "Retrain" updates the model and displays new land cover predictions in the interface. In this example, the user provided a handful of point corrections in the impervious surface initially misclassified as water. LCN facilitates fine-tuning for this type of systems.

Because currently it is very costly to gather labels from everywhere in the world and classes of interest tend to change depending on the problem being solved, a more efficient way to tackle the geographic generalization issue is through interactive tools like the one shown in Figure 6.2. A base model is created and its predictions for an area of interest are

exposed to users through an interface. Users can fine-tune the model by providing training samples and retraining until the model works well in the area of interest and/or the user is happy with the predictions [27]. These type of interactive systems require of base models that facilitate fine-tuning. These tools become infeasible using GN, since a user won't be able to recognize whether changes in the predictions are happening because of fine-tuning or simply because of changes in the image input statistics. With LCN, predictions depend only on the statistics within the feature neighborhood; inference does not change when the input is shifted.

6.1 Local Context Normalization

6.1.1 Formulation

In the previous chapter we briefly refer to some normalization layers, now we revisit then and explain it in detail in order to provide required background for the formulation of LCN.

Local Normalization. In the Local Response Normalization (LRN) scheme proposed by [120], every feature $x_{i,h,w}$ – where i is refers to channel i and h,w refer to spatial position of the feature – is normalized by equation 6.1, where W_{pq} is a Gaussian weighting window of size 9×9, $\sum_{ipq} W_{pq} = 1$, c is set to be $mean(\sigma_{hw})$, and σ_{hw} is the weighted standard deviation of all features over a small spatial neighborhood. h and w are spatial coordinates, and i is the feature index.

$$\widehat{x}_{ihw} = \frac{x_{ihw} - \sum_{ipq} W_{pq}.x_{i,h+p,w+q}}{max(c, \sigma_{hw})} \tag{6.1}$$

Global Normalization. Most recent normalization techniques, including BN, LN, IN, and GN, apply global normalization. In these techniques, features are normalized follow-

ing equation 6.2.

$$\widehat{x}_i = \frac{x_i - \mu_i}{\sigma_i} \tag{6.2}$$

For a 2D image, $i = (i_B, i_C, i_H, i_W)$ is a 4D vector indexing the features in (B, C, H, W) order, where B is the batch axis, C is the channel axis, and H and W are the spatial height and width axes. μ and σ are computed as:

$$\begin{aligned} \mu_i &= \frac{1}{m} \sum_{k \in S_i} x_k \\ \sigma_i &= \sqrt{\frac{1}{m} \sum_{k \in S_i} (x_k - \mu_i)^2 + \epsilon} \end{aligned} \tag{6.3}$$

with ϵ as a small constant. S_i is the set of pixels in which the mean and standard deviation are computed, and m is the size of this set. As shown by [114], most recent types of feature normalization methods mainly differ in how the set S_i is defined. Figure 6.1 shows graphically the corresponding set S_i for different normalization layers.

For BN, statistics are computed along (B, H, W). S_i is defined as:

$$\text{BN} \implies S_i = \{k | k_c = i_C\} \tag{6.4}$$

For LN, normalization is performed per-sample, within each layer. μ and σ are computed along (C, H, W). S_i is defined as:

$$\text{LN} \implies \mathcal{S}_i = \{k | k_B = i_B\}, \tag{6.5}$$

For IN, normalization is performed per-sample, per-channel. μ and σ are computed along (H,W). S_i is defined as:

$$\text{IN} \implies \mathcal{S}_i = \{k | k_B = i_B, k_C = i_C\}, \tag{6.6}$$

For GN, normalization is performed per-sample, within groups of size G along the channel axis as shown in equation 6.7.

$$\text{GN} \implies \mathcal{S}_i = \{k | k_B = i_B, \lfloor \frac{k_C}{C/G} \rfloor = \lfloor \frac{i_C}{C/G} \rfloor\}, \tag{6.7}$$

All global normalization schemes (GN, BN, LN, IN) learn a per-channel linear transformation to compensate for the change in feature amplitude:

$$y_i = \gamma \widehat{x}_i + \beta \tag{6.8}$$

where γ and β are learned during training.

Local Context Normalization. In LCN, the normalization statistics μ and γ are computed following equation 6.2 using the set S_i defined by 6.9. We propose performing the normalization per-sample, within a window of size $p \times q$, for groups of filters of size predefined by the number of channels per group (c_group) along the channel axis, as shown in equation 6.9. We consider windows much bigger than the ones used in LRN and can compute μ and γ in a computationally efficient manner. The size p and q should be adjusted according to the input size and resolution and can be different for different layers of the network.

$$\text{LCN} \implies \mathcal{S}_i = \{k | k_B = i_B, \lfloor \frac{k_C}{c\_group} \rfloor = \lfloor \frac{i_C}{c\_group} \rfloor, \\ \lfloor \frac{k_H}{p} \rfloor = \lfloor \frac{i_H}{p} \rfloor, \lfloor \frac{k_W}{q} \rfloor = \lfloor \frac{i_W}{q} \rfloor \}, \tag{6.9}$$

Relation to Previous Normalization Schemes. LCN allows an efficient generalization of most previously proposed mini-batch-independent normalization layers. Like GN, we perform per-group normalization. If the chosen p is greater than or equal to H and the chosen q is greater than or equal to W, LCN behaves exactly as GN, but keeping the number of channels per group fixed throughout the network instead of the number or groups. If in that scenario the number of channels per group (c_group) is chosen as the total number of channels (c_group = C), LCN becomes LN. If the number of channels per group (c_group) is chosen as 1 (c_group = 1), LCN becomes IN.

6.1.2 Implementation

LCN can be implemented easily in any framework with support for automatic differentiation like PyTorch [121] and TensorFlow [122]. For an efficient calculation of mean and variance, we used the summed area table algorithm, also known in computer vision as the integral image trick [123], along with dilated convolutions [29, 124]. Algorithm 1 shows the pseudo-code for the implementation of LCN. We first create two integral images using the input features and the square of the input features. Then, we apply dilated convolution to both integral images with proper dilation (dilation depends on c_group, p, and q), kernel and stride of one. This provides us the sum and sum of squares tensors for each feature x_{ihw} within the corresponding window and group. From the sums and sum of square tensors we obtain mean and variance tensors needed to normalize the input features. Note that the running time is constant with respect to the window size making LCN efficient for arbitrarily large windows.

