

# Scale-invariant Image Segmentation using Machine Learning

By

Rasheed Andrews

A Thesis Submitted to  
Saint Mary's University, Halifax, Nova Scotia  
in Partial Fulfillment of the Requirements for  
the Degree of Master of Science in Computing & Data Analytics

December 18<sup>th</sup>, 2018, Halifax, Nova Scotia

© Rasheed Andrews, 2018

Approved: Dr. Jason Rhinelanders  
Supervisor  
Division of Engineering

Approved: Dr. Mohamed Shehata  
External Examiner  
Engineering and Applied Science  
Memorial University

Approved: Dr. Pawan Lingras  
Committee Member  
Department of Mathematics  
and Computing Science

Approved: Dr. Yasushi Akiyama  
Committee Member  
Department of Mathematics  
and Computing Science

Date: December 18<sup>th</sup>, 2018

# Acknowledgments

The QE2 Scholarship funded the work presented in this thesis, and I want to show my utmost gratitude for providing me with the opportunity to pursue my dreams and make a change for my country, Jamaica.

I want to thank my supervisor, Dr. Jason Rhineland from the Division of Engineering for his mentoring, expertise, motivation and patience as he met with me on a consistent basis to offer his guidance. His influence was an integral part in the completion of this thesis. To my committee members, Dr. Lingras, Dr. Akiyama and external examiner, Dr. Mohamed Shehata, I am honored that you have decided to be a part of my research.

Special thanks to my fiancée, Chantal Stewart for proofreading and assisting me with the creation of my diagrams. I am genuinely grateful that I had my parents support and encouragement to complete this mission.

My International student representative Alida Campbell, thank you for making this journey possible, for checking up on me and ensuring that everything was on track, you are a real guardian angel.

Finally, I want to thank God for health, strength and all the persons he placed in my life that guided me throughout the completion of this milestone.

# **Abstract**

## **Scale-invariant Image Segmentation using Machine Learning**

by Rasheed Andrews

The increased application of segmentation requires more robust machine learning algorithms that can handle variations of the input. The areas of robotics, self-driving cars, automated drone delivery systems, and Speed Enforcement Cameras (SEC) all rely on accurate predictions from machine learning algorithms. CNNs are the current state of the art in image recognition which has led to their increased application in areas of classification, object detection, and segmentation.

However, scale invariance poses a significant issue for CNNs; fixed kernel size hinders the prediction accuracy of the network for objects of varying sizes. This research introduces a training methodology and image pre-processing techniques which makes these networks more robust or invariant to the changes in the size of the object.

December 18<sup>th</sup>, 2018

## Table of Contents

<b>Chapter 1</b> .....	1
1.1 Applications of Image Segmentation .....	1
1.2 Thesis Objectives .....	4
1.3 Organization of thesis .....	5
<b>Chapter 2</b> .....	6
2.1 Segmentation using Machine Learning .....	6
2.2 Artificial Neural Network .....	8
2.2.1 Multi-Layer Perceptron .....	10
2.3 Deep Learning .....	11
2.4 Convolutional Neural Network .....	13
2.4.1 Convolution .....	13
2.4.2 Pooling .....	15
2.4.3 Local Connectivity .....	16
2.4.4 Shared Parameters – Weight Sharing .....	16
2.5 Convolutional Neural Network for Localization .....	17
2.5.1 Classification .....	18
2.5.2 Object Localization .....	18
2.5.3 Segmentation .....	19
2.6 Fully Convolutional Neural Network for Segmentation .....	20
2.6.1 Transposed Convolution (Deconvolution) .....	21
2.6.2 Skip Connections (Layer Fusing) .....	22
2.7 U-Net: Convolutional Networks for Biomedical Image Segmentation .....	23
2.8 Techniques to overcome scale invariance with CNNs .....	25
2.9 Data from Remote Sensing .....	28
2.9.1 Spatial Resolution .....	29
2.10 Summary of Chapter 2 .....	30
<b>Chapter 3</b> .....	31
3.1 Overview of Dataset .....	32
3.1.1 Data Preparation .....	35
3.1.2 Training Procedure .....	37
3.1.3 Result Evaluation .....	39

3.1.3.1	IOU (Intersection-Over-Union)	40
3.2	Experiment applications and software	41
3.2.1	Loss Function	41
3.2.2	Optimizer	42
3.2.3	Software and Tools	42
3.3	Problem Scope	43
3.3.1	Algorithm Use Case	45
3.4	Approach to problem	47
3.4.1	Zoom Augmentation – Image Scaling	49
3.5	Summary of Chapter 3	52
<b>Chapter 4</b>		<b>53</b>
4.1	Training on One Spatial Resolution	53
4.1.1	Training on the Highest Spatial Resolution	54
4.1.2	Training on the Lowest Spatial Resolution	58
4.2	Training on Zoom Augmented (ZA) Image	63
4.2.1	Zoom Augmentation Experiment for Pond Dataset	66
4.2.2	Zoom Augmentation Experiment for Door Dataset	68
4.3	Issues with Zoom Augmentation	70
4.4	Waterfall Method	71
4.4.1	Waterfall Experiments	73
4.4.2	Ensemble Learning - Bootstrap Aggregation (Bagging)	77
4.5	Summary of Chapter 4	78
<b>Chapter 5</b>		<b>79</b>
5.1	Waterfall Method vs Zoom Augmentation	79
5.1.1	ZA vs WM Experiment for Pond Dataset	79
5.1.2	ZA vs WM Experiment for Door Dataset	82
5.2	Issues with the Waterfall Method	84
5.2.1	Sobel Filter for all Scales	86
5.2.2	No Sobel Filter for Coarsest scale	88
5.2.3	Contrast Enhancement using Histogram Equalization	88
5.3	ZA vs WM Experiment for Grass Dataset	92
5.4	Summary of Chapter 5	95
<b>Chapter 6</b>		<b>97</b>

6.1	Summary and Conclusion .....	97
6.2	Future Directions .....	98
	<b>References</b> .....	100
	<b>Appendix</b> .....	106
8.1	Appendix A – Experiment Images .....	106
8.2	Appendix B - Future work with drone images .....	109
8.3	Appendix C – Padding Methods .....	112
8.4	Appendix D – Overview of Training Procedure .....	114
8.5	Appendix E – Image Processing Results .....	115
8.5.1	Pond Dataset .....	115
8.5.2	Door Dataset .....	117
8.5.3	Grass Dataset .....	119
8.6	Appendix F – Optimizer Results .....	121
8.7	Appendix G – Training time Results .....	122
8.8	Appendix H – Confidence Level Results for Segmentation .....	123

## List of Figures

Figure 1-1:	Actual crowd count vs estimated crowd count. ....	2
Figure 2-1:	Segmentation of whiteflies .....	7
Figure 2-2:	An illustration of a single layer neural network or perceptron. ....	9
Figure 2-3:	Linear and Non-Linear Decision Plane .....	9
Figure 2-4:	Layers from a Deep Neural Network. ....	11
Figure 2-5:	An illustration of convolutional filters producing feature maps. ....	14
Figure 2-6:	Feature map before and after max pooling .....	15
Figure 2-7:	Evolution from coarse-grained to fine-grained inference. ....	17
Figure 2-8:	Transforming classification networks into segmentation networks .....	20
Figure 2-9:	An illustration of upsampling feature maps using transposed convolution. ..	21
Figure 2-10:	Skip Connections between fine and coarse feature maps. ....	22

Figure 2-11: A diagram of U-Net. ....	24
Figure 2-12: Scale-invariant CNN ensemble for classification. ....	26
Figure 2-13: Multi-Scale Image Pyramid. ....	27
Figure 2-14: An illustration of reflection captured by aerial devices. ....	28
Figure 2-15: Difference between fine and coarse spatial resolution.....	29
Figure 3-1: Images from the highest spatial resolution for each ROI.....	34
Figure 3-2: A diagram showcasing the structure of the dataset.....	35
Figure 3-3: Formula for normalizing pixel values.....	37
Figure 3-4: Thresholding of probability values to produce a binary mask.....	39
Figure 3-5: Formula for calculating pixel accuracy for segmentation.....	40
Figure 3-6: Formula for calculating Intersection-Over-Union (IOU).....	40
Figure 3-7: Comparison between examples of overlaps and corresponding IOU score...	41
Figure 3-8: Loss or Cost function to be optimized. ....	41
Figure 3-9: Difference between fine and coarse spatial resolution.....	44
Figure 3-10: An illustration of a farmer capturing images on-ground and identification performed from an aerial device. ....	46
Figure 3-11: Difference between late and early detection. ....	47
Figure 3-12: Process of up and downscaling an image.....	48
Figure 3-13: Pixel count for the region of interest at scale one and two. ....	50
Figure 3-14: Pixel count for the region of interest after downscaling the image.....	51
Figure 4-1: Zoom Augmentation process. ....	64
Figure 4-2: An illustration of the Waterfall Method.....	72
Figure 4-3: An ensemble of SVMs. ....	77

Figure 5-1: Comparison of the IOU across all scales (2, 3, 4) between ZA and WM.....	81
Figure 5-2: Comparison of the IOU across all scales (2, 3) between ZA and WM.....	83
Figure 5-3: Training image used to train scale 3 of WM.....	84
Figure 5-4: Histogram equalized image used to train scale 2.....	89
Figure 5-5: Histogram equalized image used to train scale 3.....	90
Figure 5-6: Comparison of the IOU across all scales (2, 3) between ZA and WM.....	91
Figure 5-7: Comparison of the IOU across all scales (4, 8) between ZA and WM.....	93
Figure 5-8: Predictions from using ZA versus WM.....	94
Figure 8-1: Training data for all scales from the pond dataset.....	106
Figure 8-2: Training data for all scales from the door dataset.....	107
Figure 8-3: Training data for all scales from the grass dataset.....	108
Figure 8-4: Images from the first drone footage.....	109
Figure 8-5: Images from the second drone footage.....	110
Figure 8-6: Images from the third drone footage.....	111
Figure 8-7: Shows four padding methods used after ZA.....	113

## List of Tables

Table 4-1: Network was trained on the highest scale for the pond dataset.....	54
Table 4-2: Network was trained on the highest scale for the door dataset.....	55
Table 4-3: Network was trained on the highest scale for the window dataset.....	56
Table 4-4: Network was trained on the highest scale for the wall dataset.....	57
Table 4-5: Network was trained on the lowest scale for the pond dataset.....	58
Table 4-6: Network was trained on the lowest scale for the door dataset.....	59



Table 4-7: Network was trained on the lowest scale for the window dataset. ....	60
Table 4-8: Network was trained on the lowest scale for the wall dataset. ....	61
Table 4-9: Zoom Augmented image with mean padding and corresponding mask. ....	65
Table 4-10: Pixel (px) count for ROI at each scale.....	66
Table 4-11: Predictions from three networks trained on ZA image. ....	67
Table 4-12: Pixel (px) count for ROI at each scale.....	68
Table 4-13: Predictions from two networks trained on ZA image. ....	69
Table 4-14: An illustration of the loss in contextual information when using ZA. ....	70
Table 4-15: Predictions from each scale in WM.....	74
Table 4-16: Predictions from each scale in WM.....	75
Table 5-1: Network was trained on only the highest spatial resolution for the pond dataset using ZA.....	80
Table 5-2: Network was trained on the highest spatial resolution for the pond dataset using WM.....	80
Table 5-3: Network was trained on only the highest spatial resolution for door using ZA. .....	82
Table 5-4: Network was trained on only the highest spatial resolution for door using WM. .....	82
Table 5-5: Network was trained on predictions from scale 2 using WM. ....	85
Table 5-6: Network was trained on the highest spatial resolution for door using WM with Sobel Filter applied. ....	86
Table 5-7: Sobel filtered door for scale 2 and 3.....	87

Table 5-8: Network was trained on the highest spatial resolution for door using WM, no Sobel filter was used at scale 3. ....	88
Table 5-9: Network was trained on the highest spatial resolution for door using WM with HE. ....	89
Table 5-10: Network was trained on the highest spatial resolution for the grass dataset using ZA.....	92
Table 5-11: Network was trained on the highest spatial resolution for the grass dataset using WM.....	92
Table 5-12: Summary table showing average IOU score for each method across all datasets.....	95
Table 8-1: IOU scores from four padding methods used after ZA .....	113
Table 8-2: Shows the training parameters used for experiments.....	114
Table 8-3: Results from increasing training size using data augmentation. ....	114
Table 8-4: Results from using no data augmentation on the Pond Dataset. ....	115
Table 8-5: Results from using ZA on the Pond Dataset. ....	115
Table 8-6: Results from using WM on the Pond Dataset. ....	116
Table 8-7: Results from using WM with HE on the Pond Dataset. ....	116
Table 8-8: Results from using no data augmentation on the Door Dataset. ....	117
Table 8-9: Results from using ZA on the Door Dataset. ....	117
Table 8-10: Results from using WM on the Door Dataset. ....	118
Table 8-11: Results from using WM with HE on the Door Dataset. ....	118
Table 8-12: Results from using no data augmentation on the Grass Dataset. ....	119
Table 8-13: Results from using ZA on the Grass Dataset. ....	119

Table 8-14: Results from using WM on the Grass Dataset. ....	120
Table 8-15: Results from using WM with HE on the Grass Dataset. ....	120
Table 8-16: Results from scale 2 of the Door Dataset using several optimizers. ....	121
Table 8-17: Running Time and Final epoch from the Pond Dataset. ....	122
Table 8-18: Running Time and Final epoch from the Door Dataset. ....	122
Table 8-19: Running Time and Final epoch from the Grass Dataset.....	122
Table 8-20: Results from different Confidence levels on the Pond Dataset. ....	123
Table 8-21: Results from different Confidence levels on the Door Dataset. ....	123
Table 8-22: Results from different Confidence levels on the Grass Dataset. ....	124

## List of Acronyms

<b>ANN</b>	Artificial Neural Network
<b>API</b>	Application Programming Interface
<b>C</b>	Number of Channels or Classes
<b>CNN</b>	Convolutional Neural Network
<b>FC</b>	Fully Connected
<b>FCM</b>	Fuzzy c-means
<b>FCN</b>	Fully Convolutional Network
<b>FN</b>	False Negative
<b>FP</b>	False Positive
<b>GPU</b>	Graphics Processing Unit
<b>H</b>	Height
<b>HE</b>	Histogram Equalization
<i>i</i>	Row Index from Matrix
<b>ILSVRC</b>	ImageNet Large Scale Visual Recognition Challenge
<b>IOU</b>	Intersection Over Union
<i>j</i>	Column Index from Matrix
<b>LIoU</b>	Loss for IOU
<b>ML</b>	Machine Learning
<b>MLP</b>	Multi-Layer Perceptron
<b>MNIST</b>	Modified National Institute of Standards and Technology
<b>PA</b>	Pixel Accuracy
<b>RGB</b>	Red, Green, and Blue
<b>ROI</b>	Region of Interest
<b>SB</b>	Sobel Filtering
<b>SGD</b>	Stochastic Gradient Descent
<b>SVM</b>	Support Vector Machine
<b>TN</b>	True Negative
<b>TP</b>	True Positive
<b>W</b>	Width
<b>WM</b>	Waterfall Method
<b>X</b>	Input image.
<b>Y</b>	Label (Binary Mask)
<b>ZA</b>	Zoom Augmentation

# Chapter 1

## Introduction

Segmentation is the task of partitioning a digital image into several regions. Each pixel in the image is assigned a class label, thus allowing us to highlight or segment each region. An image can be segmented using simple thresholding or more complex machine learning algorithms such as clustering. The current state of the art for recognizing objects in images are Convolutional Neural Networks (CNNs). Several variations of CNNs have won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) which makes these networks a viable option for performing image recognition tasks such as segmentation.

### 1.1 Applications of Image Segmentation

The importance of performing accurate segmentation is a core computer vision problem, which is highlighted by the increasing number of applications (Garcia et al., 2017). Applications include autonomous driving, robotics, augmented reality and agriculture. The areas aforementioned rely on improved segmentation to provide a more reliable and safer experience to the user. Segmentation is regarded as a high-level task since it generates predictions of semantic labels at the pixel level which provides more information and an increased understanding of the regions in the image.

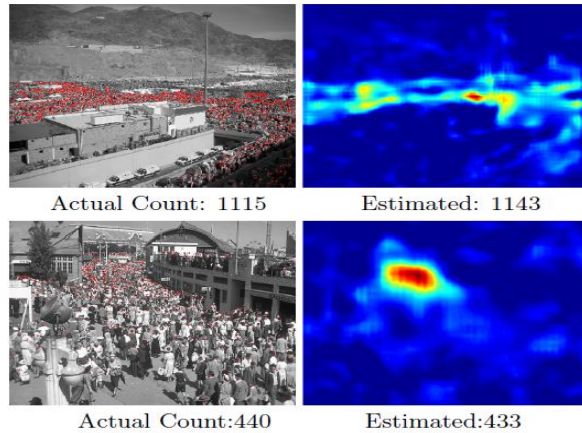


Figure 0-1: Actual crowd count vs estimated crowd count. Retrieved from “CrowdNet: A Deep Convolutional Network for Dense Crowd Counting.” (Boominathan, Kruthiventi & Babu, 2016, p.1).

From Figure 1-1, a CNN was used by Boominathan et al., to perform segmentation (Boominathan et al., 2016). The segmented image is used to determine the number of persons in the crowd, as well as the crowd density. The researchers mention that detecting persons at varying sizes posed a significant challenge; however, the issue can be effectively tackled by using data augmentation techniques.

Although image segmentation can be performed using various types of machine learning algorithms, the advancement of CNNs in the areas of image classification and segmentation have led to the creation of more accurate computer vision systems. The prediction accuracy of CNNs is on par with that of a human, which explains why there is an increased usage of CNNs in areas requiring human-like accuracy such as self-driving cars. Companies such as Mobileye and NVIDIA are using such CNNs in their upcoming vision systems for self-driving cars (LeCun, Bengio, & Hinton, 2015).

Another area of application for image segmentation is agriculture. The presence of pest has been a significant issue in agriculture, in which late detection results in a significant loss in agricultural production and excessive use of pesticides. To facilitate the early detection of pest, different machine learning algorithms can be used to segment pest infested regions from digital images. Figure 2-1 shows segmentation of whiteflies utilizing a machine learning algorithm called clustering.

Agriculture is one of several areas that can benefit from an improved segmentation architecture. An improved architecture will facilitate more accurate and faster detection of pest outbreaks, which allows for interventions to be made as soon as possible. The progression of CNNs in the area of image recognition make them a favourable option for providing improved image segmentation.

## 1.2 Thesis Objectives

Computers have been used to automate tedious tasks, for (e.g., factory jobs). Generally, machines complete these tasks much faster than humans. Although machines excel in their speed, accuracy has been a major issue. Previously, knowledge and instructions for a computer would have to be programmed or prewritten, but due to the significant amount of variation in the world in terms of lighting, orientation or size of the object, this makes it impossible to program all possible scenarios.

The rise of Machine Learning (ML) has made computers more accurate or “smarter” making them capable of handling more complex task. This has led to an increase of applications for ML algorithms, especially in areas that require human-like accuracy such as self-driving cars and robotics. However, the development of these areas is dependent on the improvement of existing ML algorithms.

Convolutional Neural Networks have shown tremendous success in the area of image recognition; however, multi-scale recognition continues to be a challenge. There is an increased utilization of CNNs in the area of image recognition such as: classification and segmentation. Therefore, the improvement of segmentation will depend on the performance of these networks. In this study, an improved image segmentation technique will be suggested which allow these networks to be scale-invariant.



### **1.3 Organization of thesis**

The focus of this research is to develop an improved technique for image segmentation. Chapter 2 provides a review of the literature, showcasing different methods used by researchers for segmentation, these include Support Vector Machine (SVM), Clustering and Artificial Neural Networks. Chapter 3 contains the data set, the experimental design, and the evaluation metrics. Chapter 4 demonstrates a significant problem with the machine learning algorithm of choice that hinders segmentation accuracy; therefore, several methods were tested with the aim of alleviating the problem. These methods will be referred to as Zoom Augmentation (ZA) and the Waterfall Method (WM). Chapter 5 includes a quantitative comparison between ZA and WM. Each method was evaluated across several datasets using the Intersection-Over-Union (IOU) metric. Finally, Chapter 6 concludes that WM was the superior image scaling method compared to ZA. Also, including image processing techniques such as histogram equalization is essential for improving prediction accuracy. Future directions for this research are mentioned in Chapter 6.

# Chapter 2

## Literature Review

In computer vision, image segmentation is one of the oldest and most widely studied problems (Szeliski, 2010). Many applications arise from the need for more accurate segmentation algorithms. This rise coincides with the increase in deep learning; researchers are now using deep convolutional neural networks to achieve human-like accuracy for the task of segmentation. Before the upsurge in deep learning, machine learning algorithms such as clustering and Support Vector Machines (SVM) have been used to segment digital images.

### 2.1 Segmentation using Machine Learning

The field of Machine Learning (ML) is concerned with the question of how to construct computer programs that automatically improve with experience (Mitchell, 1997). In recent years, several machine learning algorithms have been implemented including, but not limited to clustering, SVMs, and ANN.

The goal of clustering is to divide ( $n$ ) objects into ( $k$ ) clusters such that each cluster has maximum similarity (homogeneity) among its members and maximizes the difference (heterogeneity) between clusters (Pratheba et al., 2014). Pixel values in an image that are similar to each other will form clusters, the center value (centroid) of these clusters can then be used to segment the image, highlighting the object of interest. Clustering has been used by researchers for recognition as well as counting of pest in agriculture.

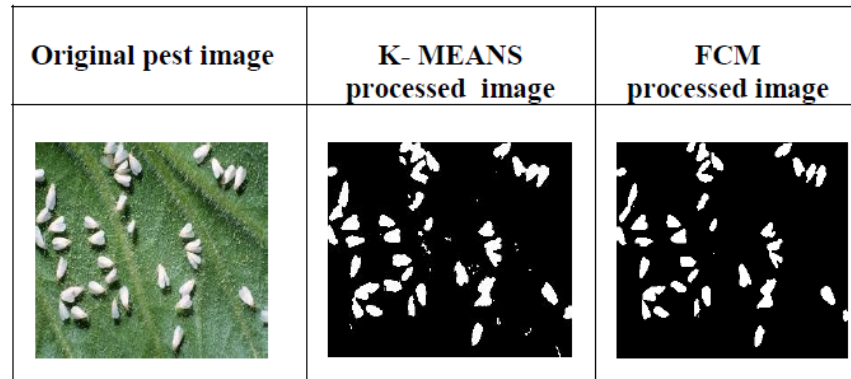


Figure 2-1: Segmentation of whiteflies. Retrieved from “Performance analysis of pest detection for agricultural field using clustering techniques.” (Pratheba, Sivasangari & Saraswady, 2014, p.2).

From Figure 2-1, a comparison between clustering algorithms k-means and fuzzy c-means was performed by Pratheba et al., for segmenting whiteflies (Pratheba et al., 2014). The results showed that k-means has a faster execution time, but it is less accurate than fuzzy c-means. However, segmenting an image using clustering solely relies on color information; therefore objects of similar color will often be misclassified.