6.2 Experimental Results

In this section we evaluate our proposed normalization layer for the tasks of object detection, semantic segmentation, and instance segmentation in several benchmark datasets, and we compare its performance to the best previously known normalization schemes.

6.2.1 Semantic Segmentation on Cityscapes

Semantic segmentation consists of assigning a class label to every pixel in an image. Each pixel is typically labeled with the class of an enclosing object or region. We test for semantic segmentation on the Cityscapes dataset [125] which contains 5,000 finely-annotated images. The images are divided into 2,975 training, 500 validation, and 1,525 testing images. There are 30 classes, 19 of which are used for evaluation.

Algorithm 1 LCN pseudo-code

Input: x: input features of shape [B, C, H, W],

c_group: number of channels per group ,

$window\_size$: spatial window size as a tuple (p, q),

γ, β: scale and shifting parameters to be learned

Output: $\{y = LCN_{\gamma,\beta}(x)\}$

$S \leftarrow dilated\_conv(I(x), d, k)$ `/* I(x) is integral image of x, dilation d is (c_group,p,q), kernel k is a tensor with -1 and 1 to substract or add dimension */`

$S_{sq} \leftarrow dilated\_conv(I(x_{sq}), d, k)$ `// I(`x_{sq}`) is integral image of` x_{sq}

$\mu \leftarrow \frac{S}{n}$ `// Compute Mean` $n = c\_group * p * q$

$\sigma^2 \leftarrow \frac{1}{n}(S_{sq} - \frac{S \odot S}{n})$ `// compute Variance`

$\widehat{x} \leftarrow \frac{x-\mu}{\sqrt{\sigma^2+\epsilon}}$ `// Normalize activation`

$y \leftarrow \gamma\widehat{x} + \beta$ `// Apply affine transform`

Table 6.1: Cityscapes Semantic Segmentation Performance

Method	Normalization	mIoU Class (%)	Pixel Acc. (%)	Mean Acc. (%)
HRNetV2 W48	BN	76.22	**96.39**	83.73
HRNetV2 W48	GN	75.08	95.84	82.70
HRNetV2 W48	LCN (ours)	**77.49**	96.14	**84.60**
HRNetV2 W18 Small v1	BN	71.27	**95.36**	79.49
HRNetV2 W18 Small v1	IN	69.74	94.92	77.77
HRNetV2 W18 Small v1	LN	66.81	94.51	75.46
HRNetV2 W18 Small v1	GN	70.31	95.03	78.99
HRNetV2 W18 Small v1	LCN (ours)	**71.77**	95.26	**79.72**
Δ GN		1.46	0.23	0.73

Table 6.2: GN Performance for Different Numbers of Groups

Method	Number of Groups	mIoU Class (%)	Pixel Acc. (%)	Mean Acc. (%)
HRNetV2 W18 Small v1	1 (=LN)	66.81	94.51	75.46
HRNetV2 W18 Small v1	2	69.28	94.78	77.39
HRNetV2 W18 Small v1	4	67.00	94.50	76.13
HRNetV2 W18 Small v1	8	67.67	94.76	75.81
HRNetV2 W18 Small v1	16	**70.31**	**95.03**	**78.99**
HRNetV2 W18 Small v1	C (=IN)	69.74	94.92	77.77

Implementation Details. We train state-of-the-art HRNetV2 [126] and HRNetV2-W18-Small-v1 networks as baselines [1]. We follow the same training protocol as [126]. The data is augmented by random cropping (from 1024×2048 to 512×1024), random scaling in the range of [0.5, 2], and random horizontal flipping. We use the Stochastic Gradient Descent (SGD) optimizer with a base learning rate of 0.01, momentum of 0.9, and weight decay of 0.0005. The poly learning rate policy with the power of 0.9 is used for reducing the learning rate as done in [126]. All the models are trained for 484 epochs. We train HRNetV2 using four GPUs and a batch size of two per GPU. We then substitute sync-batch normalization layers by BN, GN, LCN and compare results. We do exhaustive comparisons using HRNetV2-W18-Small-v1, which is a smaller version of HRNetV2; all training details are kept the same except for the batch size, which is increased to four images per GPU for faster training.

Quantitative Results. Table 6.1 shows the performance of the different normalization layers on the Cityscapes validation set. In addition to the mean of class-wise intersection over union (mIoU), we also report pixel-wise accuracy (Pixel Acc.) and mean of class-wise pixel accuracy (Mean Acc.).

[1] We used the official implementation code from: `https://github.com/leoxiaobin/deep-high-resolution-net.pytorch`

We observe that our proposed normalization layer outperforms all other normalization techniques including BN. LCN is almost 1.5% better than the best GN configuration in terms of mIoU. For LCN, c_group was chosen as 2, with a window size of 227 $\times$ 227 (p = q = 227) for HRNetV2 W18 Small v1 and 255 $\times$ 255 for HRNetV2 W48. For GN, we tested different numbers of groups as shown in Table 6.2, and we report the best (using 16 groups) for comparison with other approaches in Table 6.1. Table 6.2 shows that GN is somewhat sensitive to the number of groups, ranging from 67% to 70.3% mIoU. Table 6.2 also shows results for IN and LN, both of which perform worse than the best GN performance. These results were obtained using HRNetV2-W18-Small-v1 network architecture. It is important to mention that we used the same learning rate values to train all models, which implies that LCN still benefits from the same fast convergence as other global normalization techniques; this is not true for local normalization schemes such as LRN, which tend to require lower learning rates for convergence.

Sensitivity to Number of Channels per Group. We tested the sensitivity of LCN to the number of channels per group (c_group) parameter by training models for different values of c_group while keeping the window size fixed to 227 $\times$ 227 (p = q = 227). Table 6.3 shows the performance of LCN for the different number of channels per group, which is fairly stable among all configurations.

Table 6.3: LCN sensitivity to number of channels per group for a fixed window size (227, 227)

Method	Channels per Group	mIoU Class (%)	Pixel Acc. (%)	Mean Acc. (%)
HRNetV2 W18 Small v1	2	**71.77**	**95.26**	**79.72**
HRNetV2 W18 Small v1	4	70.26	95.07	78.49
HRNetV2 W18 Small v1	8	70.14	94.97	78.11
HRNetV2 W18 Small v1	16	70.11	94.78	79.10

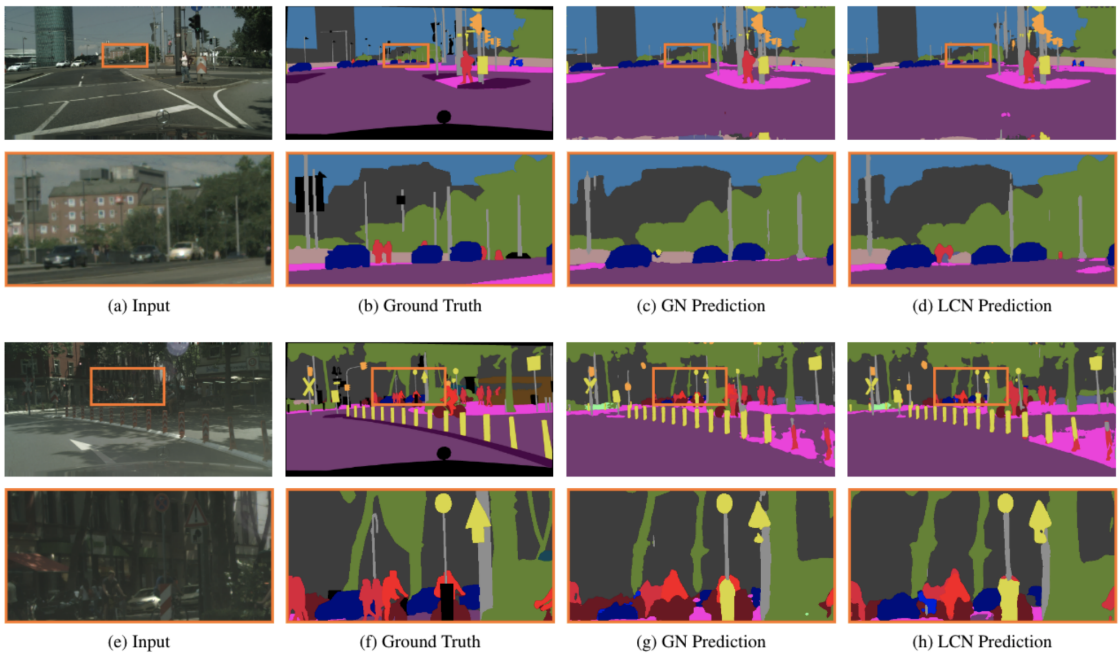


Figure 6.3: **Qualitative results on Cityscapes.** Going from left to right, this figure shows: **Input**, **Ground Truth**, **Group Norm Predictions**, and **Local Context Norm Predictions**. The second and fourth rows are obtained by maximizing the orange area from the images above. We observe how LCN allows the model to detect small objects missed by GN and offers sharper and more accurate predictions.

Sensitivity to Window Size. We also tested how LCN performance varies with respect to changes in window size while keeping the number of channels per group fixed. The results are shown in Table 6.4. The bigger the window size is the closer LCN gets to GN. When the window size (p, q) is equal to the entire spatial dimensions LCN becomes GN. From Table 6.4 we see how performance decreases as the window size gets closer to the GN equivalent.

Qualitative Results Figure 6.3 shows two randomly selected examples of the semantic segmentation results obtained from HRNetV2-W18-Small-v1 using GN (last column) and LCN (second-to-last column) as the normalization layers. The second and fourth rows are

Table 6.4: LCN sensitivity to Window Size

Method	Window Size	mIoU Class (%)	Pixel Acc. (%)	Mean Acc. (%)
HRNetV2 Small v1	199	71.55	95.18	**79.89**
HRNetV2 Small v1	227	71.77	**95.26**	79.72
HRNetV2 Small v1	255	**71.80**	95.18	79.26
HRNetV2 Small v1	383	70.09	95.06	77.64
HRNetV2 Small v1	511	70.03	95.09	77.94
HRNetV2 Small v1	all/GN	70.30	95.04	78.97

Table 6.5: Detection and Instance Segmentation Performance on the Microsoft Coco Dataset

Method	AP^{bbox} (%)	AP^{bbox}_{50} (%)	AP^{bbox}_{75} (%)	AP^{mask} (%)	AP^{mask}_{50} (%)	AP^{mask}_{75}
R50 BN	37.47	59.15	40.76	34.06	55.74	36.04
R50 GN	37.34	59.65	40.34	34.33	56.53	36.31
R50 LCN (Ours)	**37.90**	**59.82**	**41.16**	**34.50**	**56.81**	**36.43**

obtained by maximizing the orange area from the images above them. By zooming in and looking at the details in the segmentation results, we see that LCN allows sharper and more accurate predictions. Carefully looking at the second row, we can observe how using GN HRNet misses pedestrians, which are recognized when using LCN. From the last row, we can see that using LCN results in sharper and less discontinuous predictions. LCN allows HRNet to distinguish between the bike and the legs of the cyclist while GN cannot. LCN also provides more precise boundaries for the cars in the background than GN.

6.2.2 Object Detection and Instance Segmentation on Microsoft COCO Dataset

We evaluate our LCN against previously-proposed normalization schemes for object detection and instance segmentation. Object detection involves detecting instances of objects from a particular class in an image. Instance segmentation involves detecting and segmenting each object in an image. The Microsoft COCO dataset [127] is a high-quality dataset which provides labels appropriate for both detection and instance segmentation and is the standard dataset for both tasks. The annotations include both pixel-level segmentation masks and bounding boxes for objects belonging to 80 categories.

These computer vision tasks in general benefit from higher-resolution input. We experiment with the Mask R-CNN baselines [128], implemented in the publicly available Detectron codebase. We replace BN and/or GN by LCN during finetuning, using the model pre-trained from ImageNet using GN. We fine-tune with a batch size of one image per GPU and train the model using four GPUs.

The models are trained in the COCO [127] train2017 set and evaluated in the COCO val2017 set (a.k.a. minival). We report the standard COCO metrics of Average Precision (AP), AP_{50}, and AP_{75}, for both bounding box detection (AP^{bbox}) and instance segmentation (AP^{mask}).

Table 6.5 shows the performance of the different normalization techniques[2]. LCN outperforms both GN and BN by a substantial margin in all experiments, even using hyperparameters tuned for the other schemes.

6.2.3 Image Classification in ImageNet

We also experiment with image classification using the ImageNet dataset [129]. In this experiment, images must be classified into one of 1000 classes. We train on all training

[2]Our results differ slightly from the ones reported in the original paper, but this should not affect the comparison across normalization schemes.