SVMs are binary classifiers that separate data points into distinct groups or classes using a decision boundary. They also require the selection of special kernel functions for the extraction of features. Segmentation using SVMs was performed by Sakthivel (Sakthivel et al., 2015); However, they relied on an additional algorithm, fuzzy c-means (FCM) which extracts color and texture-based features that will be used to train the SVM. The researchers mentioned that a drawback of their proposed segmentation architecture is the lack of robustness towards noise – variations of the input data. In Rajan et al., research, SVMs were used to classify various agricultural pest (Rajan et al., 2016). The results indicated that automatic pest identification using SVM is inferior to manual

identification. This was due to the variation in lighting, which affected the color of the pest in the image. However, Rajan et al., stated that the accuracy of the model could be improved by applying image processing techniques (Rajan et al., 2016). A significant drawback to SVMs is that features relevant to the task at hand must be manually extracted from images before training, while more recent machine learning models such as Convolutional Neural Networks provide automatic feature extraction.

Although Artificial Neural Networks (ANNs) have existed for decades, recent resurgence as refinements in existing techniques, newer hardware and the growth of big data has created a boom in the field of artificial intelligence that shows no sign of slowing down (Lemley et al., 2017). This had led to several advancements in computer vision such as classification, object detection and segmentation using ANN.

## **2.2 Artificial Neural Network**

An artificial neural network (ANN) models the function of the biological neural network inside the human brain, regarding structure (interconnected neurons) and information flow. Neurons fire when they have exceeded a certain threshold based on a combination of inputs from previous neurons. The first type of ANN is called the *perceptron* which is a single layer neural network with a single output layer.

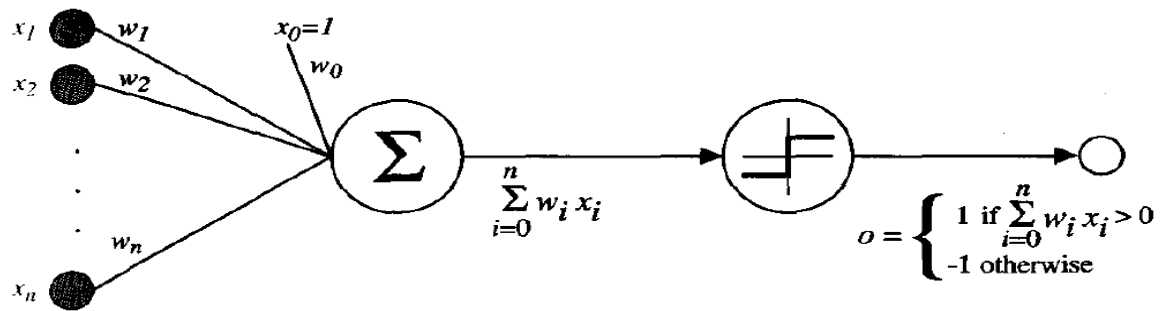


Figure 2-2: An illustration of a single layer neural network or perceptron. Retrieved from “Machine Learning.” (Mitchell & Tom, 1997, p. 87).

From Figure 2-2, the network takes as input a vector of real values ( $x_1 \dots x_n$ ), calculates a linear combination of these inputs ( $\sum w \cdot x$ ) by a single processing unit called a *neuron*, it then passes the result to an activation function, in this example sigmoid is used, which outputs a 1 if the result is greater than some threshold else -1 (Mitchell & Tom, 1997). The perceptron is a linear classifier as it can only achieve 100% accuracy on data that is linearly separable.

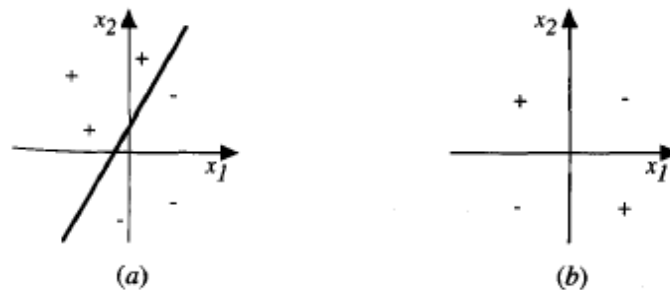


Figure 2-3: Linear and Non-Linear Decision Plane. Retrieved from “Machine Learning.” (Mitchell & Tom, 1997, p. 87).

The famous XOR problem shown in Figure 2-3, highlighted the perceptron's inability to separate a nonlinear decision plane(b), this problem can be circumvented by introducing "hidden-layers" between the input layer and output layer – this type of ANN is called a Multi-Layer Perceptron (MLP).

### **2.2.1 Multi-Layer Perceptron**

The perceptron can only form linear decision boundaries; however, MLPs can express a variety of non-linear decision boundaries (Mitchell & Tom, 1997). Instead of having a single layer which takes an input and produces an output, MLPs consists of stacks of layers called "hidden layers" which receives input from previous layers and produces an output, which will be the input to subsequent layers. Adding of hidden layers allows us to distort the input in a nonlinear way so that categories become linearly separable by the last layer in the network (LeCun, Bengio, & Hinton, 2015).

A multilayer perceptron can contain a single hidden layer or multiple hidden layers. This presents a thin line when discriminating between a MLP and Deep neural network, since there is no written definition as to how many layers make a neural network a deep one, however an ANN with a single layer is referred to as a shallow net, so researchers loosely define deep nets as having two or more hidden layers. Furthermore, the quintessential example of a deep learning model is the multilayer perceptron (Goodfellow, Bengio, & Courville, 2016). Deep learning uses the classical MLP architecture, but also includes novel approaches to automate feature engineering.

## 2.3 Deep Learning

Conventional machine learning algorithms were limited in their ability to process raw data and required a domain expert to carefully engineer feature extractors which would transform the structure of the input data into a suitable internal representation for the detection of patterns and classification of objects (LeCun, Bengio, & Hinton, 2015).

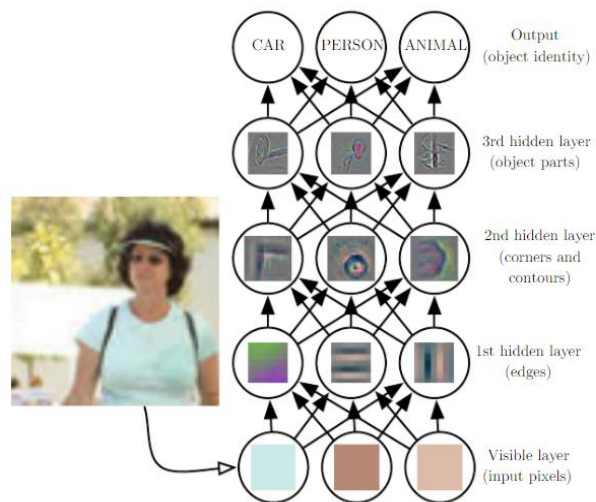


Figure 2-4: Layers from a Deep Neural Network. Retrieved from “Deep Learning.”

(Goodfellow, Bengio, & Courville, 2016, p. 6)

A deep learning network consists of multiple layers of representations (an  $n$ -dimensional feature vector describing an object) derived by nonlinear operations, where the preceding layers are used to construct a more higher-level abstraction which helps to distinguish and classify objects. In Figure 2-4, a hierarchical representation is learnt where the lowest level learns to detect edges, these combine to form corners and contours in the next layer, then parts of the objects called motifs, and finally, all these features or representations correlate to create a higher level of abstraction such as a car, person or animal.

A key point to note is that deep learning is not tied to neural networks, but it is instead demonstrated using neural networks. Neural networks and their training methods have existed for decades. However, since recently there has been a resurgence in the field due to factors such as refinements in machine learning algorithms and training methods, the arrival of GPUs (Graphics Processing Unit) which allows for faster training of the network, and access to large datasets (Lemley et al., 2017). Deep learning tools are no longer confined to researchers but are now integrated with consumer applications such as Google's assistant and Amazon's Alexa, which enables the technology to reach a wider audience.

The introduction of deep learning has resulted in a significant improvement in the areas of speech recognition and computer vision. Multiple layers of neurons stacked on top of each other allow the network to extract relevant features and build a layer by layer representation of the given input space.



## **2.4 Convolutional Neural Network**

Convolutional Neural Networks (CNNs) are analogous to traditional ANNs in that they are comprised of neurons that self-optimize through learning (O'Shea, & Nash, 2015). Training of CNNs existed from the early 1990s, but due to the small size of labeled image datasets available at the time and long training times, they were not as powerful as they are today. It was not until 2012 when Alex Krizhevsky introduced deep convolutional networks which had more layers and more training data provided by ImageNet. The network utilized the GPU to decrease training time and included a regularization method to prevent overfitting, called dropout. This method achieved the new state of the art in image classification and from then, a significant amount of papers has been published on CNNs. The most notable difference between ANNs and CNNs is that CNNs are primarily used in the field of pattern recognition within images (O'Shea, & Nash, 2015). CNNs are a family of multi-layered neural networks and is generally composed of three main parts, convolution, pooling and fully connected layers (Amara et al., 2017).

### **2.4.1 Convolution**

The first layer in a CNN is a convolutional layer which contains a set of learnable filters (also called kernel filters) that convolve (move) around different subregions in an image, searching for features. Each filter is applied to the raw pixel values in the image across all channels (red, green, blue) in a sliding window fashion, calculating the dot product between the filter values ( $W$ ) and pixel values ( $x$ ) (Amara et al., 2017).

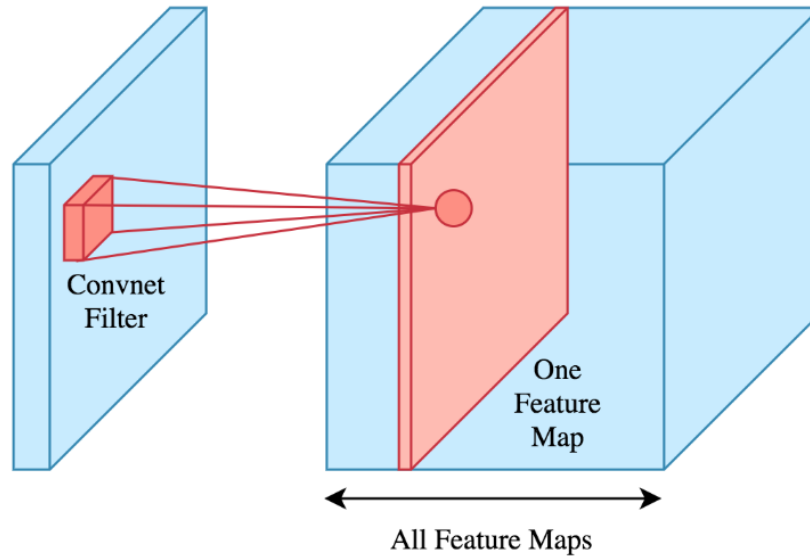


Figure 2-5: An illustration of convolutional filters producing feature maps. Retrieved June 26, 2018, from <https://brilliant.org/wiki/convolutional-neural-network>.

From Figure 2-5, each filter outputs an *activation* or *feature map*, which indicates what features are more pronounced at different spatial locations in the image. There are four hyperparameters to consider in the convolution layer of a CNN, these are: 1) number of filters, 2) size of filter – spatial extent, 3) stride and 4) padding. Each of these parameters determines the number of weights in the network. The size of the filter is known as the receptive field, which is a local patch in the image that the filter is applied to, while stride determines how many units of pixel the filter should be shifted across the image (Maggiore et al., 2016). A 3x3 filter with a stride of one will move one pixel across the input; a smaller stride outputs a larger feature map while a larger stride produces a smaller feature map.

## 2.4.2 Pooling

After the convolutional process, each activation map is passed to a pooling operation which reduces the dimensionality of the image (width and height), also called downsampling. The role of pooling is to merge semantically similar features into one, because the relative position of features that form an object (called motifs) may vary (LeCun, Bengio, & Hinton, 2015) therefore, discarding the finer details allows the network to make a general assumption as to what features are present in that region. This helps to reduce overfitting and allows the network to be invariant to small shifts and distortions. Pooling also reduces the number of parameters, thereby increasing the computational efficiency.

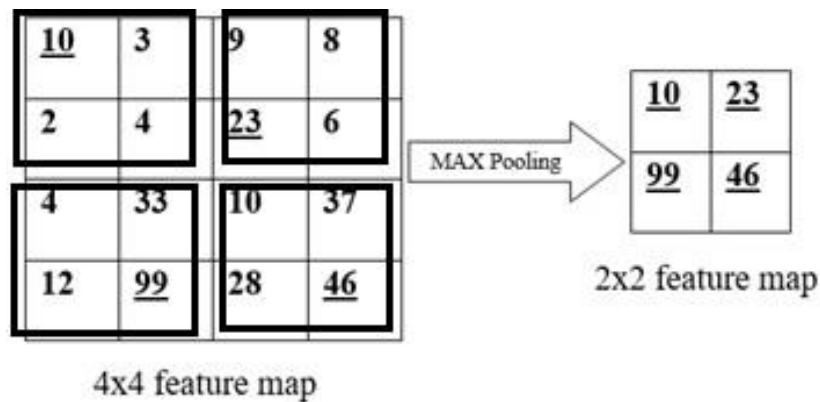


Figure 2-6: Feature map before and after max pooling

There are different types of pooling operation such as max, average, and L2-pooling, with the most common one being max pooling. Max pooling takes the largest value from each subregion after the convolutional operation as shown in Figure 2-6. The output from max pooling is then passed to an activation function such as Rectified Linear Unit (ReLU). CNNs are more than neural networks with convolution and pooling layers.

There are a few characteristics about CNNs that make them superior to the traditional neural networks, especially in the realm of image classification.

### **2.4.3 Local Connectivity**

Neural networks are fully connected, that is, each neuron in a layer is connected to all neurons in the previous layer. The famous benchmark dataset of handwritten digits - MNIST, has a small image dimensionality of  $28 \times 28$ , which means the first hidden layer will contain 784 weight values ( $28 \times 28 \times 1$ ). However, this is merely a black and white image. Considering a color image which is usually  $64 \times 64$  with three additional channels for RGB (red, green and blue) then we would have 12,288 weights (O'Shea, & Nash, 2015).

Connecting to all neurons results in an exponential growth in weight values which would take a long time to update during the backpropagation algorithm. Using a filter size of  $6 \times 6$  across three channels would give us a total of 108 weights compared to 12,288 weights using an ANN that is fully connected. Connecting to local patches (a group of pixels) is not only more computationally efficient but allows the CNN to take into consideration the relationship between pixels in close proximity, as pixels close in space are more correlated than pixels far apart.

### **2.4.4 Shared Parameters – Weight Sharing**

Neurons (filters) share the same weights, allowing the filter to detect the same pattern at different locations in the image – spatially invariant. Sharing the same weight values results in less learnable weight parameters.

A deep network named AlexNet was trained on ImageNet Large Scale Visual Recognition Challenge (ILSVRC) for 2012 in which it outperformed all other methods used for large scale image classification and rekindled interest in CNNs (Triantafyllidou & Tefas, 2016). The immense success in image classification has fueled breakthroughs in other areas of computer vision such as object recognition and segmentation.

## 2.5 Convolutional Neural Network for Localization

CNNs have performed exceptionally well in image classification – that is assigning a single label to the entire image after prediction. Another application of convolutional networks, which has seen a significant amount of progress in the last five years, is the task of localization; we should not only identify *what* is in the image, but also the *positioning* of the object within the image.

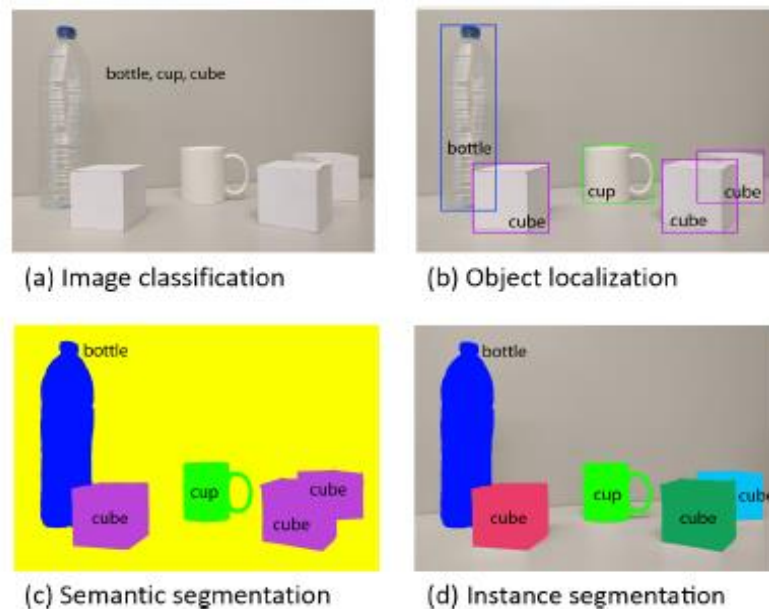


Figure 2-7: Evolution from coarse-grained to fine-grained inference. Retrieved from “A Review on Deep Learning Techniques Applied to Semantic Segmentation.” (Garcia et al., 2017, p. 1)

### **2.5.1 Classification**

Classification, also referred to as coarse-grain inference is assigning a single label based on the most pronounced object in the image (Sermanet et al., 2013). The state of the art networks such as AlexNet (Krizhevsky et al., 2012), VGG (Simonyan & Zisserman, 2014), GoogleNet (Zhong et al., 2015), and ResNet (He et al., 2016) were all in classification. The main issue with classification is that it requires the object of interest to be more pronounced (occupy a majority of the image) and centered to get an accurate classification but since objects can be at varying sizes, the relative location of the object in the image is required for higher classification accuracy.

### **2.5.2 Object Localization**

Localization is defined as predicting a bounding box for the object in the image (Sermanet et al., 2013). The terms localization, detection, and recognition are used interchangeably, therefore, recognizing or detecting an object imply that a bounding box and class label is assigned to the object of interest.

The first paper published on object detection using CNNs was from Sermanet et al., called OverFeat (Sermanet et al., 2013). The network is like Krizhevsky's which won the ILSVRC 2012, but this network was solely based on classification. To achieve localization, a regression layer was added after the last convolution layer, whose purpose is to output the coordinates of the bounding box encapsulating the object. The network, therefore, makes two predictions, the classification head gives the class label while the regression head gives the bounding box coordinates. After OverFeat, a plethora of networks based on Object localization was released, these include Region-Based CNNs (RCNN) (Girshick et al., 2014), Fast RCNN (Girshick, 2015), Faster RCNN (Ren et al.,

2016), Grid-based CNN (GCNN) (Najibi, Rastegari, & Davis, 2016), Single Shot Detectors (SSD) (Liu et al., 2016) and You Only Look Once (YOLO) (Redmon et al., 2016).

### **2.5.3 Segmentation**

At times, placing a bounding box around an object will not suffice as this produces a coarse representation of the object's location in the image. To achieve more accurate localization, researchers have moved to fine-grain inferencing; that is pixel-wise prediction. According to Bhadane, segmentation is the process of partitioning a digital image into multiple regions - groupings of pixels (Bhadane, 2013). More precisely, for every pixel (x) in the input image, the model assigns a probability of each pixel being associated with a given class (y). Generating pixel level predictions allows for a polygon mask to be overlaid on top of the object of interest which provides more precise localization of the object compared to overlaying with a bounding box.

The task of segmentation is divided into semantic and instance-based segmentation. Both tasks involve pixel-wise prediction, however, for instance-based segmentation, groups of pixels or instances of the same class can be differentiated and counted (Garcia et al., 2017), while information about how many instances of the same class being present in the image is not known for semantic segmentation.

Similarly, object detection networks which take an existing CNN built for classification and fine-tune it for the task of object detection, Long et al., proposed Fully Convolutional Neural Networks (FCNNs) which is a classification CNN fine-tuned for segmentation (Long, Shelhamer, & Darrell, 2015).

## 2.6 Fully Convolutional Neural Network for Segmentation

A fully convolutional network is a neural network that is composed of only convolutional layers – no fully connected layer. The last layer of CNNs used for classification is fully connected. However, this discards the spatial arrangement of the input, which is necessary for localization; it instead produces global information which only resolves *what* is in the image, but not local information, which determines *where* in the image (Long et al., 2015).

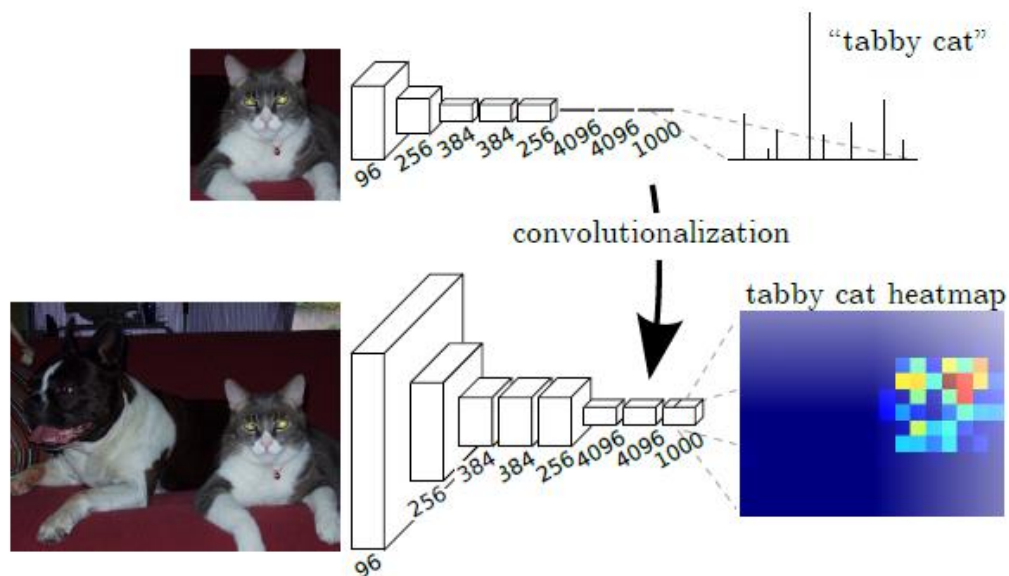


Figure 2-8: Transforming classification networks into segmentation networks. Retrieved from “Fully Convolutional Networks for Semantic Segmentation.” (Long, Shelhamer, & Darrell, 2015, p.3)

From Figure 2-8, using fully connected layers discards all the spatial information in the image, which is needed for localization. The fully connected layer will output an N-dimensional vector that contains a list of probability values of the input being associated with a given class. Replacing the last fully connected layer with a convolutional layer allows the network to output a heatmap, highlighting the activations



of the object and its location. There are a few new operations in CNN's built for segmentation.

### 2.6.1 Transposed Convolution (Deconvolution)

During training, a series of pooling operations downsamples the image such that the output dimensions are smaller in comparison to the original image, resulting in coarse prediction. However, for fine-grained pixel-wise prediction to be performed at the original resolution of the image we need to upscale/upsample our small dimension representation, this upscaling is performed by *Transposed Convolution*.

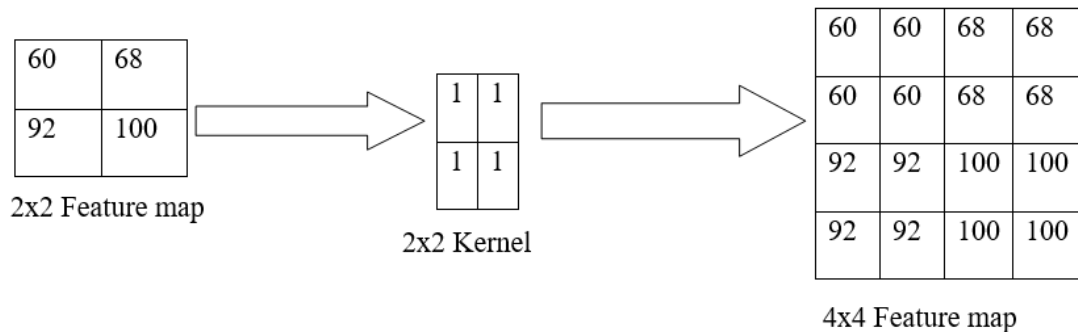


Figure 2-9: An illustration of upsampling feature maps using transposed convolution.