Table 6.6: Image Classification Error on Imagenet

Network Architecture	Normalization	Top 1 Err. (%)	Top 5 Err. (%)
Resnet 50	BN	23.59	6.82
Resnet 50	GN	24.24	7.35
Resnet 50	LCN	24.23	7.22

images and evaluate on the 50,000 validation images, using the ResNet models [130].

Implementation Details. As in most reported results, we use eight GPUs to train all models, and the batch mean and variance of BN are computed within each GPU. We use He's initialization [131] to initialize convolution weights. We train all models for 100 epochs, and decrease the learning rate by $10\times$ at 30, 60, and 90 epochs.

During training, we adopt the data augmentation of Szegedy et al. [132] as used in [114]. We evaluate the top-1 classification error on the center crops of 224×224 pixels in the validation set. To reduce random variations, we report the median error rate of the final five epochs [133].

As in [114] our baseline is the ResNet trained with BN [130]. To compare with GN and LCN, we replace BN with the specific variant. We use the same hyper-parameters for all models. We set the number of channels per group for LCN as 32, and we used $p = q = 127$ for the window size parameters. Table 6.6 shows that LCN offers similar performance as GN, but we don't see the same boost in performance observed for object detection and image segmentation. We hypothesize that this happens because image classification is a global task which might not benefit from local context.

6.2.4 Aerial Image Labeling on INRIA Dataset

We tested LCN on the INRIA dataset described in Chapter 3, where the task is to perform semantic segmentation for the classes of building and non-building.

Table 6.7: Performance in Inria Aerial Image Labeling Dataset. LCN outperforms all the other normalization layers overall.

Method	Bellingham		Bloomington		Innsbruck		San Francisco		East Tyrol		**Overall**	
	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.
U-Net + BN	**65.37**	**96.53**	55.07	95.83	67.62	96.08	72.80	91.00	67.00	96.91	67.98	95.27
U-Net + GN	55.48	93.38	55.47	94.41	58.93	93.77	72.12	89.56	62.27	95.73	63.71	93.45
U-Net + LCN	63.61	96.26	**60.47**	**96.22**	**68.99**	**96.28**	**75.01**	**91.46**	**68.90**	**97.19**	**69.90**	**95.48**

Implementation Details. We trained different versions of U-Net [33] where just the normalization layer was changed. We trained all models in this set of experiments using 572×572 randomly sampled patches from all training image tiles. We used the Adam optimizer with a batch size of 12. All networks were trained from scratch with a starting learning rate of 0.001. We keep the same learning rate for the first 60 epochs and decay it to 0.0001 over the next 40 epochs. Every network was trained for 100 epochs. In every epoch 8,000 patches are seen. Binary cross-entropy loss was used as the loss function.

Table 6.7 summarizes the performance of the different normalization layers in the INRIA aerial image labeling dataset. Our proposed LCN outperforms all the other normalization layers with an overall mIoU almost 2% higher than the next-best normalization scheme, and more than 6% better than GN in terms of overall IoU. LCN provides much better performance than other methods in almost every test city. LCN was trained using a 91×91 window size and four channels per group.

6.2.5 Land Cover Mapping

Finally, we evaluate LCN on a land cover mapping task previously studied in [12, 134]. Land cover mapping is a semantic image segmentation task where each pixel in an aerial or satellite image must be classified as belonging to one of a variety of land cover classes. This process of turning raw remotely sensed imagery into a summarized data product is an important first step in many downstream sustainability related applications. For example,

Table 6.8: Landcover Mapping Tested on Maryland 2013 Test Tiles

Method	mIoU (%)	Pixel Acc. (%)
UNet + BN	76.69	87.15
UNet + GN	74.15	85.18
UNet + LCN	76.51	86.96

the Chesapeake Bay Conservancy uses land cover data in a variety of settings including determining where to target planting riparian forest buffers [42]. The dataset can be found at [134] and contains 4-channel (red, green, blue, and near-infrared), 1m resolution imagery from the National Agricultural Imagery Program (NAIP) and dense pixel labels from the Chesapeake Conservancy's land cover mapping program over 100,000 square miles intersecting 6 states in the northeastern US. We use the Maryland 2013 subset - training on the 50,000 multi-spectral image patches, each of size $256 \times 256 \times 4$, from the train split. We test over the 20 test tiles[3]. Each pixel must be classified as: water, tree canopy / forest, low vegetation / field, or impervious surfaces.

Implementation Details. We trained different versions of U-Net architecture used on [12] for different normalization layers without doing any data augmentation and compared results. We used the Adam optimizer with a batch size of 96. All networks were trained from scratch for 100 epochs with a starting learning rate of 0.001 with decay to 0.0001 after 60 epochs. The multi-class cross-entropy loss was used as criterion. The best GN results are obtained using 8 groups. LCN results are obtained using 4 channels per group and and a 31×31 window.

Table 6.8 shows the mean IoU and Pixel Accuracy of the different normalization layers for land cover mapping. LCN outperforms GN for this task with performance slightly lower than BN. We notice that LCN benefits from larger input images. When input images are small like in this setting the performance boost from using LCN becomes smaller.

[3]Consisting of $\sim 900,000,000$ pixels

6.3 Chapter Summary and Conclusions

In this chapter we proposed *Local Context Normalization* (LCN) as a normalization layer where every feature is normalized based on a window around it and the filters in its group. We empirically showed that LCN outperforms all previously-proposed normalization layers for object detection, semantic segmentation, and instance image segmentation across a variety of datasets. The performance of LCN is invariant to batch size, and it is well-suited for transfer learning and interactive systems.

We note that we used hyper-parameters which were already highly optimized for BN and/or GN without tuning, so it is likely that we could obtain better results with LCN by just searching for better hyper-parameters. In our experiments we also do not consider varying the window size for different layers in the network, but it is a direction worth exploring: adjusting the window size during training via gradient descent may further improve performance for LCN.

Chapter 7

Conclusions and Future Work

This thesis presented solutions to what we see as three of the central issues in learning from overhead imagery - high dimensionality and lack of labeled data, security concerns, and model generalization.

In Chapter 3, we proposed a novel way to perform dimensionality reduction through supervised feature subset selection on the input space jointly with the process of training a neural network to perform the task of interest. *Integrated Learning and Feature Selection (ILFS)*, as we named it, reduces data dimensionality while at the same time also improves performance of the machine learning models for the task of interest. We showed that for the task of pixel-wise image classification ILFS greatly improves performance over standard models trained using all spectral bands as input.

Our study about adversarial examples on neural networks trained using multispectral images presented in Chapter 4 is, to the best of our knowledge, the first rigorous study of the robustness of non-RGB (VNIR, SWIR, Panchromatic) image-based models against adversarial examples. We show that it is not only feasible to generate deceptive examples for machine learning models based on non-RGB images, it is easier than attacking models for RGB images. However, we show that performing band selection using ILFS can substantially improve the robustness of machine learning models against these adversarial examples. We also introduced a framework that integrates input subset feature selection, adversarial training, and a detector network to drastically improve the robustness of this type of models without sacrificing performance.

Lastly, we focused on one of the more important issues for overhead image analysis, the generalization across geographies. In Chapter 5, we presented *conditional networks* as

a network architecture that leverages conditional information to improve generalization to different types of geographies. We showed that conditional networks consistently reduced the loss in performance observed when there is a shift of the underlying distribution at test time. After carefully studying the network feature activations, we found that the improved generalization ability of the proposed network is not due to the ability of learning more invariant features. However, conditional networks learnt a smaller collection of features more relevant to the task of interest.

Unfortunately, models are still unable to generalize to types of geographies far away from any geography seen in the training data. This forced us and other researchers in the area to consider a more interactive approach where humans and machines interact together through model fine-tuning where users provide immediate feedback on mistakes and tune the machine learning models until they are happy with the results [27]. This type of systems require models which can be easily tuned. To tackle that issue we proposed, in chapter 6, *Local Context Normalization* (LCN) as novel normalization layer which facilitates transfer learning and improves performance for overhead image-based tasks, like land cover mapping, and for object detection and segmentation in general with state of the art performance across multiple datasets. LCN is a very efficient generalization of other previously proposed normalization layers including GN, LN, IN, and Positional Normalization (PN). It is invariant to batch size and it is well-suited for transfer learning and interactive systems.

Now we discuss different research directions for overhead image analysis that we consider are worth studying. The model generalization issue is far from being solved and remains one of the more important areas the research community should focus on to create real impact. Besides the fact that we see a lot of room for improvement in the neural network architecture design, perhaps by using *causal learning and/or meta-learning* techniques, we consider a hybrid human-machine interactive system to be a much more compelling approach. In this scenario, for users the machine learning model is simply a powerful tool to augment their work. Users can easily identify the correct labels from over-

head imagery, the issue is scalability. However, it is not feasible for users like NGOs and other nonprofits to hand label every area they care about and even if they do those labels will change with time and the objects of interest change with time and from organization to organization. We argue that a *hybrid human-machine interactive system* is more suitable in this type of settings. Designing proper ways of interaction between human users and machine learning models including finding better ways for users to provide feedback to the models, designing models with efficient and fast adaptation, and novel fine-tuning techniques are critical in our opinion.

Another very exciting area of future research relates to the task of change detection from overhead imagery. Satellite companies are collecting data from everywhere on Earth frequently, but integrating temporal information has been under study. Most approaches to change detection from overhead imagery has been mostly limited to doing prediction in different timestamps and calculating the difference as the change. Novel network architectures must be developed for optimally exploiting the temporal information jointly with the spatial and spectral information of overhead imagery. Smarter ways to *exploit this temporal data* could have great impact for applications like glacier mapping and climate change monitoring. Remote sensing data is often available from different sensor and data modalities at a wide variety spatial and spectral resolutions. Sensor data fusion aims to use these complementary sources of information jointly to create better models. Already prior to a joint information extraction, a crucial step is to develop novel architectures for the matching of images taken from different perspectives and even different imaging modalities.

A particular feature of overhead imagery is the vast availability of raw data and the scarce amount of labels. To make use of this much larger amount of unlabeled data, one way is to set the learning objectives properly so as to get supervision from the data itself. *Self-supervised learning* does just that and should be exploited for overhead image analysis. *Conditional networks* proposed on Chapter 5 indirectly uses self-supervised learning by exploiting metadata to get supervision from it. In self-supervision the self-supervised

task, also known as pretext task, guides initial supervised learning. However, we usually do not care about the final performance of this invented task. Rather we are interested in the learned intermediate representation with the expectation that this representation can carry good semantic or structural meanings and can be beneficial to a variety of practical downstream tasks. Certain pretext task developed for natural image analysis like predicting image rotation [106] might not work because the rotation concept doe not apply to overhead imagery. However, other approaches that use unsupervised contrastive pretraining [135], where representations are learnt by contrasting positive and negative examples, like contrastive predictive coding, MoCo, and SimCLR [107, 108, 109] are promising and should be explored.

References

[1] Nathan A Hagen and Michael W Kudenov, "Review of snapshot spectral imaging technologies," *Optical Engineering*, vol. 52, no. 9, pp. 090901, 2013.

[2] Luca Giannoni, Frédéric Lange, and Ilias Tachtsidis, "Hyperspectral imaging solutions for brain tissue metabolic and hemodynamic monitoring: past, current and future developments," *Journal of Optics*, vol. 20, no. 4, pp. 044009, 2018.

[3] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.

[4] Lichao Mou, Pedram Ghamisi, and Xiao Xiang Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens*, vol. 55, no. 7, pp. 3639–3655, 2017.

[5] Lichao Mou, Pedram Ghamisi, and Xiao Xiang Zhu, "Unsupervised spectral–spatial feature learning via deep residual conv–deconv network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 1, pp. 391–406, 2018.

[6] Adriana Romero, Carlo Gatta, and Gustau Camps-Valls, "Unsupervised deep feature extraction for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1349–1362, 2016.

[7] Ying Li, Haokui Zhang, and Qiang Shen, "Spectral–spatial classification of hyperspectral imagery with 3d convolutional neural network," *Remote Sensing*, vol. 9, no. 1, pp. 67, 2017.

[8] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[9] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*. IEEE, 2015, pp. 4959–4962.

[10] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected topics in applied earth observations and remote sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.

[11] Yushi Chen, Xing Zhao, and Xiuping Jia, "Spectral–spatial classification of hyperspectral data based on deep belief network," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2381–2392, 2015.

[12] Caleb Robinson, Le Hou, Kolya Malkin, Rachel Soobitsky, Jacob Czawlytko, Bistra Dilkina, and Nebojsa Jojic, "Large scale high-resolution land cover mapping with multi-resolution data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12726–12735.

[13] Kolya Malkin, Caleb Robinson, Le Hou, Rachel Soobitsky, Jacob Czawlytko, Dimitris Samaras, Joel Saltz, Lucas Joppa, and Nebojsa Jojic, "Label super-resolution networks," *International Conference on Learning Representations (ICLR)*, 2019.