From Figure 2-9, transposed convolution increases the dimension of a 2x2 feature map to a 4x4. Each value from the feature map is multiplied by the weights in the kernel; the result is then used to create a larger feature map. While pooling halves the dimension, transposed convolution doubles the dimension. Transposed convolution does not require a special type of kernel; it uses the same kernel employed in the convolution operation, the role of the kernel is merely to enlarge the dimensions of the image.

## 2.6.2 Skip Connections (Layer Fusing)

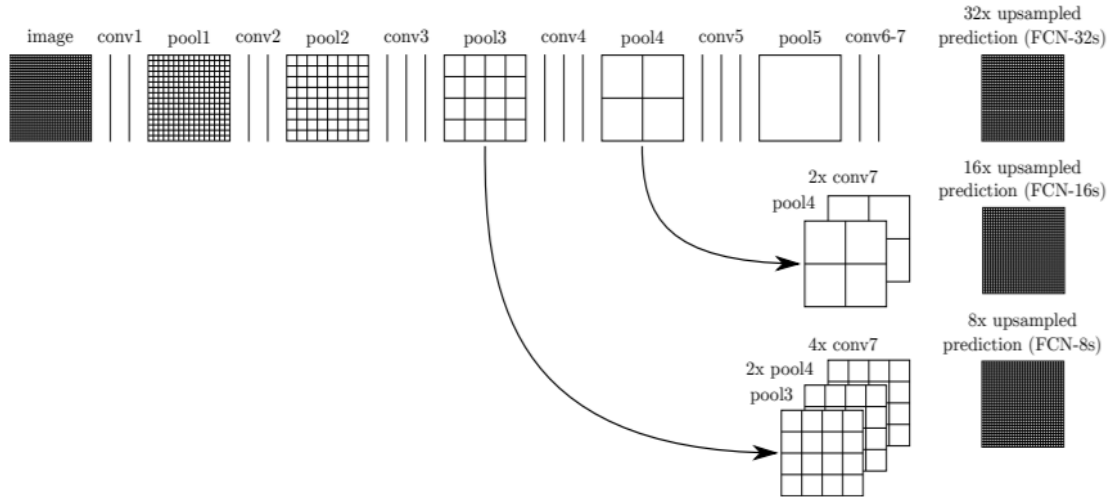


Figure 2-10: Skip Connections between fine and coarse feature maps. Retrieved from “Fully Convolutional Networks for Semantic Segmentation.” (Long, Shelhamer, & Darrell, 2016, p.6)

Another way to connect coarse feature maps to fine feature maps is with Skip & Fuse connections (Long et al., 2015). Skip connections connect deep coarse layers at the end of the network to earlier layers. This allows for finer details to be accessed from earlier layers before it has been lost due to downsampling. Fuse connections combine activations from earlier layers by summing or interpolation. Receptive fields are also smaller in earlier layers allowing for more sharper details to be captured, this results in a much smoother and defined prediction mask. Since the introduction of FCN, there have been various encoder-decoder variants for segmentation.

## **2.7 U-Net: Convolutional Networks for Biomedical Image Segmentation**

U-Net is a convolutional network architecture created by Ronneberger et al., which has won several competitions such as “Cell Tracking Challenge”, “Challenge for Computer-Automated Detection of Caries in Bitewing Radiography” and “Challenge for Segmentation of Neuronal Structures in Electron Microscopic Stacks.”

Previously, successful training of deep networks required thousands of images but annotating thousands or millions of images for a segmentation problem is a challenging task. The advancement in the field of segmentation using CNNs has been performed on prepackaged datasets such as PASCAL VOC or Berkeley, which provides thousands of annotated images created by a community of computer vision experts and enthusiasts. Researcher Ronneberger et al., showed that U-Net could be trained with a few annotated samples combined with data augmentation (Ronneberger, Fischer, & Brox, 2015). Data augmentation increases the size of the dataset and is used to simulate different states of the input thereby allowing the network to learn the desired invariance and robustness properties. Some examples of data augmentation with regards to images are zooming, rotating, cropping etc.

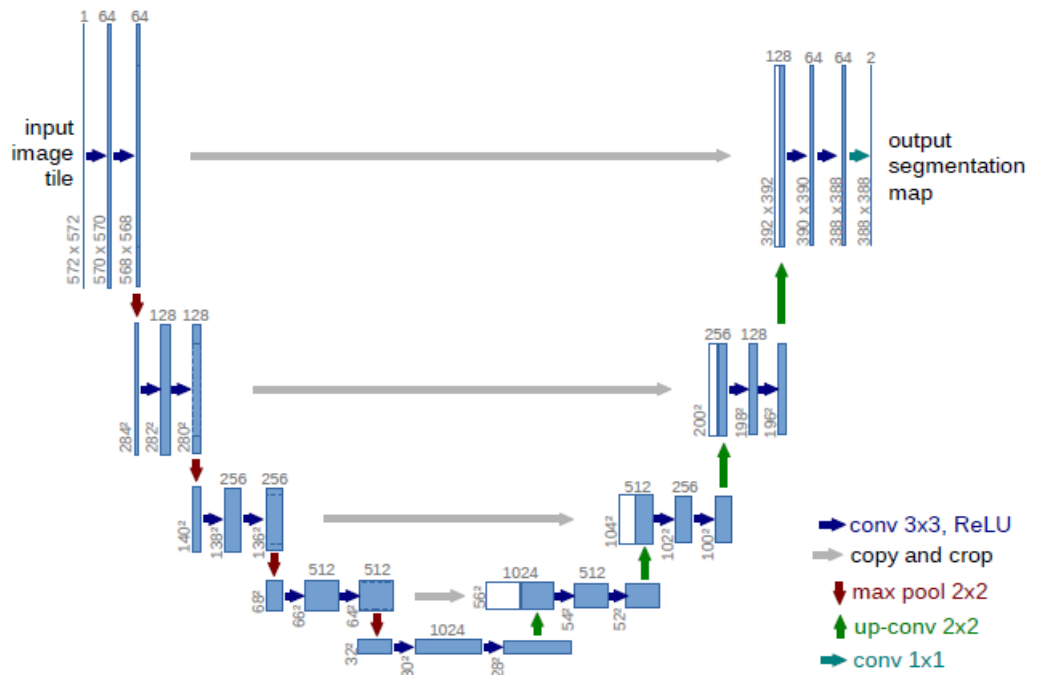


Figure 2-11: A diagram of U-Net. Retrieved from “U-Net: Convolutional Networks for Biomedical Image Segmentation.” (Ronneberger, Fischer, & Brox, 2015, p.2).

From Figure 2-11, the network resembles an auto-encoder. It has a contracting path (left) or encoder which produces a low-resolution image representation or feature map from successive pooling (downsampling) operations. The extracting path (right) or decoder then takes this coarse representation and upsamples it back to the original resolution of the image. Like FCN, skip connections are also used to fuse activations from earlier layers such that predictions are smoother.

After U-Net, there was deconvolution network which is another encoder-decoder variant that achieved the new state of the art with a mean IOU of 0.725 on the Pascal VOC dataset. It uses deconvolution and unpooling operations to upsample feature maps and removed skip connections as the researchers mention that improvement in

segmentation is minimal and it slows down training time. This research uses the U-Net architecture; creation of training data for segmentation is a lengthy task. However, Ronneberger et al., states that U-Net requires a few annotated images for successful training which made it a viable option for faster experimentation (Ronneberger, Fischer, & Brox, 2015). Finally, at the time of this thesis, U-Net is an established architecture hence, there is a significant amount of information and support in open-source APIs for the implementation of this network.

Researchers highlighted a significant problem with fully convolutional networks however, this problem affects all convolutional networks. The problem is the network's inability to handle objects at multiple scales. The term scale refers to the size of the object; varying sizes or scales of the object impact the performance of the network. According to Noh, the network can only handle single scale semantics within an image due to fixed size receptive field (Noh, Hong & Han, 2015).

Furthermore, Garcia et al., states that the kernel size will have an impact on the number of pixels that correspond to a single pixel in the convolved feature map, thereby causing filters to implicitly learn to detect features at specific scales, making generalization at different scales difficult (Garcia et al., 2017).

## **2.8 Techniques to overcome scale invariance with CNNs**

The task of representing objects should be invariant (never changing) to the transformation of the input data; The presence of light or orientation of the object should have minimal effect in distinguishing between objects. Three main sources of natural variations of the input are location, viewpoint, and size of the object or pattern (van Nord & Postma, 2017). Weight sharing, see section 2.4.4, allow CNNs to handle variations in

location and filters are created to recognize the object at varying viewpoints. However, size variations pose a challenge for CNNs (van Nord & Postma, 2017). Furthermore, it was determined in two separate publications, from (Noh et al., 2015) and (Garcia et al., 2017) that CNNs inability to handle size variations of the pattern is due to the kernel sizes.

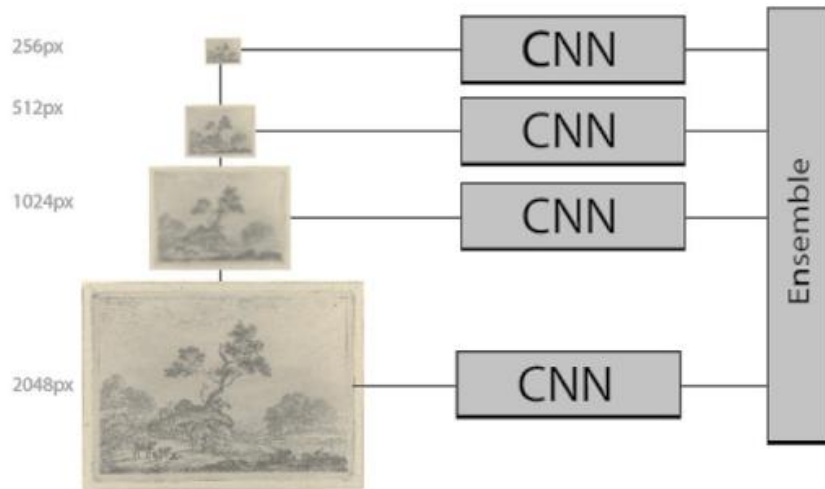


Figure 2-12: Scale-invariant CNN ensemble for classification. Retrieved from “Learning scale-variant and scale-invariant features for deep image classification.” (van Noord & Postma 2017, p.4)

Researchers have implemented various algorithms to tackle the problem of scale invariance. From Figure 2-12, van Nord (van Nord & Postma, 2017) used an ensemble of CNNs for classification where each CNN is trained on a lower resolution version of the original image generated from a Gaussian pyramid, starting from 2048 by 2048 to 256 by 256 pixels. Allowing each network to focus its learning and prediction at one scale eliminates the problem of scale invariance. However, the network is applied to the task of

classification and not segmentation which is two different yet similar problems. A convolutional network for classification cannot be used for the task of segmentation and vice versa. The task of localization or identifying the object's position in an image cannot be performed by a CNN for classification. The last layer of CNNs used for classification are fully connected however, fully connected layers discards the spatial arrangement of the pixels which is necessary for localization (Long et al., 2015), this was discussed in section 2.6.

Multi-scale CNNs were proposed by Raj et al., for semantic segmentation using the VGG architecture but relied on additional depth information to improve scale invariance (Raj et al., 2015). The multi-scale network used can accurately perform segmentation when an RGB image along with its recorded depth information is given as input. The researchers have appeared to develop their methodology independently, as no mention of each other's work was made.

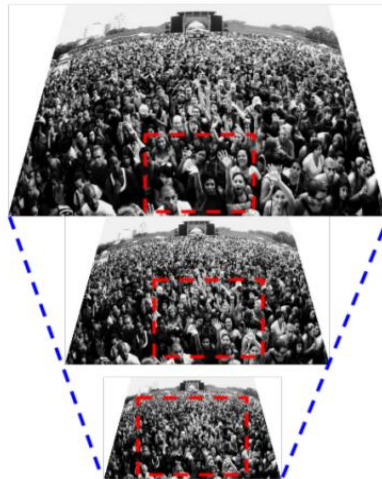


Figure 2-13: Multi-Scale Image Pyramid. Retrieved from CrowdNet: “A Deep Convolutional Network for Dense Crowd Counting.” (Boominathan, Kruthiventi & Babu, 2016, p.3).

Finally, researcher Boominathan et al., of CROWDNET states that the challenge of varying scales is a significant issue for segmenting faces in dense crowds (Boominathan et al., 2016); However, the scale issue can be effectively tackled by augmenting the training images. Patches of 225x225 were cropped from the original image and used for training which increased segmentation accuracy.

## 2.9 Data from Remote Sensing

Remote sensing is the use of instruments or sensors to acquire information about the earth's surface. The introduction of the camera resulted in early forms of remote sensing such as capturing aerial images using a camera attached to a balloon or a bird (Salle, 2010). Currently, the primary source of remotely sensed data is from aircraft and satellites, where it is used for analysis of different wavelengths of electromagnetic radiation.

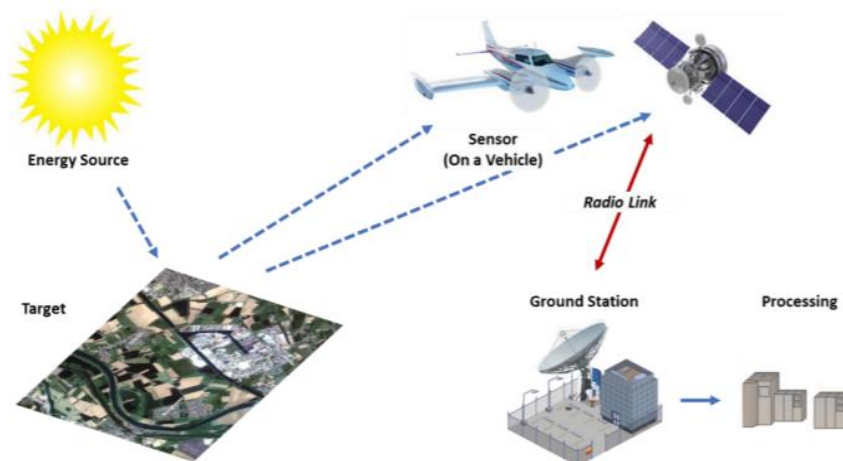


Figure 2-14: An illustration of reflection captured by aerial devices. Retrieved from “Remote Sensing System.” (Mohammad, 2015, p. 13)



From Figure 2-14, the energy emanating from the earth's surface is recorded by a sensor mounted on a platform such as an aircraft or satellite (Richards, 1999). Reflection or transmission of energy from the earth's surface in the form of waves is referred to as electromagnetic radiation. The sensor captures the percentage or intensity of energy reflectance with respect to each wavelength and used for the construction of images (Richards, 1999). The most crucial factor to consider when selecting or working with remotely sensed images is their resolution which can determine the outcome of your research. There are four imagery resolutions, Spatial, Spectral, Radiometric and Temporal resolution.

### 2.9.1 Spatial Resolution

Spatial resolution refers to the distance between the sensor and the target; the distance will determine the number of pixels that will be used to construct the object (National Resources Canada, 2015). A sensor that is further away from the target can cover a larger area but will have a low or coarse spatial resolution and cannot provide a high level of detail, while a sensor closer to the target can capture a higher degree of detail.

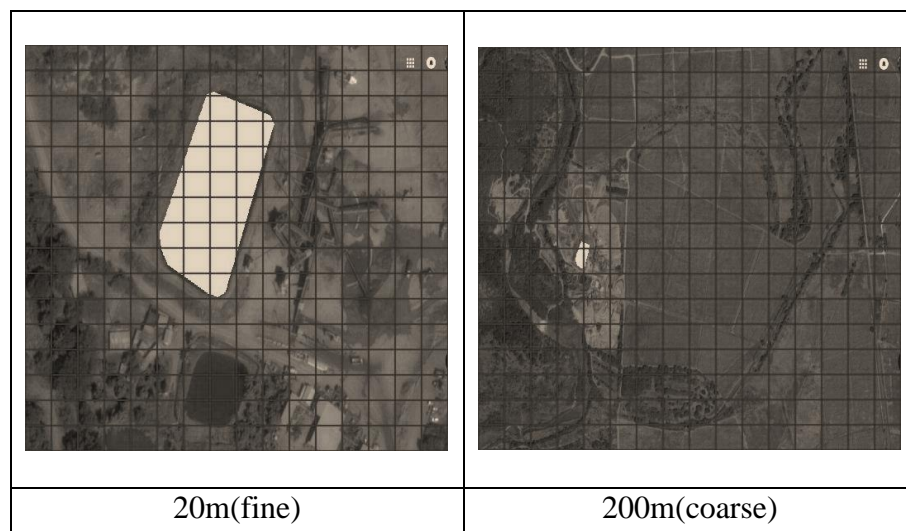


Figure 2-15: Difference between fine and coarse spatial resolution.

From Figure 2-15, the object at 20m has more information or pixels, but as we move to a coarser spatial resolution, we start to lose information about the object. This loss in information is because neighboring pixels are aggregated as we move further away, resulting in a smaller pixel count for the object. Segmentation of objects in an image is done by grouping homogenous pixels. However as pixel size becomes larger (fewer pixels) for the object of interest, this grouping is difficult or impossible to achieve.

## **2.10 Summary of Chapter 2**

Machine Learning methods such as clustering and SVM can be used to segment images. However, the advancement in the field of deep learning has resulted in a significant increase in image recognition accuracy by CNNs.

Unfortunately, CNNs struggle with variations in the size of the object due to fixed kernel sizes. Spatial resolution determines the number of pixels used to construct the region; therefore, changes in spatial resolution will result in misclassification by the network. Researchers have tried different methods such as using an ensemble of neural networks or generating an image-scale pyramid to tackle the issues of varying scales. The scale problem was investigated in future chapters.

# Chapter 3

## Study and experimental design






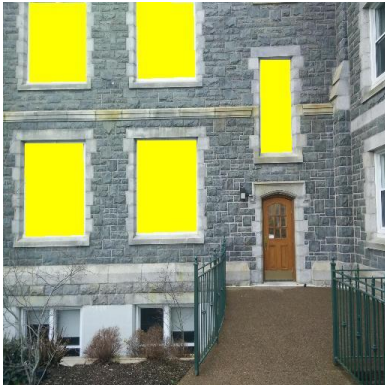
CNNs are state of the art in image recognition; however, multi-scale recognition is a problem rooted into the engine that drives these networks, and that is filters or kernels. Filters are responsible for extracting and learning features during training, and then at prediction, features are extracted from the image which is used to recognize the object of interest. Features are learnt by grouping neighboring pixels, however, when at a different scale, the number of pixels used to construct the object changes, therefore, a different arrangement or groups of pixels will be used to construct the object.

Many areas such as self-driving cars, would benefit immensely from scale-invariant CNNs. Cars would be able to identify small objects from further distances and can apply brake accordingly based on the speed of the vehicle, providing a safer experience. However, CNNs need to be scale invariant to achieve high levels of accuracy. The goal of this study is to identify and develop techniques which can be used to improve image segmentation or produce more accurate segmentations using CNNs, specifically U-Net.

### **3.1 Overview of Dataset**

Three datasets were used in these experiments with different Regions Of Interest (ROI) identified. The pond dataset consists of aerial images captured from Google Earth with an oval-shaped pond being the ROI. The door dataset was captured using a camera at McNally main, a building on Saint Mary's University. Three different regions of interest were used, these are door, window, and wall. The grass dataset was captured using a camera; the images were captured from a front lawn at a resident's location in Jamaica. The region of interest is a disease or fungus (Brown patches) which was affecting the Zoysia grass.

The datasets captured differ from each other in terms of the color, shape, and texture of each region. This allows for a general application of the proposed multi-scale segmentation technique.

ROI	Image( $X_{ij}$ - RGB)	Mask( $Y_{ij}$ - Binary)
Pond		
Door		
Window		

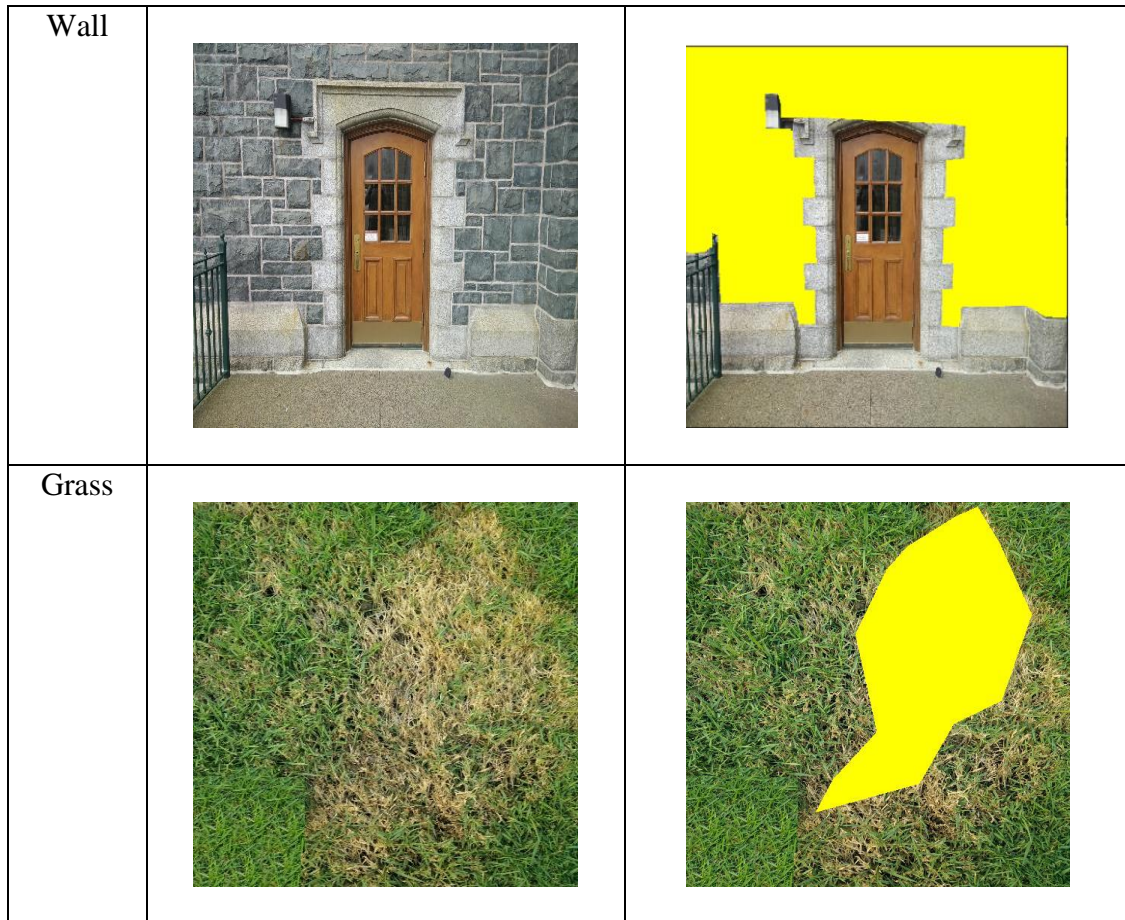


Figure 3-1: Images from the highest spatial resolution for each ROI.

In Figure 3-1, the shaded areas in the images represent the region of interest for each dataset, the corresponding label for each region can be found in column one. There are several scales associated with each region where the goal is to design a methodology that maximizes the prediction accuracy across several scales, specifically the coarsest scale. Images at coarser spatial resolutions can be found in Appendix 8.1.

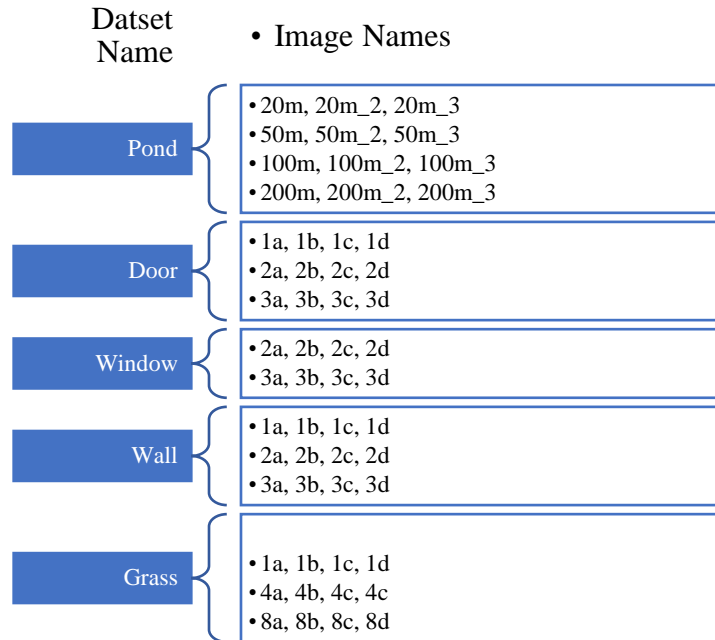


Figure 3-2: A diagram showcasing the structure of the dataset.

From Figure 3-2, each region has several spatial resolutions and multiple images (3-4) per spatial resolution. For example, Door has the range of spatial resolutions one to three, with one being the highest and three being the lowest. Each spatial resolution has four training points labeled *a* to *d*. The distances between scales will vary across dataset, but what is important is that the region is at different scales.

### 3.1.1 Data Preparation

The data used in this research consisted of input data ( $x_{ij}$ ) and corresponding label ( $y_{ij}$ ). Images ( $x_{ij}$ ) were captured using a camera or screen capture device then an annotation software was used to manually draw polygons using a mouse guided by the researcher. Annotating the image creates a mask over the region of interest. Usually, deep nets would require thousands of images, however, creators of U-Net Ronneberger et al., showed that U-Net could be trained with a few annotated samples combined with data

augmentation (Ronneberger et al., 2015). Data augmentation increases the size of the dataset, see Appendix 8.4 for the results of using different training sizes. The results show that the network can produce accurate predictions with a small dataset combined with data augmentation. The input image ( $x_{ij}$ ) was stored as a jpg file and contained intensity values from 0-255 captured from the Red, Green, and Blue (RGB) channels. The associated label ( $y_{ij}$ ) contains a mask that encapsulates the region of interest in the input image, this encapsulation of  $y_{ij}$  over  $x_{ij}$  is referred to as segmentation. The mask is stored in a .tiff file where Boolean values (True or False) indicate which pixels from the mask belong to the region. A mask containing Boolean values is called a binary mask. There is no justification for the use of a tiff file to store the mask, as a jpg image could be used; any image-based file format can be used, these include but are not limited to PNG, BMP, and GIF.

Before training can commence, images must be preprocessed, this includes resizing and scaling pixel intensity values. Images are resized to a resolution of 512 by 512, as training and inferencing on varying image resolutions produce poor results. The intensity values are then normalized from 0-255 to 0-1. According to Sola, normalizing the input data prior to network training results in faster convergence and lower estimation errors (Sola & Sevilla, 1997). Normalization ensures that each variable is assigned equal weight; the variable with the largest scale will dominate the outcome. Having intensity values too large or small will make the network overcompensate weight adjustment in one layer while under compensating in others leading to frequent and lengthy oscillations during training.



There are several normalization methods, where each method depends on the input data. Min-Max or Feature Scaling is one of the most common normalization methods used.

$$x_{ij} = \frac{x_{ij} - \min(x)}{\max(x) - \min(x)}$$

Figure 3-3: Formula for normalizing pixel values.

In Figure 3-3, each pixel contained in the image is represented as  $x_{ij}$ . The parameter -  $\max(x)$ , represents the largest pixel value in the current matrix or channel, while  $\min(x)$  represents the minimum pixel value. The formula is applied for each pixel  $x_{ij}$  across all channels in the image, these are red, green and blue. The normalized pixel value will range from 0 to 1.

### 3.1.2 Training Procedure

Training was performed per scale, where each scale consist of four images. Each image is preprocessed, and appropriate data augmentation applied which doubles or triples the size of the training set. After performing augmentation such as rotations the resulting training size consist of eight images. If additional augmentation is done, then the dataset would increase by four images. The validation set was a 20% split of the training set, which results in the training size consisting of six images and validation size would contain two images. At the end of training, the model is evaluated using the test data set, which is the region of interest at the next scale. The test set consist of four images. See

Appendix 8.4 for the training parameters used in these experiments. The number of images across all regions is not the same, e.g. Pond only consists of three images, therefore, the size of the training and validation set will differ.

Deep networks usually require thousands of images to learn an accurate representation of a given object. However, several researchers such as (Ronneberger, Fischer, & Brox, 2015) and (Boominathan, Kruthiventi & Babu, 2016) mentioned that a few annotated samples coupled with data augmentation enable segmentation network to learn to desired invariance and robustness properties. Prior experiments were conducted to determine if more data is required to improve prediction accuracy. See Appendix 8.4 for the results of these experiments.

The results show that the network can produce accurate predictions with a small dataset combined with data augmentation. Types of data augmentation methods used in this thesis include Zoom Augmentation, Rotation, Contrast Enhancement, and Sobel Filtering. Each method of augmentation produces four additional images, which allows for exponential growth in the size of the training set. However, each augmentation method may be used independently based on the problem.

### 3.1.3 Result Evaluation

Predictions from segmentation networks are three-dimensional tensors with dimensions  $W \times H \times C$ , where  $W$  and  $H$  represent the width and height of the input image and  $C$  is the number of classes. If the input image is  $512 \times 512$  and training is performed on one class, then the output tensor is  $512 \times 512 \times 1$ . Each class matrix from the output tensor consists of probability values. A level of confidence of 0.99 was used to segment the image generating a binary mask, where true pixels belong to the region of interest and false do not. The IOU score does not differ drastically when different confidence levels are used, as the probability scores for true pixels are close to 0.99, see Appendix 8.8.

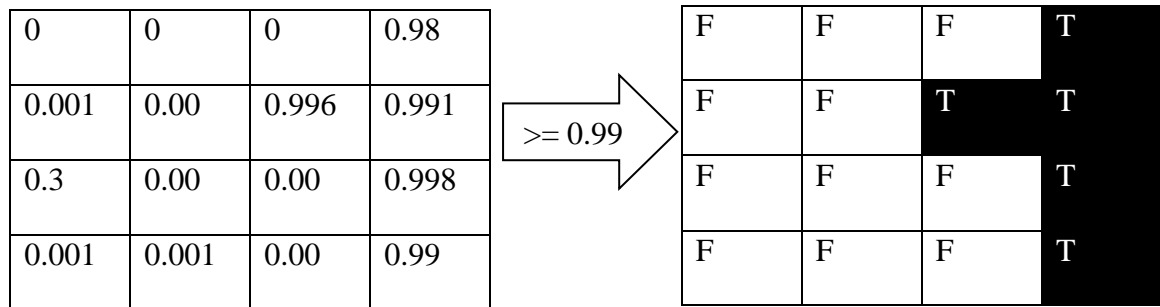


Figure 3-4: Thresholding of probability values to produce a binary mask.

From Figure 3-4, probability values greater than or equal to the confidence level of 99% will be segmented as part of the region. Evaluation of segmentation mask is a measure of how well the predicted mask overlays the region of interest; it is a measure of similarity between pixels predicted as belonging to the region and the actual pixels that belong to the region. The two most common evaluation metrics for segmentation is pixel accuracy and intersection over union.

$$PA = \frac{TP + TN}{TP + TN + FP + FN}$$

Figure 3-5: Formula for calculating pixel accuracy for segmentation.

Pixel accuracy is the ratio between correctly classified pixels and the total number of pixels in the image (Garcia et al., 2017). However, pixel accuracy is not the standard metric used to evaluate segmentation since it does not consider false positives – misclassified pixels. If the region of interest is fully segmented but other regions of the image are mis-segmented, this will result in high accuracy.

### 3.1.3.1 IOU (Intersection-Over-Union)

The intersection over union is the most used evaluation metric for segmentation. It computes a similarity score between zero and one based on the overlap between the predicted pixels and the actual pixels. Unlike pixel accuracy, the score is penalized for false positive; therefore a large portion of mis-segmented pixels will result in a low IOU score.

$$IoU = \frac{TP}{FP + TP + FN}$$

Figure 3-6: Formula for calculating Intersection-Over-Union (IOU). Retrieved from “Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation.” (Rahman & Wang, 2016, p. 5)

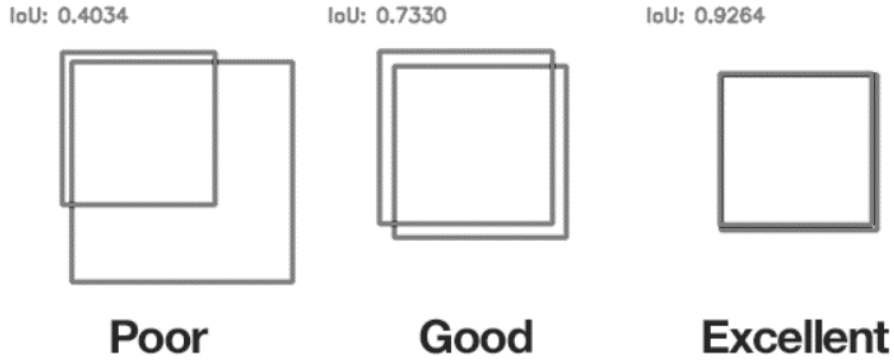


Figure 3-7: Comparison between examples of overlaps and corresponding IOU score.

In Figure 3-7, the greater the overlap between the predicted region and the actual region, the higher the IOU score.

## 3.2 Experiment applications and software

### 3.2.1 Loss Function

Cross-entropy has been used by (Ronneberger et al., 2015) and (Long et al., 2015) as their loss function for training U-Net and FCN respectively. However, Rahman stated that this loss function is suited for classification problems and not segmentation, instead the objective function to be minimized should be the Intersection Over Union (IOU) (Rahman & Wang, 2016). Researchers Rahman et al., also argue that optimizing the IOU score directly is superior to using a classification loss function such as cross-entropy.

$$L_{IoU} = 1 - IoU$$

Figure 3-8: Loss or Cost function to be optimized.

### **3.2.2 Optimizer**

Stochastic Gradient Descent (SGD) was used by (Ronneberger et al., 2015) and (Long et al., 2015) for the training of UNet and FCN respectively. However, selecting a learning rate can be a difficult task, if a low learning rate is selected convergence at the minima of the objective can be slow, but if a high learning rate is selected, then there is a possibility of overshooting the minima. Adaptive Learning algorithms such as Adagrad (Duchi, Hazan, & Singer, 2011), Adadelta (Zeiler, 2012) and Adam (Kingma, & Ba, 2014) are variations of stochastic gradient descent that adapts the learning rate hyperparameter during training. Adadelta was used by Simo-Serra et al., for their experiments which converted sketch drawings into computerized vectors (images) using FCNs (Simo-Serra et al., 2016). Adadelta was also used by Wang for training a CNN to recognize pedestrians and vehicles (Wang & Xu, 2015). The Adam optimizer was used for this research as other optimizers did not perform as well. Adam proved to be the most stable and consistent across all datasets. See Appendix 8.6 for the results from several optimizers.

### **3.2.3 Software and Tools**

Python was the programming language of choice; several frameworks or packages were installed for additional functionality. The main frameworks used were TensorFlow (Abadi et al., 2016) and Keras (Chollet, 2015), these provided the tools for building a CNN for Segmentation. Tensorflow is an open-source framework for Machine learning applications developed by Google and was released in 2015. Keras is a high-level API built on top of Tensorflow, it provides all the required layers and optimizers required for this research.

All experiments were performed on an NVIDIA GTX 1080 equipped with 8 GB of GDDR5 RAM, this allows for extremely fast training and inferencing time. The CUDA library provided by NVIDIA was installed to accelerate calculations. See Appendix 8.7 for training times.

### **3.3 Problem Scope**

According to van Nord, variations in the scale of an object or region have posed a problem for convolutional neural networks (van Nord & Postma, 2017). The source of the scale problem was highlighted by (Garcia et al., 2017) and (Noh et al., 2015), which states that the kernel size is the reason why CNNs struggle with variations in scale. CNNs learn by grouping neighboring pixels; pixels close in proximity are highly correlated.

However, as the scale of the object changes so does the number of pixels used in its construction. Edges are low-level features and are defined by a set of points or pixels; therefore differences in pixel count, i.e., changes in scale will impact the construction of edges by the network. This explains why objects are mis-segmented at unlearned scales because the edges appear to be different. The edges also define the structure or shape of the region, therefore if the edges are undetectable then so will the region.

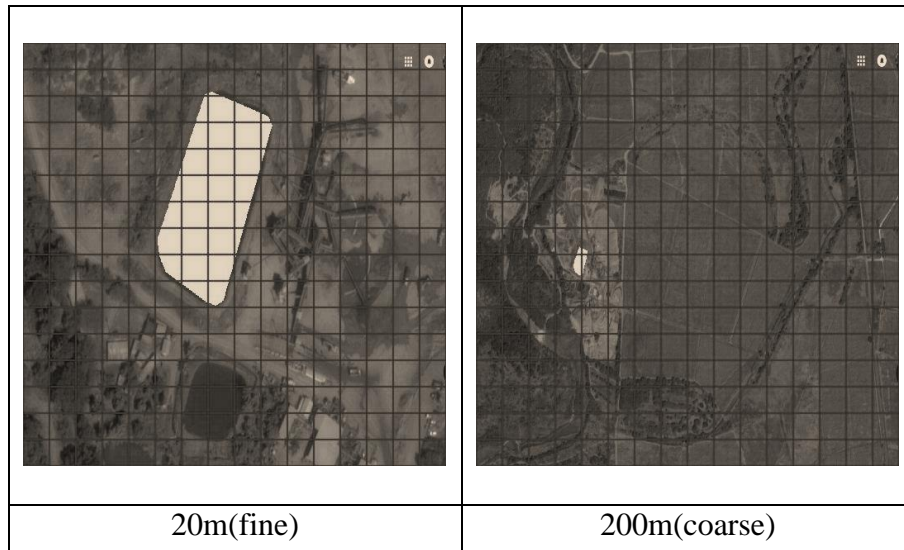


Figure 3-9: Difference between fine and coarse spatial resolution.

From Figure 3-9, the region at 20m was constructed using more pixels therefore if a 2x2 kernel convolved over this image, groups of four pixels will correspond to a single pixel in the feature map. However, if the kernel was applied to the object at 200m a group of one or two pixels will be learnt. Features learnt from the object at 20m cannot be used to identify the object at 200m.

The goal of this research is to develop a methodology that allows proven neural networks applied to segmentation to be more robust when handling objects at varying scales. There should be a limit on the number of scales since there can exist an infinite amount of scales. A reasonable limit for the number of scales is when the human eye can no longer detect the object. These networks are expected to complete a task usually performed by humans, therefore if CNNs are to achieve human-like performance, any object visible to a human must be visible to the network. The lowest scale for each region was determined based on visual inspection.

The methodology was an experimental iterative approach. Several methods were tested based on insight from the literature and reasoning. After each iteration, based on



results from the experiments, the technique used to solve the problem of scale invariance was modified accordingly. Results from experiments are evaluated quantitatively using metrics such as IOU.

### **3.3.1 Algorithm Use Case**

As mentioned in earlier chapters, there are several applications for the task of segmentation. A possible area that can benefit from improved image segmentation is agriculture. Early pest detection implies to *constantly* monitor crops; therefore, images should be captured frequently. Manually capturing images of crops on a farm occupying a few acres is impossible which means an automated method must be applied for early pest detection. Automated pest detection can be performed on-ground using remote control vehicles. However, farmers usually plant crops in close proximity in an attempt to maximize their yield thereby maximizing their profit. These densely packed farms make it impossible or difficult for a ground unit to navigate through rows of crops and acquire data.

A more efficient approach for acquiring data for pest detection is aerial imagery. During routine inspections, farmers can capture high spatial resolution images that highlight the diseased region of the crop. This on-ground data can be used for training a neural network, then aerial images from a drone or satellite can be used for monitoring and detection of diseases. However, due to the scale issue with CNNs, detection of diseases cannot be performed at further altitudes (lower spatial resolution). Farmers would have to capture and highlight the diseased region from aerial images at different spatial resolutions.

The proposed method should improve the prediction accuracy for Convolutional Neural Networks at coarser scales, given that training data was provided at one scale (the highest).

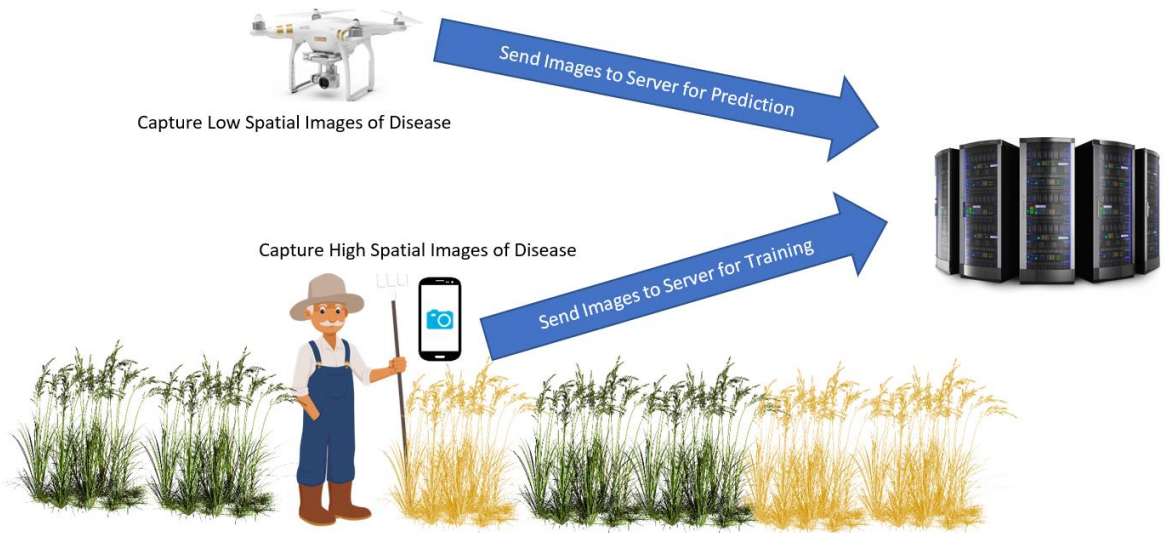


Figure 3-10: An illustration of a farmer capturing images on-ground and identification performed from an aerial device.

In Figure 3-10, a farmer is shown capturing high spatial resolution images on the ground using a cell phone. A drone is then used for detection of disease at higher altitudes.

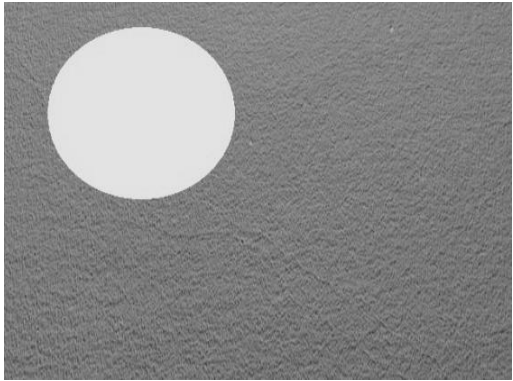



Scale	Field 1	Field 2
100m		
Key	 Diseased Crop	 Healthy Crop

Figure 3-11: Difference between late and early detection.

To highlight the importance of producing scale invariant CNNs for aerial disease detection, see Figure 3-11. Field 1 shows that the disease has spread to a large portion of the farm, this can be referred to as late detection. Field 2 indicates the detection of a small disease region - early detection. Scale-invariant CNNs is the difference between detecting a disease infestation early or late. The earlier the infestation is detected; the less money or crops will be lost.

### 3.4 Approach to problem

Simply, CNNs can only identify what they have been trained on; Classifying a region at 100 meters with a CNN trained at 10 meters, is like classifying a dog for a CNN trained on bananas. Data augmentation is used to generate new data by transforming data that already exist, it speeds up convergence, reduce overfitting and increase generalization capabilities (Garcia et al., 2017). According to Ronneberger et al., data

augmentation can be used to simulate different states of the input thereby allowing the network to learn the desired invariance and robustness properties (Ronneberger et al., 2015). The augmentation method of choice depends on the problem being solved. Therefore it is essential to understand what happens to the region of interest as it moves across different scales.

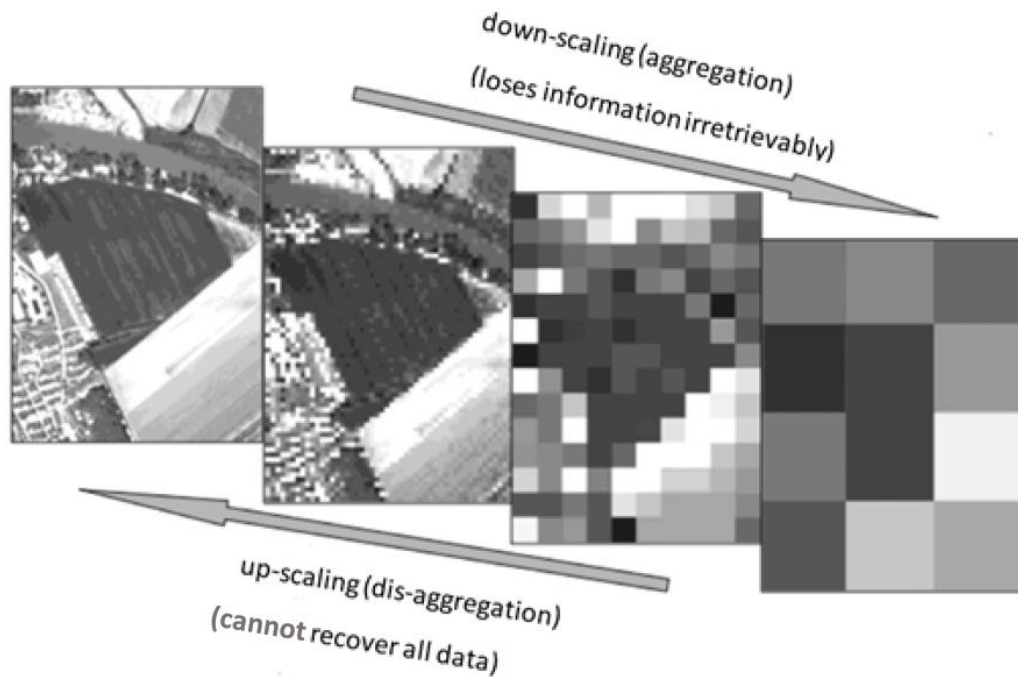


Figure 3-12: Process of up and downscaling an image. Retrieved from “Scaling of thermal images at different spatial resolution.” (Jones & Sirault, 2014, p. 2)

From Figure 3-12, transitioning from a higher to a lower spatial resolution results in a reduction in the number of pixels. The reason for this is that neighboring pixels are grouped and combined into a scalar pixel when moving from a high spatial resolution to a low spatial resolution, this combining or aggregation of pixels is known as downscaling or downsampling, also called subsampling. Low resolution is associated with a small

number of pixels while high resolution is associated with a high number of pixels. The process of aggregating pixels (downscaling) is irreversible, as information is lost during the process, so one cannot create a high-resolution image from a lower resolution image without additional data (Jones et al., 2014). With that said, to simulate the region of interest at a coarser scale, the augmentation method must be able to reduce the number of pixels for the region of interest.