[14] Kolya Malkin, Anthony Ortiz, Caleb Robinson, and Nebojsa Jojic, "Mining self-similarity: Label super-resolution with epitomic representations," *arXiv preprint arXiv:2004.11498*, 2020.

[15] Fan Li, Linlin Xu, Parthipan Siva, Alexander Wong, and David A Clausi, "Hyperspectral image classification with limited labeled training samples using enhanced ensemble learning and conditional random fields," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2427–2438, 2015.

[16] Martin Ettenberg, "A little night vision-ingaas shortwave infrared emerges as key complement to ir for military imaging.," *Advanced Imaging-Fort Atkinson*, vol. 20, no. 3, pp. 29–33, 2005.

[17] Mackenzie G Glaholt and Grace Sim, "Gaze-contingent center-surround fusion of infrared images to facilitate visual search for human targets," *Journal of Imaging Science and Technology*, vol. 61, no. 1, pp. 10401–1, 2017.

[18] Peter Pallister, Terri D'Souza, Chelsea Black, Nigel Hearns, and Jeffrey C Smith, "Explosive detection strategies for security screening at airports," in *Molecular Technologies for Detection of Chemical and Biological Agents*, pp. 243–251. Springer, 2017.

[19] Duncan A Robertson, David G Macfarlane, Robert I Hunter, Scott L Cassidy, Nuria Llombart, Erio Gandini, Tomas Bryllert, Mattias Ferndahl, Hannu Lindström, Jussi Tenhunen, et al., "High resolution, wide field of view, real time 340ghz 3d imaging radar for security screening," in *Passive and Active Millimeter-Wave Imaging XX*. International Society for Optics and Photonics, 2017, vol. 10189, p. 101890C.

[20] Guanghao Sun, Takemi Matsui, Tetsuo Kirimoto, Yu Yao, and Shigeto Abe, "Applications of infrared thermography for noncontact and noninvasive mass screening of febrile international travelers at airport quarantine stations," in *Application of Infrared to Biomedical Sciences*, pp. 347–358. Springer, 2017.

[21] Allen M Waxman, Mario Aguilar, David A Fay, David B Ireland, and Joseph P Racamato, "Solid-state color night vision: fusion of low-light visible and thermal infrared imagery," *Lincoln Laboratory Journal*, vol. 11, no. 1, pp. 41–60, 1998.

[22] Zhengrong Ying, Sergey Simanovsky, Ram Naidu, and Sorin Marcovici, "Ct scanning systems and methods using multi-pixel x-ray sources," Aug. 22 2017, US Patent 9,739,724.

[23] P WT Yuen and Mark Richardson, "An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition," *The Imaging Science Journal*, vol. 58, no. 5, pp. 241–253, 2010.

[24] Xiao Xiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 8–36, 2017.

[25] Anthony Ortiz, Alonso Granados, Olac Fuentes, Christopher Kiekintveld, Dalton Rosario, and Zachary Bell, "Integrated learning and feature selection for deep neural networks in multispectral images," *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[26] Anthony Ortiz, Olac Fuentes, Dalton Rosario, and Christopher Kiekintveld, "On the defense against adversarial examples beyond the visible spectrum," in *Military Communications, track on Big Data and Machine Learning for Tactical Networks (MILCOM 2018)*. IEEE, 2018.

[27] Caleb Robinson, Anthony Ortiz, Kolya Malkin, Blake Elias, Andi Peng, Dan Morris, Bistra Dilkina, and Nebojsa Jojic, "Human-machine collaboration for fast land cover mapping," *AAAI Conference on Artificial Intelligence (AAAI 2020)*, 2020.

[28] Anthony Ortiz, Caleb Robinson, Mahmudulla Hassan, Dan Morris, Olac Fuentes, Christopher Kiekintveld, and Nebojsa Jojic, "Local context normalization: Revisiting local normalization," *IEEE conference on computer vision and pattern recognition*, 2020.

[29] Fisher Yu and Vladlen Koltun, "Multi-scale context aggregation by dilated convolutions," *International Conference on Learning Representations (ICLR)*, 2016.

[30] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[31] Léon Bottou, "Online learning and stochastic approximations," *Online learning in neural networks*, vol. 17, no. 9, pp. 142, 1998.

[32] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *International Conference in Learning Representations (ICLR)*, 2015.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[34] DSTL, "Dstl satellite imagery feature detection.," `https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection`, 2016.

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

[36] Manuel Campos-Taberner, Adriana Romero-Soriano, Carlo Gatta, Gustau Camps-Valls, Adrien Lagrange, Bertrand Le Saux, Anne Beaupere, Alexandre Boulch,

Adrien Chan-Hon-Tong, Stephane Herbin, et al., "Processing of extremely high-resolution lidar and rgb data: outcome of the 2015 ieee grss data fusion contest–part a: 2-d contest," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 12, pp. 5547–5559, 2016.

[37] Adrien Lagrange, Bertrand Le Saux, Anne Beaupere, Alexandre Boulch, Adrien Chan-Hon-Tong, Stéphane Herbin, Hicham Randrianarivo, and Marin Ferecatu, "Benchmarking classification of earth-observation data: From learning explicit features to convolutional networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 4173–4176.

[38] ISPRS, "2d semantic labeling contest.," `http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html`, 2017.

[39] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raska, "Deepglobe 2018: A challenge to parse the earth through satellite images," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2018, pp. 172–17209.

[40] SpaceNet, "Spacenet on amazon web services (aws). "datasets." the spacenet catalog.," `https://spacenetchallenge.github.io/datasets/datasetHomePage.html`, 2018.

[41] *United States Department of Agriculture. National Aerial Imagery Program.*

[42] Chesapeake Conservancy, "Land cover data project 2013/2014," `https://chesapeakeconservancy.org/conservation-innovation-center/high-resolution-data/land-cover-data-project/`, 2016.

[43] Collin Homer, Jon Dewitz, Limin Yang, Suming Jin, Patrick Danielson, George Xian, John Coulston, Nathaniel Herold, James Wickham, and Kevin Megown, "Completion of the 2011 national land cover database for the conterminous united states–representing a decade of land cover change information," *Photogrammetric Engineering & Remote Sensing*, vol. 81, no. 5, pp. 345–354, 2015.