### **3.4.1 Zoom Augmentation – Image Scaling**

To simulate the region of interest at a coarser scale, the pixel values must be rescaled. Scaling an image involves the use of interpolation to construct new data points based on the range of intensity values for the image. These data points will be used to upsample or downsample the image. For coarse representation of the region, we need to perform downsampling since we need to reduce the number of pixels for the region, this downsampling is sometimes called zoom augmentation since we are moving further away from the region of interest. Therefore, image scaling is the data augmentation method of choice used to step across the different spatial resolutions.

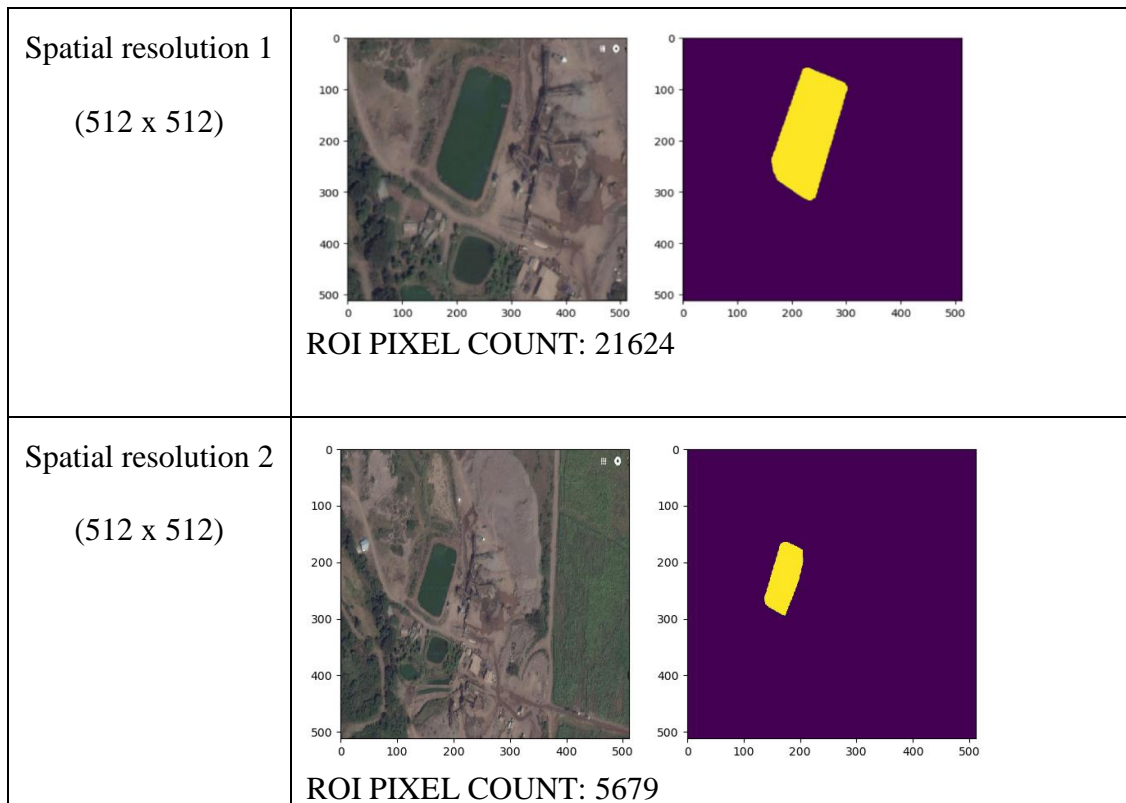


Figure 3-13: Pixel count for the region of interest at scale one and two.

In Figure 3-13, the only training data we have is at spatial resolution one and its region of interest contains 21624 pixels, and our target spatial resolution that we want to *step* to contains 5679 pixels for its region of interest. This can be done by downsampling/downscaling the image at spatial resolution one using a scaling factor. A simple method to determine what scaling factor to use is the ratio between spatial resolutions. From Figure 3-13,  $20\text{m}/50\text{m} = 0.4 = \sim 0.5$ . We can approximate and use 0.5 as the scaling factor. The output resolution after downsampling a  $512 \times 512$  by 0.5 is  $512 * 0.5 = 256 \times 256$ . Since downscaling reduces the number of pixels for the region, the pixel dimensions or resolution of the image is also reduced.

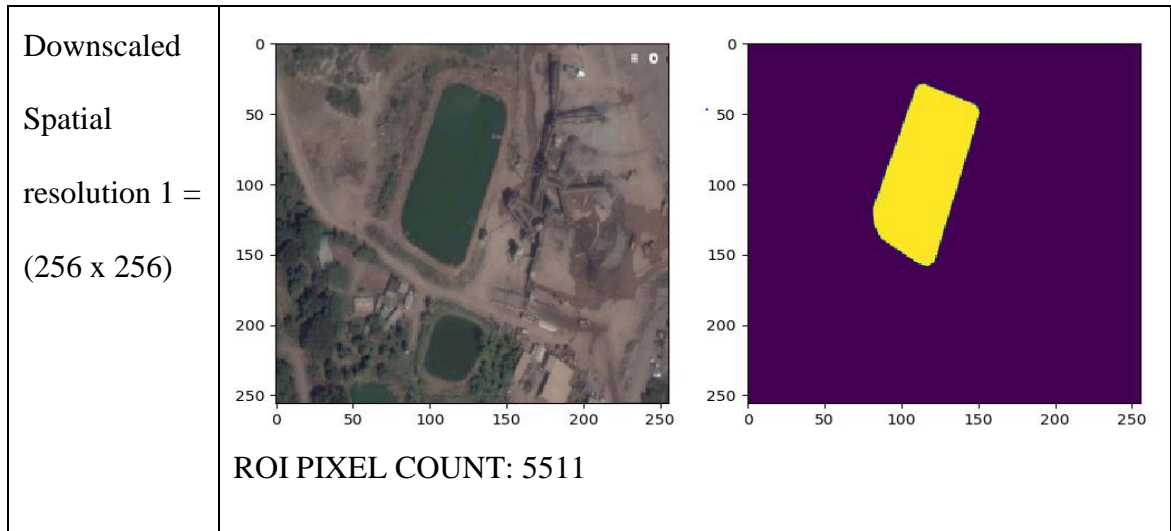


Figure 3-14: Pixel count for the region of interest after downsampling the image.

From Figure 3-14, The pixel count for the region of interest is now 5511, which is close to the next spatial resolution with 5679 pixels as seen in Figure 3-13. Before training can commence, the downscaled image should be padded with pixels such that the original image resolution is maintained. This is important since training and inferencing at different resolutions can produce results that differ drastically.

Chapter 4 will consist of experiments demonstrating the scale problem of CNNs due to the kernel sizes, then the benefits and limitations of using image scaling to improve multi-scale segmentation. Finally, what alternative method can be used to circumvent the limitations of image scaling.

### **3.5 Summary of Chapter 3**

Chapter 3 presents the dataset and training procedure used for the experiments. An object transitioning from a high to low spatial resolution will have a reduced pixel count due to pixel aggregation. Therefore, to improve prediction accuracy at lower spatial resolutions, the data augmentation of choice must perform pixel aggregation on the object of interest.

The training procedure was explained, each image is normalized using min-max scaling, then augmentation techniques are applied which results in an exponential growth in the size of the dataset. Augmentation techniques include rotation, zooming, and cropping. The training parameters including the number of data points and image resolution are mentioned, for more details see Appendix 8.4.



# Chapter 4

## Scale-Invariance

Several experiments were conducted with the aim of developing a technique that enables U-Net to be invariant to changes in the objects scale. See section 2.7 for the reason for using U-Net; U-Net is made relatively available with resources to support the implementation of the architecture and it also requires a small dataset which speeds up experimentation time. The experiments performed will demonstrate the scale issue and show techniques to alleviate it. In this chapter some terms will be used interchangeably, these are spatial resolution and scale, highest and finest, coarsest and lowest.

### 4.1 Training on One Spatial Resolution

Invariance means that the object should still be recognizable even if its appearance varies – deformation. As humans, we can identify an object after presented with a few examples, despite the angle, orientation or lighting of the object. Convolutional Neural Networks are robust to certain deformations, for (e.g.) they are invariant to changes in viewpoint (angle) and rotation of the object. However, CNNs struggle to be scale-invariant; generalization is poor at scales not trained on by the network. To demonstrate this scale problem, a network was trained on one scale and used to predict across several unseen scales. One scale for a region can contain 4-12 images, see section 3.1.2 for details about the training procedure and Appendix 8.4 for the training parameters used in these experiments.

### 4.1.1 Training on the Highest Spatial Resolution

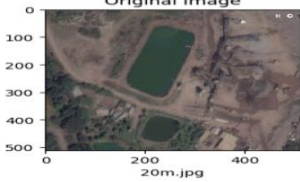
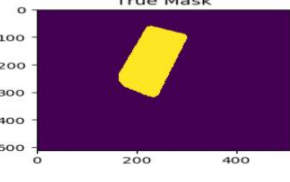
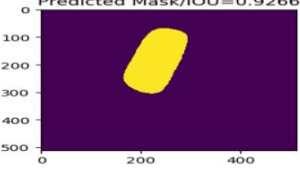
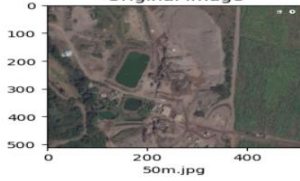
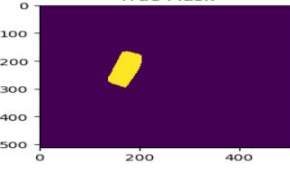
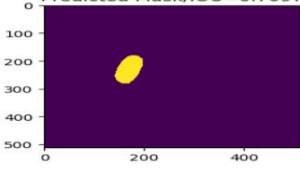

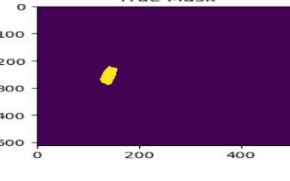
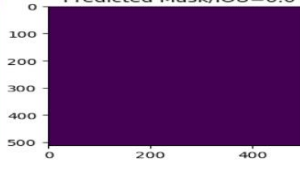

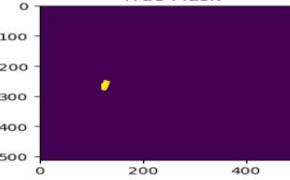
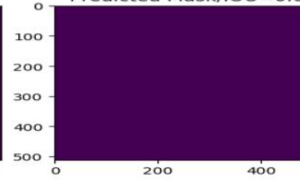
Scale	Results		
20m	 <p>Original Image 20m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.9266</p>
50m	 <p>Original Image 50m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.7097</p>
100m	 <p>Original Image 100m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0</p>
200m	 <p>Original Image 200m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0</p>

Table 4-1: Network was trained on the highest scale for the pond dataset. Images in the first column show the RGB image (x), the second column shows true mask (y) for the region of interest, and the third column shows the prediction from the classifier.

From Table 4-1, an IOU score of 0.92 and 0.7 was predicted for the 20m and 50m scales respectively, while no predictions were made for the 100m and 200m scales.


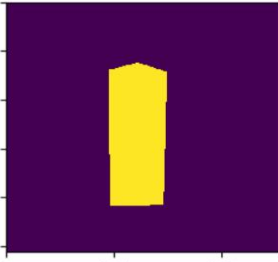
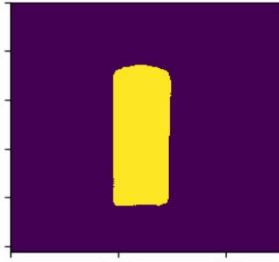

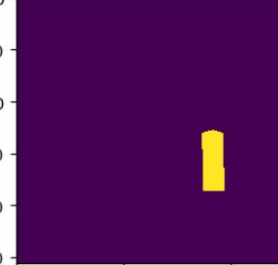
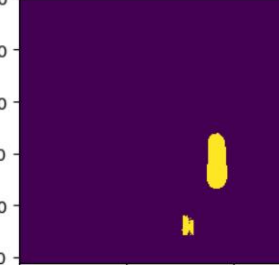

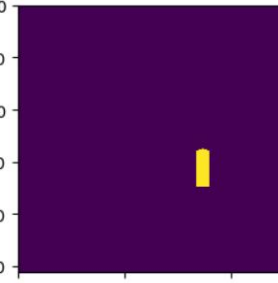
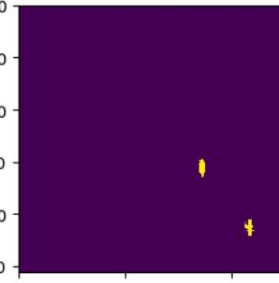
Scale	Results		
1	<p data-bbox="516 411 667 436">Original Image</p>  <p data-bbox="565 716 618 741">1a.jpg</p>	<p data-bbox="862 411 976 436">True Mask</p> 	<p data-bbox="1114 411 1390 436">Predicted Mask/IOU=0.9748</p> 
2	<p data-bbox="524 806 675 831">Original Image</p>  <p data-bbox="573 1115 626 1140">2a.jpg</p>	<p data-bbox="870 806 984 831">True Mask</p> 	<p data-bbox="1114 806 1390 831">Predicted Mask/IOU=0.6636</p> 
3	<p data-bbox="532 1209 683 1234">Original Image</p>  <p data-bbox="581 1535 634 1560">3a.jpg</p>	<p data-bbox="873 1209 987 1234">True Mask</p> 	<p data-bbox="1114 1209 1390 1234">Predicted Mask/IOU=0.1538</p> 

Table 4-2: Network was trained on the highest scale for the door dataset.

The predictions shown in Table 4-2 differs drastically across spatial resolutions. The scale that was trained on (Scale 1) gives an IOU score of 0.97 while the remaining two scales give 0.6 and 0.15 respectively.


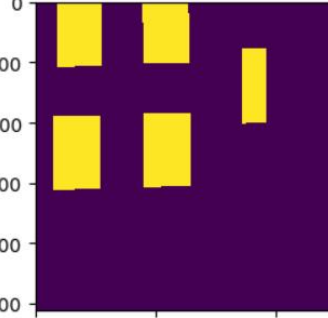
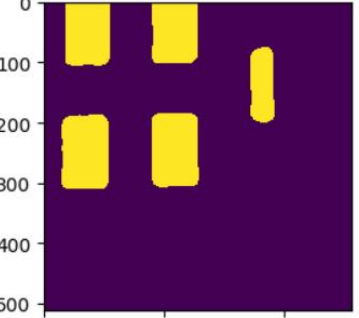
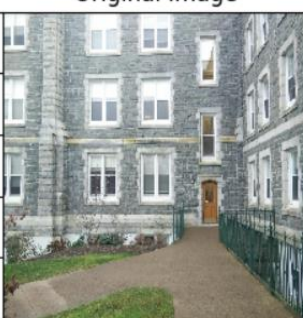
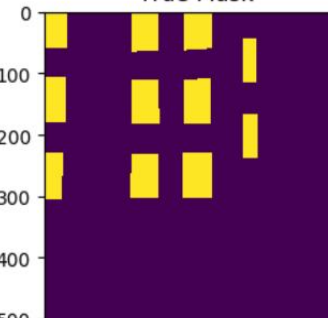
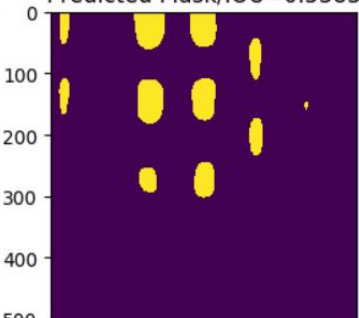
Scale	Results		
2			
3			

Table 4-3: Network was trained on the highest scale for the window dataset.

The results remain the same for the window as seen in Table 4-3. The edge or outer regions of the windows were not segmented at scale 2, resulting in a lower IOU score of 0.53.



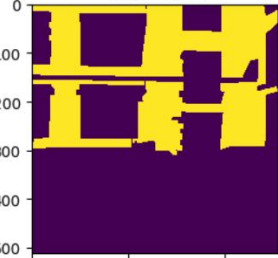
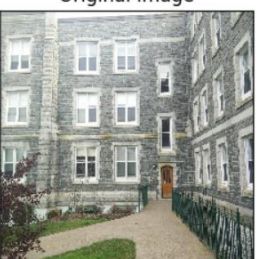
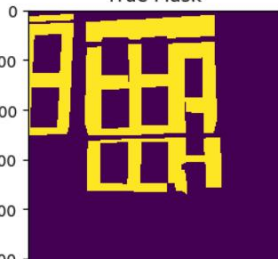
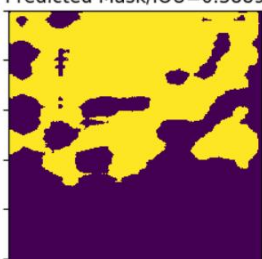
Scale	Results		
1	<p data-bbox="553 258 691 279">Original Image</p>  <p data-bbox="597 562 647 583">1a.jpg</p>	<p data-bbox="870 258 967 279">True Mask</p> 	<p data-bbox="1089 258 1349 279">Predicted Mask/IOU=0.8615</p> 
2	<p data-bbox="553 646 691 667">Original Image</p>  <p data-bbox="597 951 647 972">2a.jpg</p>	<p data-bbox="870 646 967 667">True Mask</p> 	<p data-bbox="1089 646 1349 667">Predicted Mask/IOU=0.5929</p> 
3	<p data-bbox="553 1035 691 1056">Original Image</p>  <p data-bbox="597 1339 647 1360">3a.jpg</p>	<p data-bbox="870 1035 967 1056">True Mask</p> 	<p data-bbox="1089 1035 1349 1056">Predicted Mask/IOU=0.3889</p> 

Table 4-4: Network was trained on the highest scale for the wall dataset.

The same effect is also observed for the wall dataset as shown in Table 4-4. The IOU score at the highest scale is 0.86 while the lowest scale is 0.38. These experiments show that the scale issue affects all regions of interest, despite having different features in terms of shape, color, and texture.

### 4.1.2 Training on the Lowest Spatial Resolution

To demonstrate the inability of CNNs to generalize for new or unseen scales even further, another set of experiments were conducted using only the coarsest spatial resolution image for training.


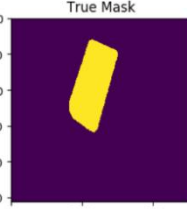
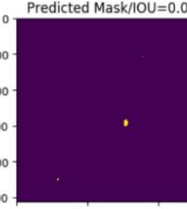
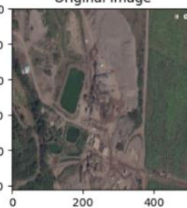
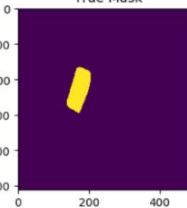
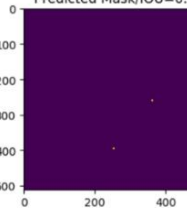
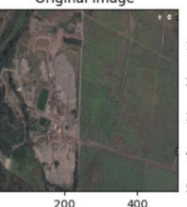
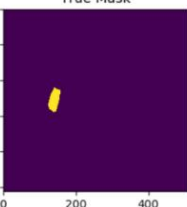
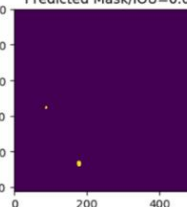

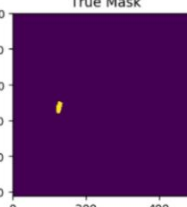
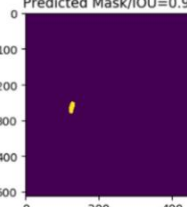
Scale	Results		
20m	 <p>Original Image 20m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0</p>
50m	 <p>Original Image 50m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0</p>
100m	 <p>Original Image 100m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0</p>
200m	 <p>Original Image 200m.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.9537</p>

Table 4-5: Network was trained on the lowest scale for the pond dataset.

In Table 4-5, the opposite effect is seen, where the network produces the highest IOU score at the scale it was trained on (200m) but fails to segment any pixels on scales not seen (higher scales). To contrast, Table 4-1 shows 0.92 at the highest spatial resolution while Table 4-5 shows 0.95 at the coarsest spatial resolution.


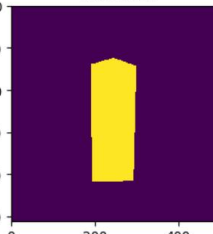
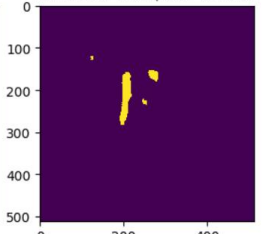

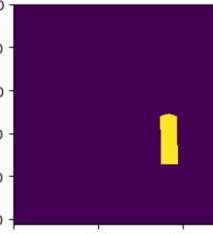
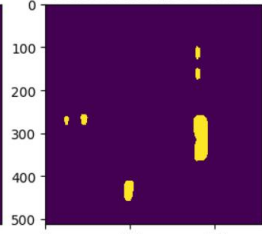

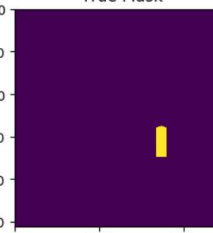
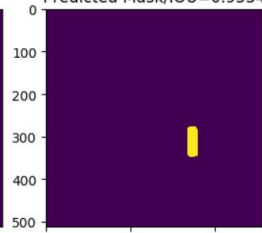
Scale	Results		
1	<p data-bbox="607 701 721 722">Original Image</p>  <p data-bbox="639 961 688 982">1a.jpg</p>	<p data-bbox="867 701 964 722">True Mask</p> 	<p data-bbox="1062 701 1289 722">Predicted Mask/IOU=0.0899</p> 
2	<p data-bbox="607 1041 721 1062">Original Image</p>  <p data-bbox="639 1302 688 1323">2a.jpg</p>	<p data-bbox="867 1041 964 1062">True Mask</p> 	<p data-bbox="1062 1041 1289 1062">Predicted Mask/IOU=0.4853</p> 
3	<p data-bbox="607 1386 721 1407">Original Image</p>  <p data-bbox="639 1646 688 1667">3a.jpg</p>	<p data-bbox="867 1386 964 1407">True Mask</p> 	<p data-bbox="1062 1386 1289 1407">Predicted Mask/IOU=0.9534</p> 

Table 4-6: Network was trained on the lowest scale for the door dataset.

Table 4-6 shows that the network struggles to segment the door at the highest scale with an IOU score of 0 however, the IOU score on the lowest scale that the network was trained on is 0.95.


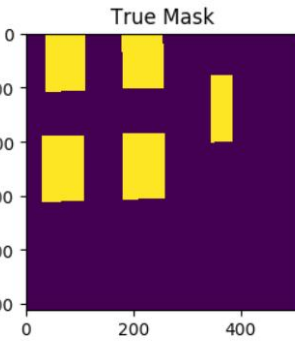
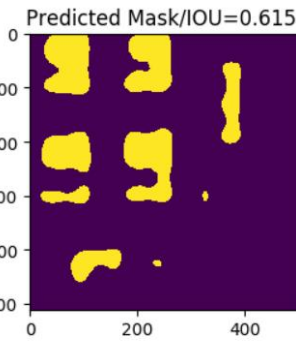
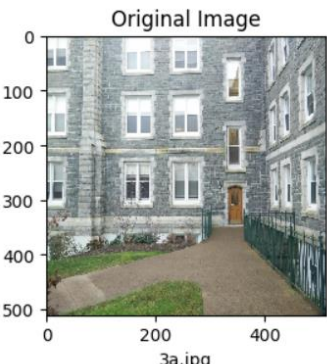
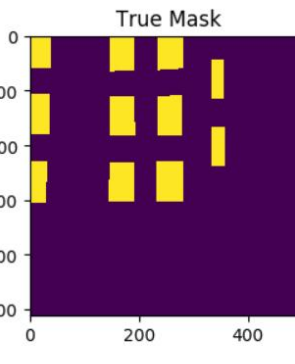
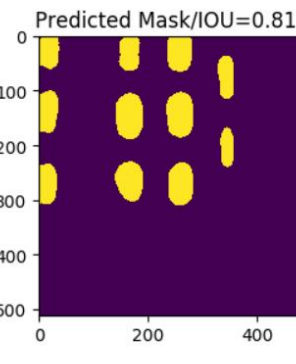
Scale	Results		
2			
3			

Table 4-7: Network was trained on the lowest scale for the window dataset.

From Table 4-7, the difference in IOU score across spatial resolutions is not as significant as seen for door and pond. This is due to the difference in the spatial gap, e.g., 20m to 200m has a larger spatial gap than 50m to 100m; therefore the network should produce better predictions since the spatial features learned would be in close proximity of the spatial features being evaluated on (segmenting). Window only has data for scale 2 and 3 hence training will be performed using 2 and segmentation done on the next spatial



resolution, which is 3. The door has a larger spatial gap since training is performed using scale 1, then segmentation done on scale 3. This indicates that there is a cut-off or drop-off point where the kernels are unable to detect features beyond a certain scale from what was trained on.


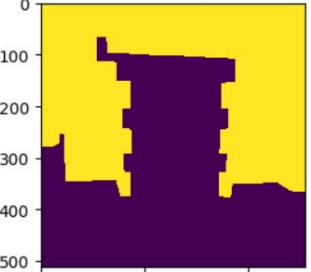
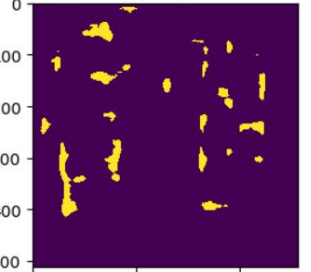

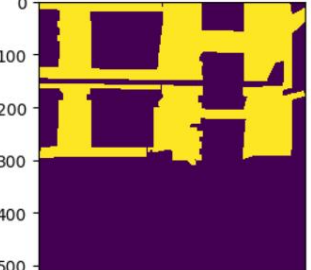
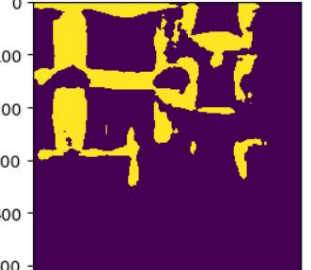
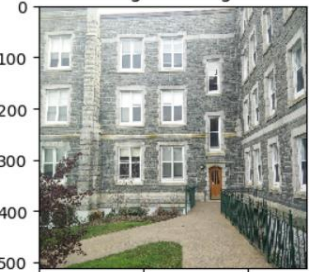
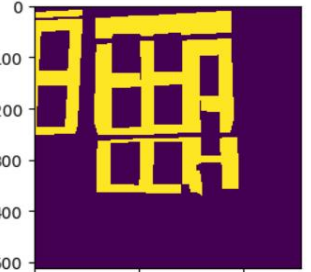
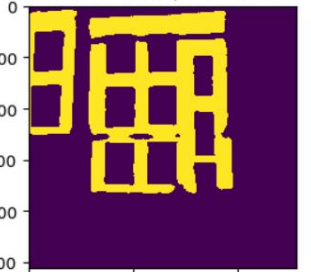
Scale	Results		
1	 <p>Original Image 1a.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.0566</p>
2	 <p>Original Image 2a.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.4484</p>
3	 <p>Original Image 3a.jpg</p>	 <p>True Mask</p>	 <p>Predicted Mask/IOU=0.9274</p>

Table 4-8: Network was trained on the lowest scale for the wall dataset.

The results in Table 4-8 highlights the scale problem again. An interesting finding is that the IOU score for door and wall is the same across all three spatial resolutions. Scale 1 has 0, Scale 2 has 0.45 and Scale 3 has 0.9. Both regions are different in terms of features such as structure, color, texture and even the number of pixels. The number of pixels for the door will reduce as we move further away due to pixel aggregation; however, wall's pixel count will increase since more regions of the wall become available the further away we are, yet the IOU score across all scales is the same.

The experiments show that CNNs generalize poorly for spatial resolutions not trained on. Training on higher scales does not generalize well on lower scales, while training on lower scales does not generalize well on higher scales. Consider the scenario of the farmer, where training images are from one scale only (ground). Images captured on-ground will contain features at a higher spatial resolution. However, aerial images will contain features at a lower spatial resolution, therefore, segmenting the region of interest will be difficult resulting in poor predictions. If training data is available at the highest scale, then a method should be used to generate data at lower scales.

## 4.2 Training on Zoom Augmented (ZA) Image

For this experiment, the highest resolution image will be selected as training data. The methods demonstrated in this research are only applicable to higher spatial resolution images. Transitioning from a lower resolution to a higher resolution will require additional data, see section 3.4. The window and wall dataset were not used for the upcoming set of experiments. The window dataset only consisted of two scales which is not applicable to the proposed contribution of this research. The purpose of the wall and window dataset was to demonstrate the scale issues as seen in early sections of Chapter 4. The Pond, Door, and Grass dataset were the focus of this research hence further experiments were conducted using them.

Zoom Augmentation (ZA) will be used to downscale the image such that the region of interest of the augmented image would have similar structure or arrangement of pixels as the region of interest in the next spatial resolution. In short, ZA allows us to simulate or approximate the pixel arrangement used to construct the region of interest at the next spatial resolution.

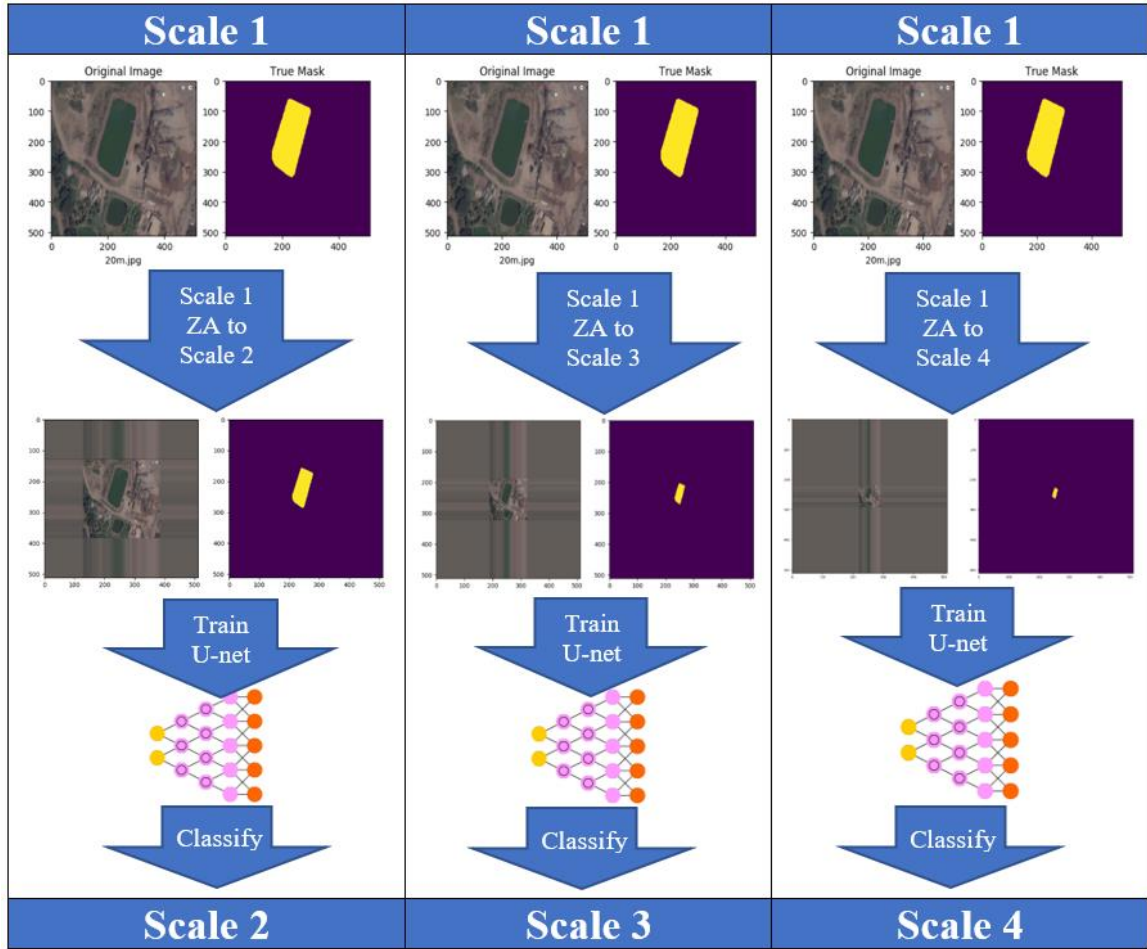


Figure 4-1: Zoom Augmentation process.

In Figure 4-1, the image at the highest scale is zoom augmented to the lower scales using a scaling factor. Recall section 3.4 which shows that downscaling (zoom augmentation) reduces the number of pixels in the image; therefore the image resolution will also be reduced. However, pre-runs prior to these experiments indicate that the same resolution should be used for training and predicting, else results will differ drastically across image resolutions.

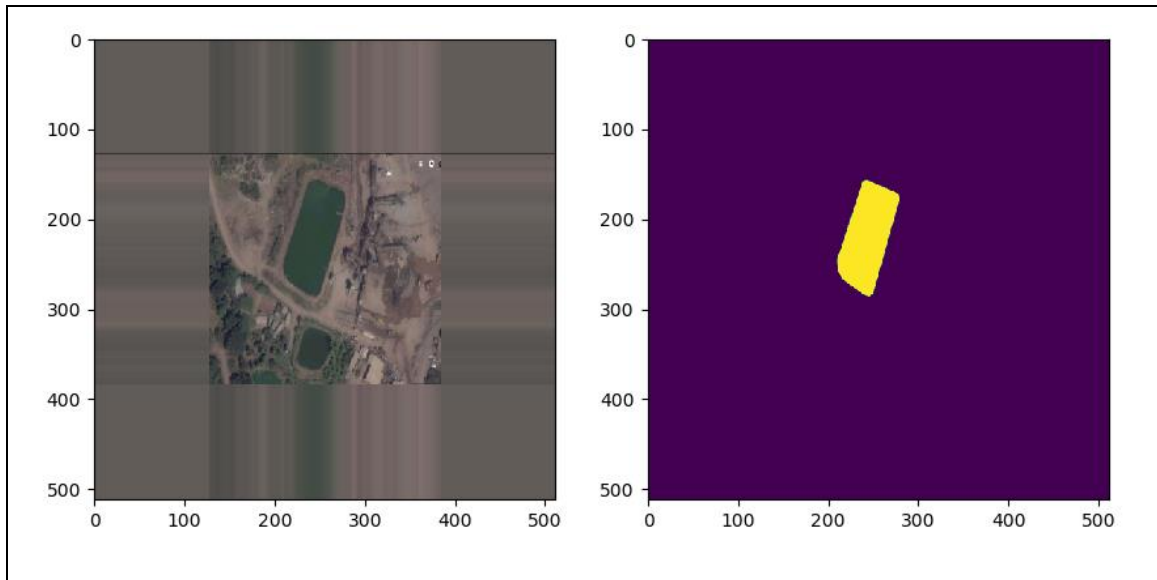


Table 4-9: Zoom Augmented image with mean padding and corresponding mask.

Several methods were used to pad the zoom augmented image with additional pixels such that the resolution would remain constant. Padding the image with zero values results in frequent failures during training and low IOU score for predictions. The method of choice to pad the image was mean padding which adds the mean of the row or column values calculated across all channels (Van der Walt et al., 2014). An important point to note is that the pixel count for the ROI remains the same after padding. See Appendix 8.3 for the results from each padding method.

### 4.2.1 Zoom Augmentation Experiment for Pond Dataset

The experiments shown in this section is for the pond dataset which contains an oval-shaped pond as the region of interest.

Spatial Resolution	Original px	Zoom Augment	Scaling factor	Zoomed px
20m	21624	-	-	-
50m	5679	20m to 50m	0.5	5511
100m	1571	20m to 100m	0.265	1547
200m	431	20m to 200m	0.14	431

Table 4-10: Pixel (px) count for ROI at each scale.

Table 4-10 shows the pixel count for the region of interest at each spatial resolution, the scaling factor used for ZA and the corresponding pixel count for the zoom augmented image. For example, using the third row from Table 4-10, the ROI at 50m has a pixel count of 5679. Therefore a scaling factor of 0.5 was applied to the ROI at 20m such that the pixel count moves from 21624 to 5511.

As mentioned previously, we want to simulate or approximate the region of interest at the different spatial resolutions. The number of pixels used in the construction of the object will reduce the further away you are from the object due to pixel aggregation. Although pixel count is used; spatial resolution is not measured by the number of pixels; however, the number of pixels influences how the kernels learn features.

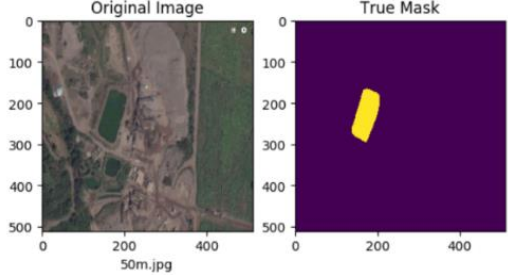
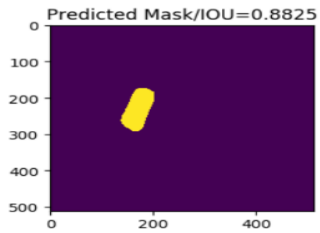
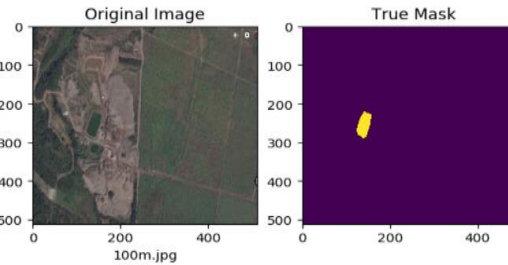
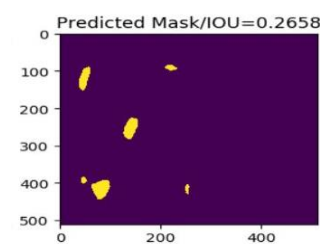
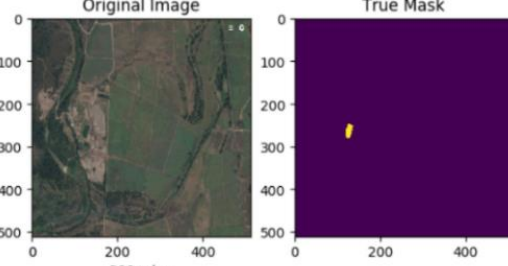
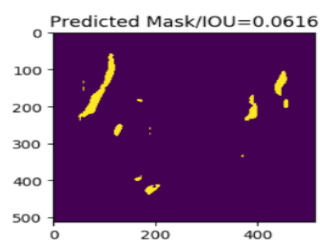
ZA	Training Image	Prediction
20m to 50m		
20m to 100m		
20m to 200m		

Table 4-11: Predictions from three networks trained on ZA image.

From Table 4-11, the highest spatial resolution image at 20m is zoom augmented to 50m, 100m, and 200m, and then each ZA image was used to train a neural network. The predictions produced by 20 to 50m improves by 0.2, recall from Table 4-1, the IOU score was 0.7 and 0 for scales 50m and 100m respectively. However, using ZA increase the IOU score to 0.88 and 0.26. It is not possible to train on an infinite amount of scales;

therefore, a discrete number of scales were selected based on visual inspection by the researcher, see section 3.3.

Despite the increase, the 20m to 100m and 20m to 200m networks produce very low IOU scores which is due to multiple false positives being predicted.

#### 4.2.2 Zoom Augmentation Experiment for Door Dataset

Spatial Resolution	Original px	Zoom Augment	Scaling factor	Zoomed px
1	29741	-	-	-
2	4687	1 to 2	0.395	4705
3	1739	2 to 3	0.265	1766

Table 4-12: Pixel (px) count for ROI at each scale.

Table 4-12 shows the pixel count for the door from the original image and the zoom augmented image. The scaling factors were selected such that the pixel count of the zoom augmented image is similar to that of the ROI at the next scale. However, there is no perfect scaling factor as identical results can be produced using a scaling factor of 0.5 and 0.3. A conservative scaling factor of 0.5 can be used based on observations made during these experiments. Typically, a scaling factor less than 0.5 will result in frequent failures during training and low IOU scores; this is as a result of the significant loss in information from using a small scaling factor.

In a real-world application, the aerial device used such as a drone should be able to record its altitude. The change or difference between altitudes can be used to determine the scaling factor. Scaling factors are used to approximate the ROI at the next scale; an approximation cannot be precise.




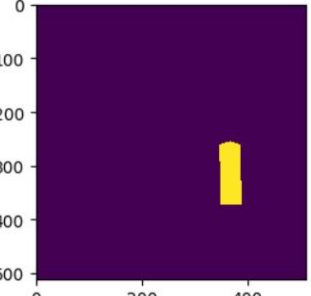
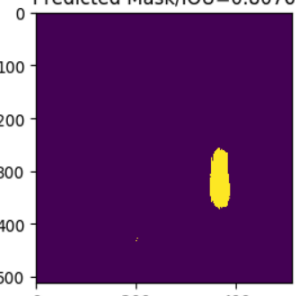

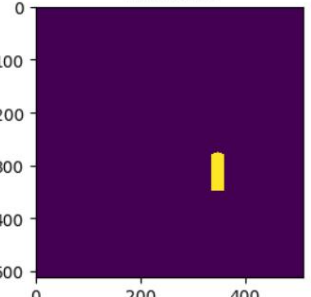
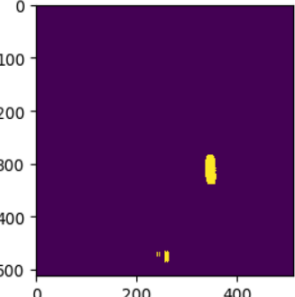
ZA	Training Image	Prediction
1 to 2	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Original Image</p>  <p>2a.jpg</p> </div> <div style="text-align: center;"> <p>True Mask</p>  </div> </div>	<div style="text-align: center;"> <p>Predicted Mask/IOU=0.8076</p>  </div>
1 to 3	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Original Image</p>  <p>3a.jpg</p> </div> <div style="text-align: center;"> <p>True Mask</p>  </div> </div>	<div style="text-align: center;"> <p>Predicted Mask/IOU=0.566</p>  </div>

Table 4-13: Predictions from two networks trained on ZA image.

From Table 4-13, the IOU scores are higher compared to the ROI from the pond dataset shown in Table 4-11, the lowest scales have IOU scores of 0.06 and 0.56. This is due to the pond dataset having a larger spatial gap which means that the loss in information when zoom augmenting from the highest to the lowest scale will be greater; the more information we lose, the harder learning becomes – frequent failures during training and the lower the IOU score. To contrast, Table 4-2 does not use ZA and produces IOU scores 0.66 and 0.15 for scales 2 and 3, however, using ZA, Table 4-13 shows that the IOU score increases to 0.8 and 0.56 respectively.

### 4.3 Issues with Zoom Augmentation

It is evident that ZA or downscaling provides improvement for predictions at coarser scales, however, if there is a significant difference in terms of scale between the finest and coarsest image resolution - referred to as a large spatial gap, then ZA performs poorly. The further we zoom augment; the more contextual information is lost during this process.

Context refers to the background portion of the image – everything excluding the region of interest. In the U-Net architecture, Ronneberger et al., stated that the contracting path of the network captures context information while the expanding path propagates the contextual information to higher resolution layers (Ronneberger et al., 2015). This contextual information is needed for the network to produce accurate predictions.

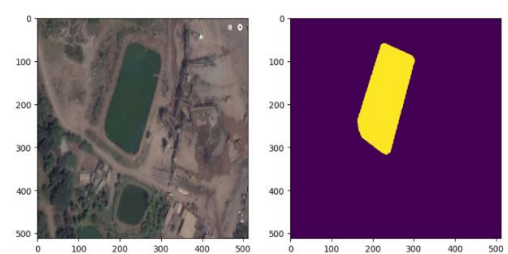
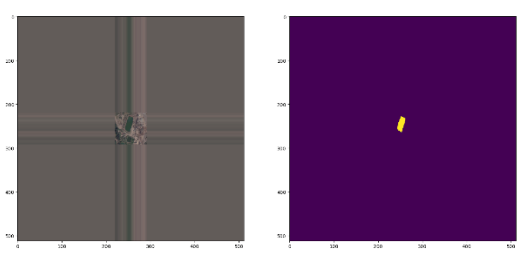
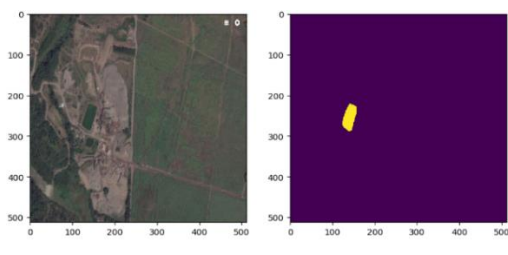
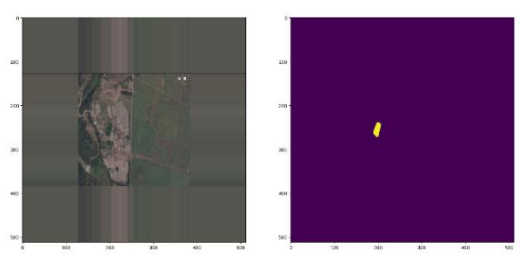
Scale	Original Image	ZA to 200m
20m		
100m		

Table 4-14: An illustration of the loss in contextual information when using ZA.

From Table 4-14, the loss in contextual information is significant when zoom augmenting from 20m to 200m versus 100m zoom augmented to 200m; this results in a reduction in the IOU score. Instead of performing large steps using ZA, we should incrementally step across spatial resolutions in an effort to preserve as much contextual information as possible. The training times for each dataset and final epoch can be seen in Appendix 8.7. The time for data preparation was not recorded as it was minuscule and was completed in a few seconds.

#### **4.4 Waterfall Method**

To preserve as much contextual information from the highest spatial resolution, we will take small steps towards the coarsest spatial resolution. The step size can be determined based on the difference or change in scale between the region of interest. Starting from the initial layer, the highest resolution image is zoom augmented to the second spatial resolution, and then a network is trained on the zoom augmented image.

Predictions are then performed at the second spatial resolution where the results will be used as training data for the third spatial resolution, and so on; all succeeding layers will receive training images from the predictions of preceding layers – called layer feeding, this hierarchy resembles a waterfall.