[44] USGS, "United states geological survey. landsat 8.," https://www.usgs.gov/land-resources/nli/landsat/landsat-8?qt-science_support_page_related_con=0#qt-science_support_page_related_con, 2013.

[45] Microsoft Building, "Microsoft. us building footprints.," https://github.com/microsoft/USBuildingFootprints, 2019.

[46] Planet Labs, "Planet lab imagery," https://www.planet.com/, 2016.

[47] OSM, "Open street maps (osm)," https://www.openstreetmap.org/, 2004.

[48] Chao Tao, Hongbo Pan, Yansheng Li, and Zhengrou Zou, "Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification," *IEEE Geoscience and remote sensing letters*, vol. 12, no. 12, pp. 2438–2442, 2015.

[49] Nataliia Kussul, Mykola Lavreniuk, Sergii Skakun, and Andrii Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 778–782, 2017.

[50] Wei Li, Guodong Wu, Fan Zhang, and Qian Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 844–853, 2017.

[51] Lichao Mou, Pedram Ghamisi, and Xiao Xiang Zhu, "Fully conv-deconv network for unsupervised spectral-spatial feature extraction of hyperspectral imagery via residual learning," in *Geoscience and Remote Sensing Symposium (IGARSS), 2017 IEEE International*. IEEE, 2017, pp. 5181–5184.

[52] Pedram Ghamisi, Yushi Chen, and Xiao Xiang Zhu, "A self-improving convolution neural network for the classification of hyperspectral data," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 10, pp. 1537–1541, 2016.

[53] Anirban Santara, Kaustubh Mani, Pranoot Hatwar, Ankit Singh, Ankur Garg, Kirti Padia, and Pabitra Mitra, "Bass net: band-adaptive spectral-spatial feature learning neural network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5293–5301, 2017.

[54] Volodymyr Mnih, "Machine learning for aerial image labeling," in *University of Toronto (Canada)*. 2013, Citeseer.

[55] Volodymyr Mnih and Geoffrey E Hinton, "Learning to detect roads in high-resolution aerial images," in *European Conference on Computer Vision*. Springer, 2010, pp. 210–223.

[56] Volodymyr Mnih and Geoffrey E Hinton, "Learning to label aerial images from noisy data," in *Proceedings of the 29th International conference on machine learning (ICML-12)*, 2012, pp. 567–574.

[57] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[58] Humayun Irshad, Alexandre Gouaillard, Ludovic Roux, and Daniel Racoceanu, "Multispectral band selection and spatial characterization: Application to mitosis

detection in breast cancer histopathology," *Computerized Medical Imaging and Graphics*, vol. 38, no. 5, pp. 390–402, 2014.

[59] Qian Du and Nicolas H Younan, "Dimensionality reduction and linear discriminant analysis for hyperspectral image classification," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer, 2008, pp. 392–399.

[60] Michael D Farrell and Russell M Mersereau, "On the impact of pca dimension reduction for hyperspectral detection of difficult targets," *IEEE Geoscience and Remote Sensing Letters*, vol. 2, no. 2, pp. 192–195, 2005.

[61] Xianghai Cao, Tao Xiong, and Licheng Jiao, "Supervised band selection using local spatial information for hyperspectral image," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 3, pp. 329–333, 2016.

[62] He Yang, Qian Du, Hongjun Su, and Yehua Sheng, "An efficient method for supervised hyperspectral band selection," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 1, pp. 138–142, 2011.

[63] Rupinder Kaur and Smriti Sehgal, "Dimension reduction of multispectral data using canonical analysis," *International Journal of Computer Applications*, vol. 975, pp. 8887.

[64] J. M. Sotoca, F. Pla, and J. S. Sanchez, "Band selection in multispectral images by minimization of dependent information," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 2, pp. 258–267, 2007.

[65] Mohsen Ghamary Asl, Mohammad Reza Mobasheri, and Barat Mojaradi, "Unsupervised feature selection using geometrical measures in prototype space for hyperspectral imagery," *IEEE transactions on geoscience and remote sensing*, vol. 52, no. 7, pp. 3774–3787, 2014.

[66] Qian Du and He Yang, "Similarity-based unsupervised band selection for hyperspectral image analysis," *IEEE Geoscience and Remote Sensing Letters*, vol. 5, no. 4, pp. 564–568, 2008.

[67] Guennadi Saiko and Andrei Betlen, "Optimization of band selection in multispectral and narrow-band imaging: An analytical approach," in *Oxygen Transport to Tissue XLI*, pp. 361–367. Springer, 2020.

[68] V Badrinarayanan, A Kendall, and R Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481, 2017.

[69] Min Bai and Raquel Urtasun, "Deep watershed transform for instance segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 2858–2866.

[70] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, 2017.

[71] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations (ICLR)*, 2015.

[72] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson, "How transferable are features in deep neural networks?," in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[73] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *International Conference on Learning Representations (ICLR)*, vol. 1050, pp. 20, 2015.

[74] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," *International Conference on Learning Representations (ICLR)*, 2017.

[75] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami, "The limitations of deep learning in adversarial settings," in *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*. IEEE, 2016, pp. 372–387.

[76] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman, "Towards the science of security and privacy in machine learning," *IEEE European Symposium on Security and Privacy (EuroSP)*, 2016.

[77] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1528–1540.

[78] John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani, "Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks," *arXiv preprint arXiv:1707.02476*, 2017.

[79] Ekin D Cubuk, Barret Zoph, Samuel S Schoenholz, and Quoc V Le, "Intriguing properties of adversarial examples," *International Conference on Learning Representations Workshop*, 2017.

[80] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, 2017, pp. 6393–6395.

[81] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 582–597.

[82] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," *stat*, vol. 1050, pp. 19, 2017.

[83] Anish Athalye, Nicholas Carlini, and David Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *International Conference on Machine Learning (ICML)*, 2018.

[84] Nicholas Carlini and David Wagner, "Defensive distillation is not robust to adversarial examples," *arXiv preprint arXiv:1607.04311*, 2016.

[85] Nicholas Carlini and David Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017, pp. 3–14.

[86] Nicholas Carlini and David Wagner, "Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[87] Shixiang Gu and Luca Rigazio, "Towards deep neural network architectures robust to adversarial examples," *International Conference on Learning Representations (ICLR) Workshop*, 2015.

[88] Alexander Toet, Jan Kees IJspeert, Allen M Waxman, and Mario Aguilar, "Fusion of visible and thermal imagery improves situational awareness," *Displays*, vol. 18, no. 2, pp. 85–95, 1997.