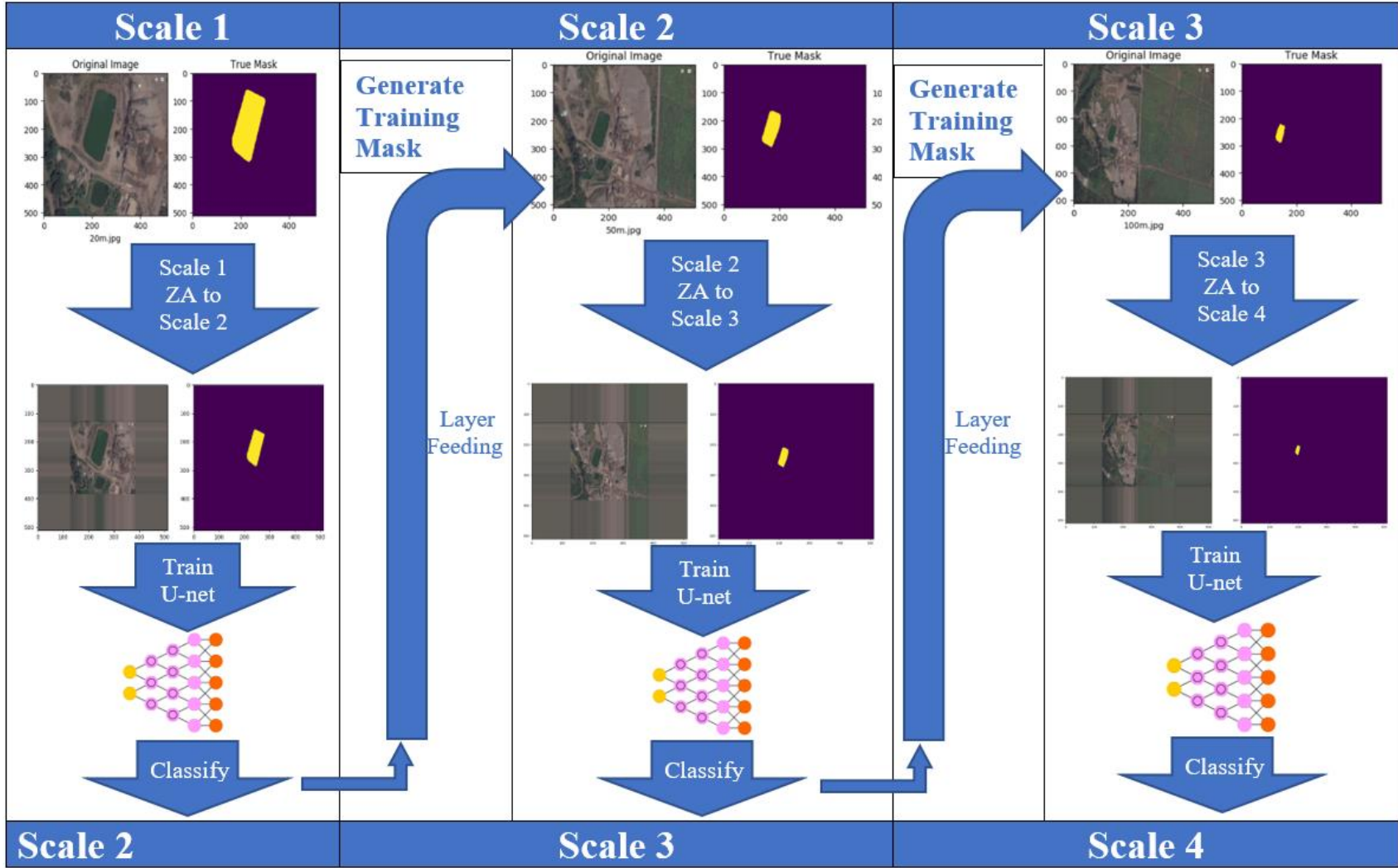


Figure 4-2: An illustration of the Waterfall Method.

From Figure 4-2, instead of performing ZA from the highest to the lowest scale, ZA is performed from scale  $i$  to scale  $i + 1$ , that is the current scale is only zoom augmented to the next scale. This ensures that less contextual information is lost during zoom augmentation; rather than zooming from scale 1 to 4, ZA is performed from  $s_1$  to  $s_2$ ,  $s_2$  to  $s_3$ , then  $s_3$  to  $s_4$ .

#### **4.4.1 Waterfall Experiments**

The experiments from this section will show predictions from each layer in the Waterfall Method, where a layer represents a spatial resolution for the region of interest. Frequent comparisons will be made between Zoom Augmentation (ZA) only and Waterfall Method (WM). It is worth mentioning that the WM uses zoom augmentation to step from one resolution to the next.

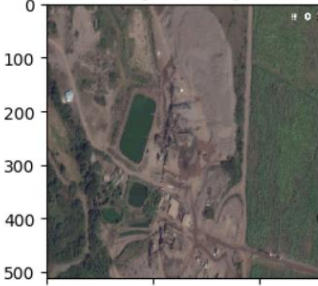
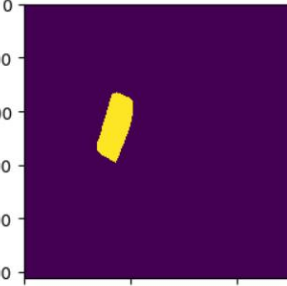
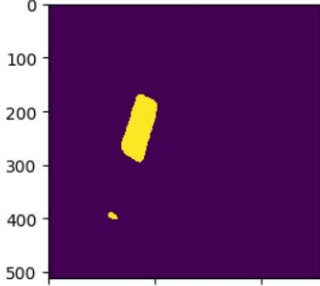
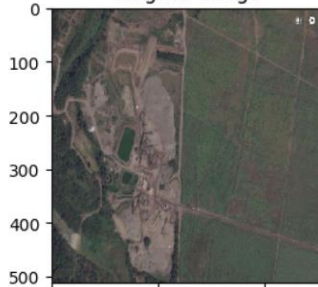
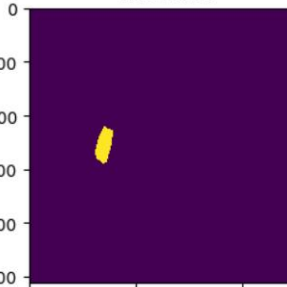
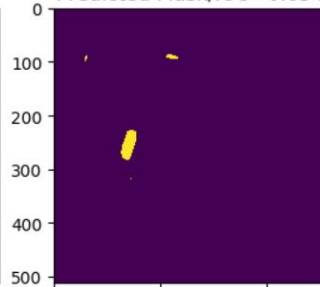
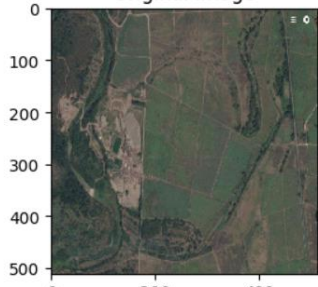
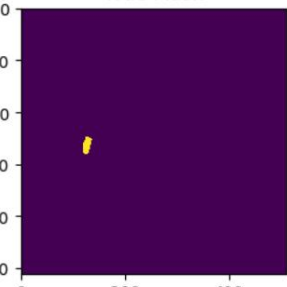
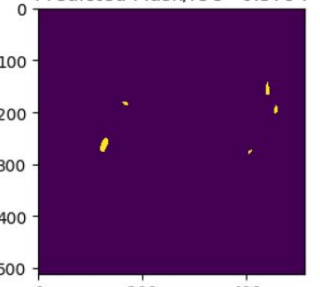
Scale	Results		
2	<p data-bbox="553 386 703 411">Original Image</p>  <p data-bbox="591 716 662 741">50m.jpg</p>	<p data-bbox="899 386 1003 411">True Mask</p> 	<p data-bbox="1138 386 1422 411">Predicted Mask/IOU=0.8954</p> 
3	<p data-bbox="553 848 703 873">Original Image</p>  <p data-bbox="591 1178 662 1203">100m.jpg</p>	<p data-bbox="899 848 1003 873">True Mask</p> 	<p data-bbox="1138 848 1422 873">Predicted Mask/IOU=0.634</p> 
4	<p data-bbox="553 1289 703 1314">Original Image</p>  <p data-bbox="591 1608 662 1633">200m.jpg</p>	<p data-bbox="899 1289 1003 1314">True Mask</p> 	<p data-bbox="1138 1289 1422 1314">Predicted Mask/IOU=0.3794</p> 

Table 4-15: Predictions from each scale in WM.

In Table 4-15, the predictions from the lowest scales are shown, recall that the training data used was from the highest scale (20m). For comparison, using ZA and WM,

the IOU score remains the same at 0.89 for the 50m scale; however, the 100m and 200m scales differ drastically. The IOU score for ZA is 0.26 and 0.06 respectively while it is 0.63 and 0.37 for WM as seen in Table 4-15.


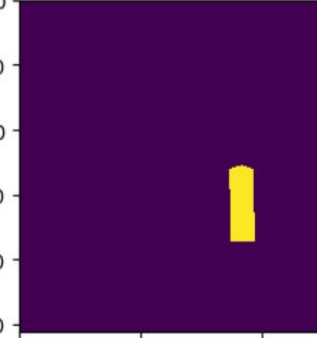
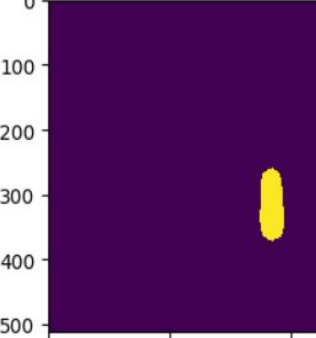
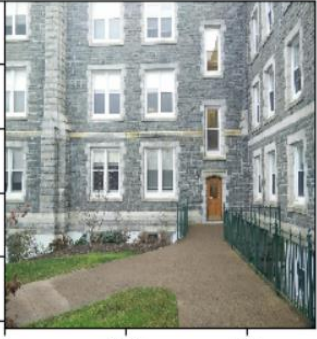
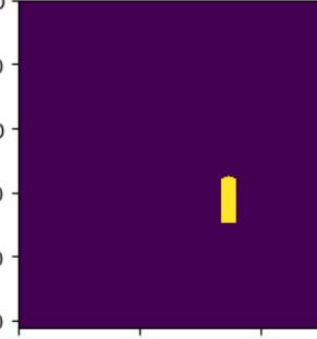
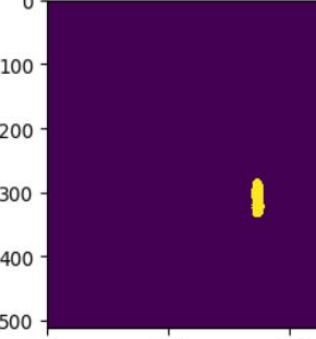
Scale	Results		
2	<p data-bbox="483 554 646 583">Original Image</p>  <p data-bbox="532 957 597 987">2a.jpg</p>	<p data-bbox="873 554 992 583">True Mask</p> 	<p data-bbox="1149 554 1463 583">Predicted Mask/IOU=0.8062</p> 
3	<p data-bbox="483 1073 646 1102">Original Image</p>  <p data-bbox="532 1476 597 1505">3a.jpg</p>	<p data-bbox="873 1073 992 1102">True Mask</p> 	<p data-bbox="1149 1073 1463 1102">Predicted Mask/IOU=0.6451</p> 

Table 4-16: Predictions from each scale in WM.

Like Table 4-15, the IOU score for the highest spatial resolution is the same. It is 0.8 for scale 2 using ZA and WM. An increase of 0.1 in IOU score is seen for scale 3 using WM. ZA produces 0.54 while WM produces 0.64.

The spatial gap is smaller for the door dataset versus the pond dataset; therefore, the loss in contextual information will not be as significant, resulting in higher IOU scores at coarser scales. WM does give an improvement in IOU score when compared to ZA. It can range from 0.1 to 0.4; however, this depends on the problem. The difference in IOU score using WM versus ZA is noticeable for coarser spatial resolutions. This is because, at coarser scales, WM preserves context while ZA discards majority or all of the contextual information.

The two benefits of using WM are:

- *Layer Feeding* - layer feeding allows us to pass along the information about the region of interest to lower scales, thereby allowing us to retain as much contextual information about the region of interest.
- Each model trains and predict at *one scale*, which is ideal for fixed size kernel since they work best with single scale semantics

Although other researchers such as (Raj et.al, 2015) and (van Nord & Postma, 2017) have used multi-scale networks to tackle scale invariance their application or use case is different. Van Nord's task was classification which is assigning a label to an image based on global information while segmentation assigns a label to each pixel based on local information. There is no layer feeding in their ensemble, but this is because fine detail contextual information is not as important for the task of classification compared to segmentation. Classification operates on coarse information while segmentation requires fine detail information that allows for accurate predictions to be made at a pixel-wise level and not coarse or image-wise. The method proposed by (Raj et.al, 2015) requires



additional depth information such that a close approximation can be made, but this requires special equipment which may not be available to the user.

#### 4.4.2 Ensemble Learning - Bootstrap Aggregation (Bagging)

The Waterfall Method (WM) has few similarities and differences between ensemble learning methods. Ensemble methods use multiple learning algorithms to improve the prediction score. There are several ensembles learning methods but the most similar method to WM is Bootstrap Aggregation, also called bagging.

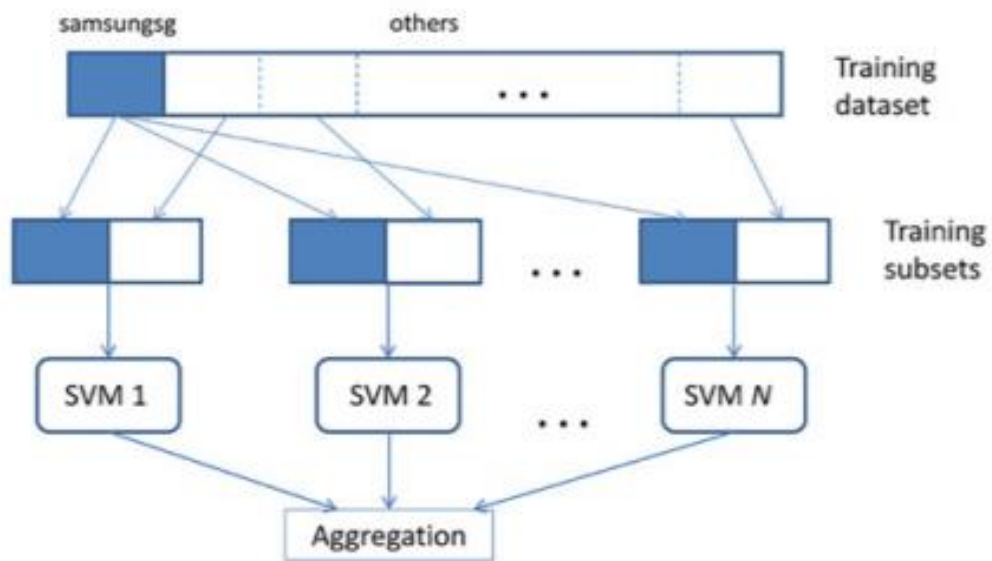


Figure 4-3: An ensemble of SVMs. Retrieved from “Using Support Vector Machine Ensembles for Target Audience Classification on Twitter.” (Lo, Chiong, & Cornforth, 2015, p. 9)

Figure 4-3 gives an overview of the bootstrap aggregation methodology, in this case, SVMs were used. Bootstrap aggregation involves training several models by using a subset of the training data, the predictions from each model is averaged into a final score. There are a few similarities and differences between bagging and WM. The methods are

similar in that they both train several models and each bag in the bagging method is equivalent to a scale in WM; however there are several differences.

The complete training data is not available initially (data from all scales) therefore subsets cannot be created to train different models. Instead, each model provides training data for the next model. This connection between models is not present in bagging however in WM, each model except the first relies on the previous model to provide its training data.

Finally, bagging aggregates or averages the predictions across all models, however, though result can be averaged from WM, more importance is placed on the IOU score for the coarsest scale since that is the weak point of CNNs.

#### **4.5 Summary of Chapter 4**

The results from chapter 4 indicate that CNNs struggle with handling objects at varying scales. To improve the network's robustness to variations in scale, ZA can be used to simulate the region at a lower scale. Although ZA, provides an improvement in prediction accuracy, the performance of the network reduces significantly at lower scales. A Waterfall Method (WM) was suggested which yields more accurate results compared to ZA.

# Chapter 5

## Results

The experiments in this chapter will be a comparison between ZA and WM. Results are quantitative where IOU was used as the metric to compare both methods. The aim is to identify which method provides the most accurate predictions at the coarsest scale and highlight issues or weaknesses of using WM.

### 5.1 Waterfall Method vs Zoom Augmentation

Zoom Augmentation (ZA) can be used to circumvent the scale issue experienced by CNNs. However, zoom augmenting directly from the highest to the lowest scale is “naïve” as the loss of contextual information is significant which results in low IOU scores and difficulties during training. WM allows us to move from the highest to lowest scale by retaining as much contextual information as possible. This chapter will assess which method produces the most accurate and consistent IOU scores.

#### 5.1.1 ZA vs WM Experiment for Pond Dataset.

Due to the stochastic nature of neural networks, five runs were performed and the IOU score was averaged across all runs for the respective scales. The tables contain five IOU scores for each scale except the scale trained on, i.e., the IOU score for scale one across all datasets will be ignored.

<b>ZA</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0.671888	0.298976958381	0.0
<b>2</b>	0.795376	0.256107045301	0.0
<b>3</b>	0.77811	0.372314150521	0.0111827351035
<b>4</b>	0.791835	0.372607861443	0.128241664819
<b>5</b>	0.77091	0.172828927757	0.0455051160734
<b>AVERAGE</b>	<b>0.7616238</b>	<b>0.294567</b>	<b>0.036986</b>

Table 5-1: Network was trained on only the highest spatial resolution for the pond dataset using ZA.

The predictions for each scale using ZA on the pond dataset is shown in Table 5-1. There is a 60% decrease in the IOU score at scale 3 with 0.29, versus scale 2 at 0.76 which highlights the significant loss in contextual information. There were no predictions made for scale 4, the IOU score was 0.03

<b>WM</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0.806615097199	0.436562838212	0.389409579147
<b>2</b>	0.570166713548	0.502918415152	0.261629749784
<b>3</b>	0.802865072335	0.527063030362	0.332860934666
<b>4</b>	0.803789139207	0.497134642537	0.439680070691
<b>5</b>	0.823985379601	0.319441976938	0.310481786915
<b>AVERAGE</b>	<b>0.76148428</b>	<b>0.456624</b>	<b>0.346812</b>

Table 5-2: Network was trained on the highest spatial resolution for the pond dataset using WM.

Table 5-2 illustrates the improvement in the preservation of contextual information as we step across spatial resolutions. There is a 40% decrease in IOU when

moving from scale 2 to scale 3, compared to the 60% using ZA. Even at the coarsest scale (scale 4), the IOU score increased to 0.34 from 0.03 using WM.

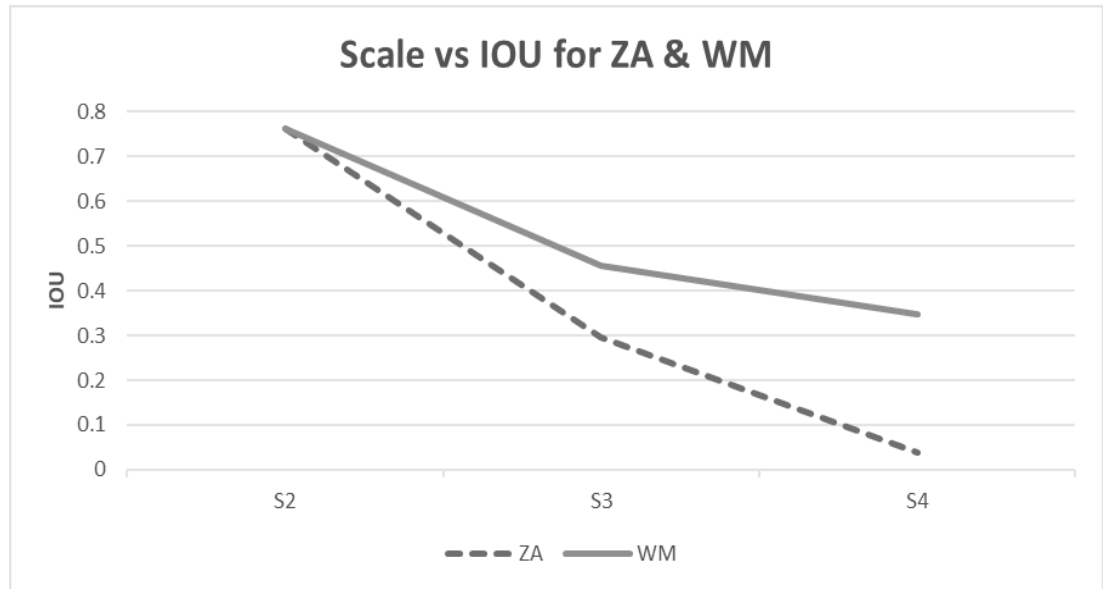


Figure 5-1: Comparison of the IOU across all scales (2, 3, 4) between ZA and WM.

From Figure 5-1, WM outperforms ZA at scales 3 and 4. The IOU score does not drop below 0.4 with WM, however using ZA the IOU score falls below 0.4 at scale 3.

### 5.1.2 ZA vs WM Experiment for Door Dataset.

The region of interest in this experiment was the door.

<b>Zoom Augment</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.755377986192	0.523978725137
<b>2</b>	0.785577779653	0.610888620163
<b>3</b>	0.716548674317	0.533183327087
<b>4</b>	0.768932974832	0.612936494778
<b>5</b>	0.784905106288	0.597264107258
<b>AVERAGE</b>	<b>0.762269</b>	<b>0.57565</b>

Table 5-3: Network was trained on only the highest spatial resolution for door using ZA.

The reduction in IOU score for the door dataset is less in comparison to the pond dataset. This is because the spatial gap or difference in spatial resolution for the region of interest is smaller; therefore less information will be lost when moving from the highest to the lowest scale.

<b>WM</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.753696785647	0.535421992341
<b>2</b>	0.811700857853	0.418043224287
<b>3</b>	0.832152141446	0.378368088879
<b>4</b>	0.810602905292	0.441032139817
<b>5</b>	0.856163909258	0.511807573824
<b>AVERAGE</b>	<b>0.812863</b>	<b>0.456935</b>

Table 5-4: Network was trained on only the highest spatial resolution for door using WM.

There is a small increase in the IOU score for scale 2, but what should be highlighted is that the IOU score has decreased for scale 3 when using WM. The IOU score for scale 3 using ZA is 0.57 while it is 0.45 for WM, a 0.1 decrease.