[89] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," *International Conference on Learning Representations Workshops*, 2017.

[90] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet, "Houdini: Fooling deep structured prediction models," *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[91] Volker Fischer, Mummadi Chaithanya Kumar, Jan Hendrik Metzen, and Thomas Brox, "Adversarial examples for semantic image segmentation," *International Conference on Learning Representations (ICLR) Workshop*, 2017.

[92] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer, "Universal adversarial perturbations against semantic image segmentation," *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[93] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille, "Adversarial examples for semantic segmentation and object detection," in *International Conference on Computer Vision. IEEE*, 2017.

[94] Anurag Arnab, Ondrej Miksik, and Philip HS Torr, "On the robustness of semantic segmentation models to adversarial attacks," *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[95] Nicholas Carlini and David Wagner, "Towards evaluating the robustness of neural networks," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017, pp. 39–57.

[96] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[97] Hidetoshi Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of statistical planning and inference*, vol. 90, no. 2, pp. 227–244, 2000.

[98] Xuezhi Wang and Jeff Schneider, "Flexible transfer learning under support and model shift," in *Advances in Neural Information Processing Systems*, 2014, pp. 1898–1906.

[99] Yeounoh Chung, Peter J Haas, Eli Upfal, and Tim Kraska, "Unknown examples & machine learning model generalization," *arXiv preprint arXiv:1808.08294*, 2018.

[100] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola, "Correcting sample selection bias by unlabeled data," in *Advances in neural information processing systems*, 2007, pp. 601–608.

[101] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola, "A kernel method for the two-sample-problem," in *Advances in neural information processing systems*, 2007, pp. 513–520.

[102] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio, "Feature-wise transformations," *Distill*, 2018, https://distill.pub/2018/feature-wise-transformations.

[103] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur, "A learned representation for artistic style," *International Conference on Learning Representations (ICLR)*, 2017.

[104] Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville, "Modulating early visual processing by language," in *Advances in Neural Information Processing Systems*, 2017, pp. 6594–6604.

[105] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville, "Film: Visual reasoning with a general conditioning layer," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[106] Spyros Gidaris, Praveer Singh, and Nikos Komodakis, "Unsupervised representation learning by predicting image rotations," *International Conference on Learning Representations (ICLR)*, 2018.

[107] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[108] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv preprint arXiv:1911.05722*, 2019.

[109] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[110] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *NIPS Deep Learning Workshop*, 2015.

[111] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio, "Fitnets: Hints for thin deep nets," *International Conference on Learning Representations (ICLR)*, 2015.

[112] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 535–541.

[113] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proceedings of the International Conference in Machine Learning (ICML)*, 2015.

[114] Yuxin Wu and Kaiming He, "Group normalization," in *European Conference on Computer Vision*. Springer, 2018, pp. 3–19.

[115] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[116] Bohao Huang, Kangkang Lu, Nicolas Audeberr, Andrew Khalel, Yuliya Tarabalka, Jordan Malof, Alexandre Boulch, Bertr Le Saux, Leslie Collins, Kyle Bradbury, et al., "Large-scale semantic classification: outcome of the first year of INRIA aerial image labeling benchmark," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 6947–6950.

[117] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry, "How does batch normalization help optimization?," in *Advances in Neural Information Processing Systems*, 2018, pp. 2483–2493.

[118] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi, "Learning multiple visual domains with residual adapters," in *Advances in Neural Information Processing Systems*, 2017, pp. 506–516.

[119] Siwei Lyu and Eero P Simoncelli, "Nonlinear image representation using divisive normalization," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[120] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al., "What is the best multi-stage architecture for object recognition?," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 2146–2153.

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

[122] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, Software available from tensorflow.org.

[123] Paul Viola, Michael Jones, et al., "Rapid object detection using a boosted cascade of simple features," in *2001 IEEE conference on computer vision and pattern recognition*. IEEE, 2001.

[124] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," *IEEE conference on computer vision and pattern recognition*, 2016.

[125] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[126] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang, "High-resolution representations for labeling pixels and regions," *CoRR*, vol. abs/1904.04514, 2019.

[127] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft COCO: Common

objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[128] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask R-CNN," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[129] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proceedings of the IEEE Conference on Computer Cisionand Pattern Recognition (CVPR)*, 2009.

[130] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Cision and Pattern Recognition*, 2016, pp. 770–778.

[131] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[132] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[133] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He, "Accurate, large minibatch SGD: Training Imagenet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[134] Chesapeake Land Cover on Lila, "Chesapeake land cover," `http://lila.science/datasets/chesapeakelandcover`, Maryland split.

[135] James Y Zou, Daniel J Hsu, David C Parkes, and Ryan P Adams, "Contrastive learning using spectral methods," in *Advances in Neural Information Processing Systems*, 2013, pp. 2238–2246.

Vita

Anthony Ortiz received a Bachelor of Science in Telematics Engineering (Electrical and Computing Engineering equivalent) at the Pontificia Universidad Catolica Madre y Maestra, Dominican Republic with Summa Cum Laude distinction. He is pursuing his Ph.D. in Computer Science at The University of Texas at El Paso under the supervision of Dr. Olac Fuentes and Dr. Christopher Kiekintveld. His research interest lies in the intersection between machine learning, computer vision, and remote sensing. He is particularly interested in the application of artificial intelligence to solve problems affecting society. He is involved with the Climate Change AI initiative, is an active member of LatinX in AI, and is leading, along Kris Sankaran, a joint effort between Mila, UTEP, and ICIMOD on improving Glacier Mapping Monitoring in the Hindu Kush-Himalayan region with the support of the Microsoft AI for Earth initiative.

He spent several months at Mila (Quebec Artificial Intelligence Institute) in Montreal working under the guidance of professor Yoshua Bengio and Kris Sankaran. He spent time at Microsoft Research as an AI Research Intern working with Nebojsa Jojic and Dan Morris. He also worked as visiting research scientist for the US Army Research Laboratory (ARL), the University of Southern California (USC), and Orbital Insight. He is recipient of multiple awards and fellowships including the "Diploma de Maxima Excelencia" from the president of the Dominican Republic, Municipal Youth Award, the Good Neighbor Scholarship from the Texas Department of Higher Education (twice), and the Anita Mochen Loya College of Engineering Graduate Fellowship, among others. His work has been published in top AI and remote sensing conferences including IGARSS, AAAI, NeurIPS, ECCV, and CVPR.