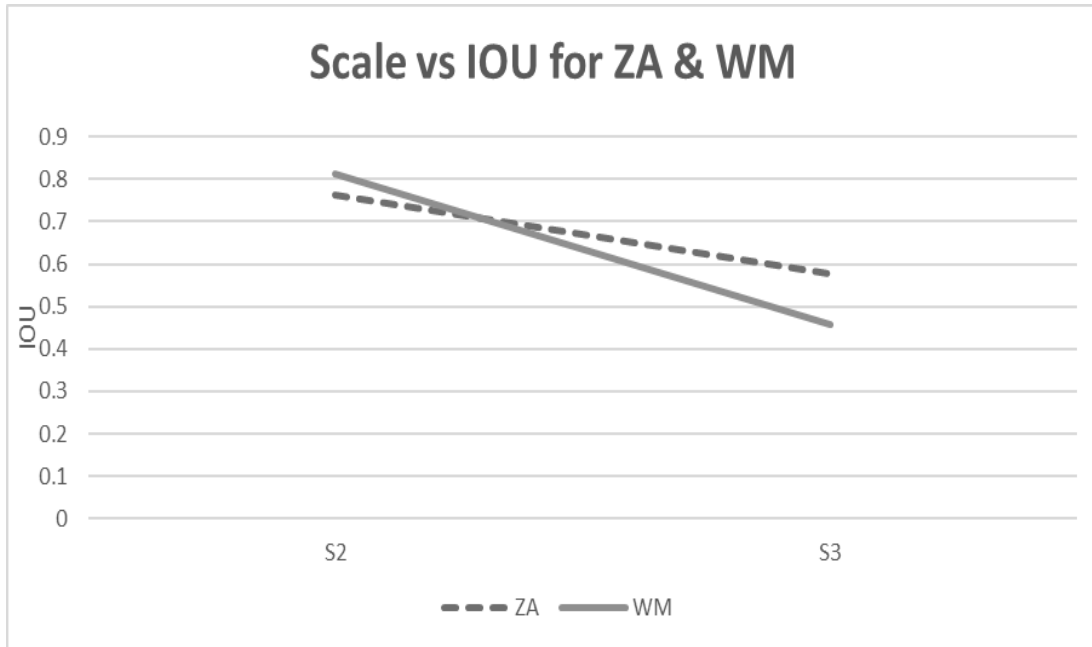


Figure 5-2: Comparison of the IOU across all scales (2, 3) between ZA and WM.

From Figure 5-2, the line graph illustrates that WM performs worst than ZA at the lowest scale. Further investigations were done to find the source of the problem.

Although there is a marginal difference in IOU score, WM is still expected to outperform ZA. This issue will be further investigated in section 5.2.

## 5.2 Issues with the Waterfall Method

The experiments highlight a problem with WM which results in lower IOU scores than the naïve approach – ZA.

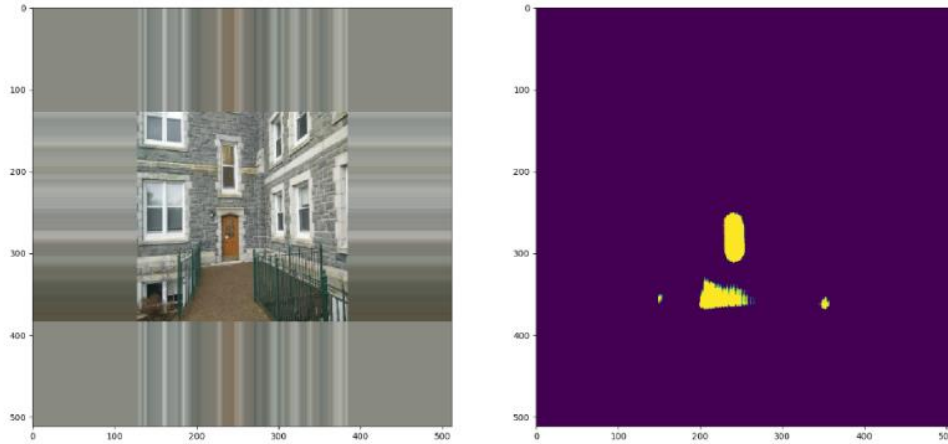


Figure 5-3: Training image used to train scale 3 of WM.

Figure 5-3 contains the prediction from scale 2 that will be used to train scale 3. The massive false positive in this image causes training at scale 3 to be difficult; it's almost confusing the network. This results in poor IOU scores when predicting at scale 3 since the network has learned the false positives from the previous layer and is propagating that incorrect information along the pipeline to other scales.



Scale	Results
3	<p>Original Image: 3b.jpg. True Mask: A single vertical yellow bar. Predicted Mask/IOU=0.0565: A yellow bar with significant false positives (yellow pixels) in the lower half of the image.</p>
3	<p>Original Image: 3c.jpg. True Mask: A single vertical yellow bar. Predicted Mask/IOU=0.1288: A yellow bar with false positives (yellow pixels) in the lower half of the image.</p>
3	<p>Original Image: 3d.jpg. True Mask: A single vertical yellow bar. Predicted Mask/IOU=0.1972: A yellow bar with false positives (yellow pixels) in the lower half of the image.</p>

Table 5-5: Network was trained on predictions from scale 2 using WM.

From the Table 5-5, the IOU scores are low for all training points at the 3<sup>rd</sup> scale. A large portion of predictions is false positives which are because of that misinformation from the previous layer. The network appears to be overfitting to the color of the door as the false pixels have similar color values to the door.

To reduce the overfitting to color, images that emphasize the structural features (edges) of the region must be introduced. This ensures that the network learns other

features, separate from color, for predicting the region of interest such as the structural arrangement of pixels (shape). Different image preprocessing techniques can be used to highlight features in an image.

### 5.2.1 Sobel Filter for all Scales

According to Rhineland, Sobel filtering (SB) can be used to enhance the edges of images (Rhineland, 2016). The research showed that the accuracy of his SVM classifier for side-scan sonar images increased most significantly when a Sobel filter is used.

<b>WM</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.918325476	0.690978554311
<b>2</b>	0.912171217	0.702808515524
<b>3</b>	0.903284112935	0.829222724282
<b>4</b>	0.908378909672	0.537623400318
<b>5</b>	0.913874752	0.481855278049
<b>AVERAGE</b>	<b>0.911206894</b>	<b>0.648497694</b>

Table 5-6: Network was trained on the highest spatial resolution for door using WM with Sobel Filter applied.

From Table 5-6, there is an increase in the IOU score for both scales. It increases to 0.91 from 0.81 at scale 2, and 0.64 from 0.45 at scale 3. There is a definite improvement in the IOU score when a Sobel image is used, however including a Sobel filtered image at coarser scales may affect the IOU score.

The IOU scores differ drastically across all five runs at scale 3, with the lowest being 0.48 and the highest 0.82. The inconsistent scores indicate that extracting and

learning features from the Sobel image at that scale is difficult for the network. This results in noisy training (Large Oscillations) and frequent failures.

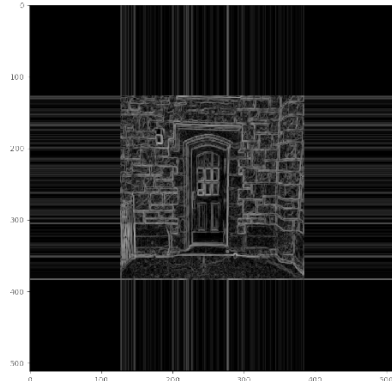
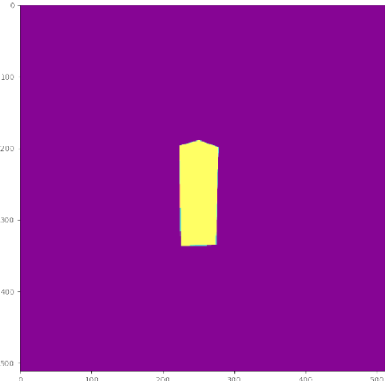
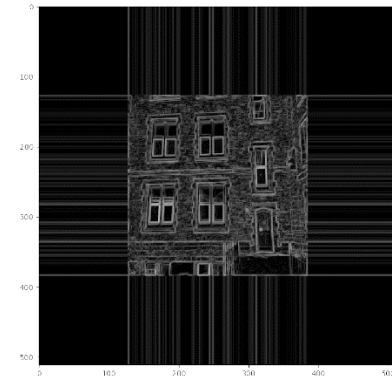
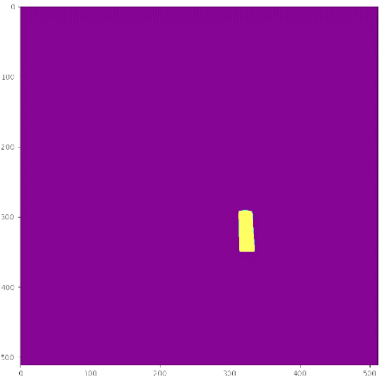
Scale	Results	
2		
3		

Table 5-7: Sobel filtered door for scale 2 and 3.

The visibility and clarity of the region of interest are low at scale 3. This results in high oscillations during training, in other words, the network cannot extract the necessary features for the identification of the ROI.

### 5.2.2 No Sobel Filter for Coarsest scale

WM	Door Dataset	
Run	Scale 2	Scale 3
1	0.89109862	0.733816022346
2	0.900329057	0.715015361411
3	0.913265374	0.731744250391
4	0.920293525926	0.690411824337
5	0.88667578863	0.554593685666
<b>AVERAGE</b>	<b>0.902332473</b>	<b>0.717747</b>

Table 5-8: Network was trained on the highest spatial resolution for door using WM, no Sobel filter was used at scale 3.

From Table 5-8, the IOU score increases for the 3<sup>rd</sup> scale, from 0.64 to 0.71. This illustrates that including the Sobel filtered image at the 3<sup>rd</sup> scale reduces the IOU score. Furthermore, the IOU scores are more consistent with the lowest being 0.55 and highest is 0.73. No further experimentation was done using Sobel filtering on the remaining datasets as this was not the focus of the research.

### 5.2.3 Contrast Enhancement using Histogram Equalization

To combat the false positive issue that plagued door, regions can be made more *defined* using contrast enhancement. The contrast of an image refers to the difference in luminance of regions in the image. By making regions more distinguishable from adjusting luminance (pixel intensities), features can be learnt easier to discriminate between objects. According to Bora, the advantage of contrast enhancement is that it removes ambiguity or uncertainty between different regions in the image (Bora & Gupta,

2016). Histogram Equalization (HE) enhances the contrast of an image by adjusting the pixel intensity values.

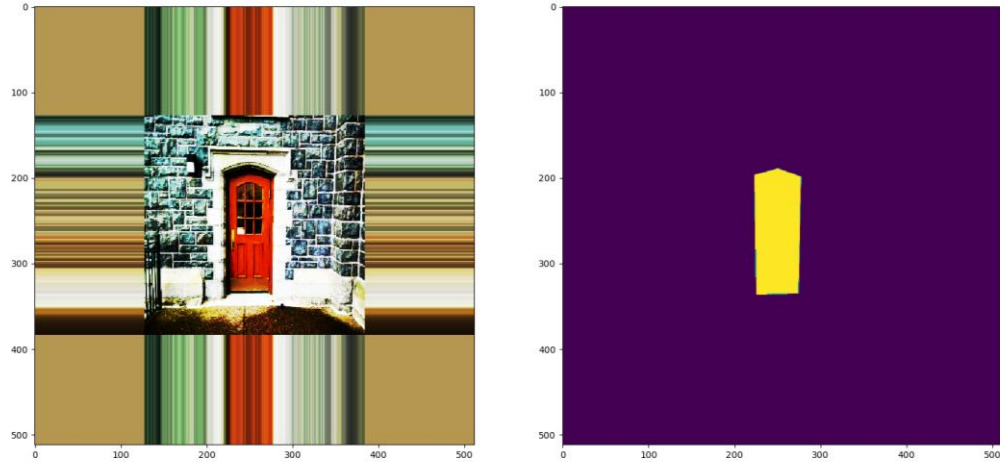


Figure 5-4: Histogram equalized image used to train scale 2.

In Figure 5-4, the door appears to be more defined and separated from the different regions such as the wall, and the ground which the networking was mis-segmenting.

WM	Door Dataset	
Run	Scale 2	Scale 3
1	0.902338099	0.762798307992
2	0.902661542	0.799544890362
3	0.921064694	0.817921326584
4	0.93043777	0.726590352478
5	0.92677762	0.769605002381
<b>AVERAGE</b>	<b>0.916656</b>	<b>0.775292</b>

Table 5-9: Network was trained on the highest spatial resolution for door using WM with HE.

From Table 5-9, the IOU score for scale 2 remains the same when using a Sobel filtered image. However, including the HE images at the coarsest scale increases the IOU

score from 0.71 to 0.77. Although the increase is not significant, the consistency of the scores has improved, the lowest is 0.72 and the highest is 0.81. Furthermore, there is a significant improvement in IOU score for scale 3 between ZA and WM with HE. Using ZA, it is 0.57 while HE produces 0.77.

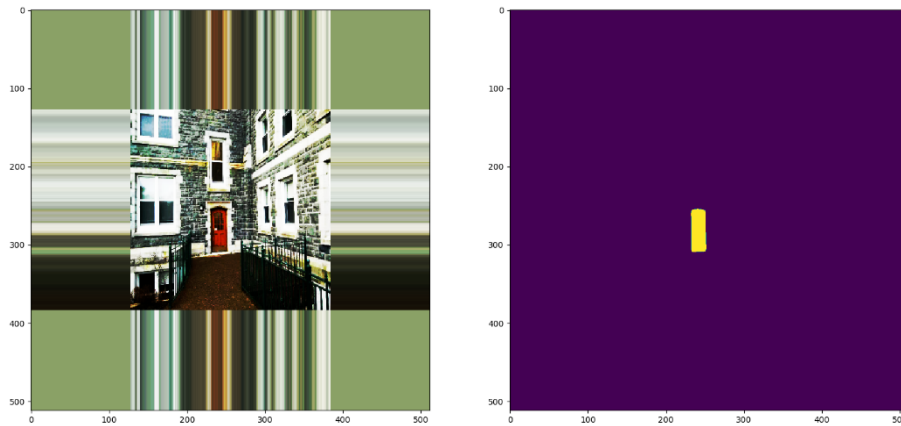


Figure 5-5: Histogram equalized image used to train scale 3.

These experiments highlighted an issue with WM, which is that false positives in early stages will be propagated to later stages resulting in low IOU scores. This can be counteracted by including a Sobel filtered or histogram equalized image at the highest scale which allows for structural features to be learned, thereby reducing false positives made from overfitting to color information.

However, including Sobel filtered images for coarser scales might be detrimental to IOU score; instead a histogram equalized image should be used for all scales.

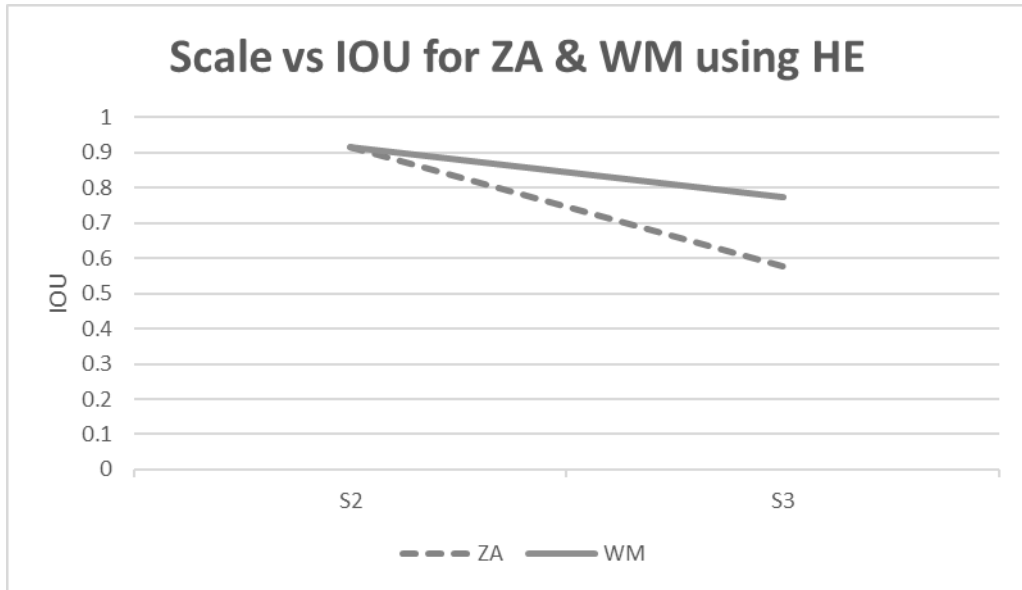


Figure 5-6: Comparison of the IOU across all scales (2, 3) between ZA and WM.

From Figure 5-6, after resolving the false positive problem for the door, WM continues to outperform ZA. At the coarsest scale, ZA produces 0.57 while WM gives 0.77. Despite WM performing poorly when false positives are present in the training data, image filtering or preprocessing techniques such as Sobel filtering and histogram equalization can be used to combat this issue.

### 5.3 ZA vs WM Experiment for Grass Dataset

This dataset contains images of grass with unhealthy regions highlighted. To test the agriculture use case mentioned in section 3.3.1, experiments were conducted to identify fungus regions in zoysia grass at different elevation using ZA and WM.

<b>Zoom Augment</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.629594278	0.479933
<b>2</b>	0.626482495	0.429818
<b>3</b>	0.615708043	0.404324
<b>4</b>	0.585047077	0.321353
<b>5</b>	0.590333073	0.232538
<b>AVERAGE</b>	<b>0.609432993</b>	<b>0.373593</b>

Table 5-10: Network was trained on the highest spatial resolution for the grass dataset using ZA.

<b>WM</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.629594278	0.577266
<b>2</b>	0.626482495	0.57497
<b>3</b>	0.615708043	0.563166
<b>4</b>	0.585047077	0.543248
<b>5</b>	0.590333073	0.482413
<b>AVERAGE</b>	<b>0.609432993</b>	<b>0.548213</b>

Table 5-11: Network was trained on the highest spatial resolution for the grass dataset using WM.

Contrasting the results from Table 5-10 and Table 5-11, WM improves the IOU score for the coarsest scale from 0.37 to 0.54. The IOU scores are low in comparison to the previous datasets; this is because dry patches do not have a defined shape – edges.



This makes it difficult to create training labels as the regions are not clearly defined, i.e., edges of dry patches are mixed with healthy grass. During inferencing, this results in edges being ignored as the network will segment the central region of the dry patches. This, therefore, results in a lower IOU score.

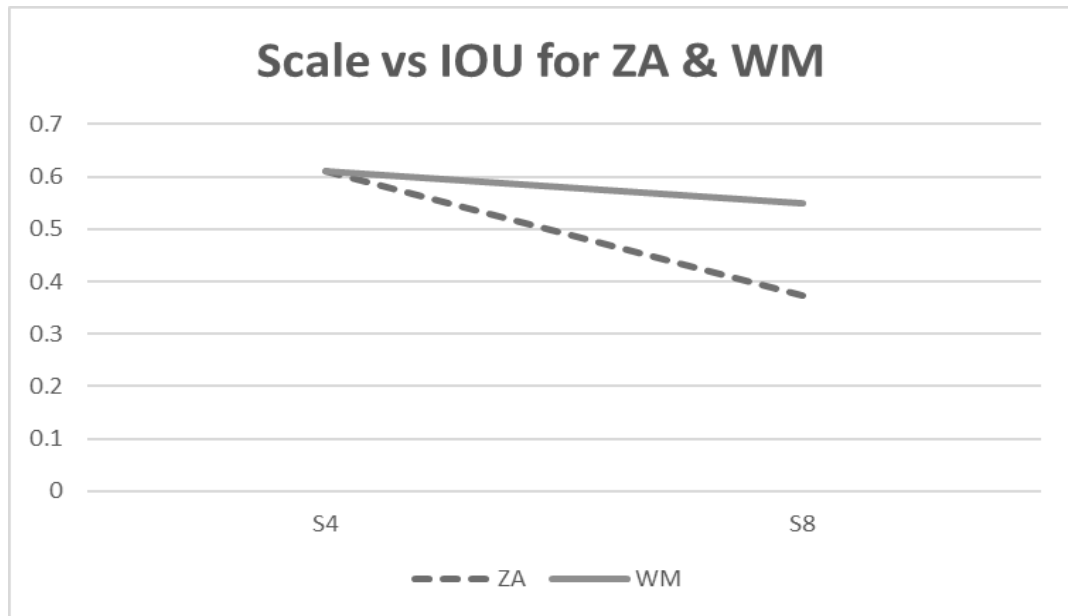


Figure 5-7: Comparison of the IOU across all scales (4, 8) between ZA and WM.

In Figure 5-7, WM and ZA produce the same IOU at scale 4; however, there is a significant difference between scale 8. Histogram equalization did not provide an improvement in IOU for this dataset, see Appendix 8.5.

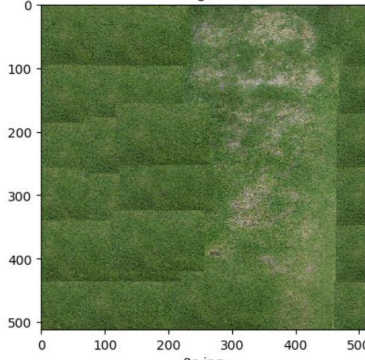
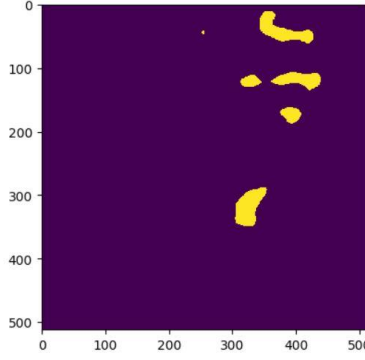
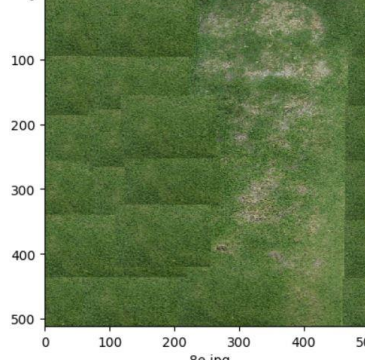
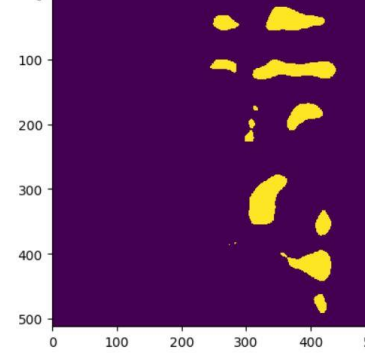
Method	Results	
ZA	 <p>Original</p> <p>8e.jpg</p>	 <p>Predicted Mask</p>
WM	 <p>Original</p> <p>8e.jpg</p>	 <p>Predicted Mask</p>

Figure 5-8: Predictions from using ZA versus WM.

It is evident that WM produces more accurate predictions than ZA. From Figure 5-8, using WM, the network can identify dry grass regions which are not detectable when using ZA. The experiments conducted in this chapter proves that WM is the superior scaling method to the naïve approach of ZA, more results on the different datasets and techniques used can be seen in Appendix 8.5.

## 5.4 Summary of Chapter 5

<b>Pond Dataset</b>			
	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>Original</b>	0.800775949767	0.001394335512	0
<b>ZA</b>	0.7616238	0.294567	0.036986
<b>WM</b>	0.76148428	0.456624	0.346812
<b>WM wt HE</b>	0	0	0
<b>Door Dataset</b>			
<b>Original</b>	0.709833356	0.563161601	-
<b>ZA</b>	0.762269	0.57565	-
<b>WM</b>	0.812863	0.456935	-
<b>WM wt HE</b>	0.916656	0.775292	-
<b>Grass Dataset</b>			
<b>Original</b>	0.501264691	0.233507182	-
<b>ZA</b>	0.609432993	0.373593	-
<b>WM</b>	0.609432993	0.548213	-
<b>WM wt HE</b>	0.556790449	0.54668215	-

Table 5-12: Summary table showing average IOU score for each method across all datasets.

The results from Chapter 5 consisted of a quantitative comparison between the Zoom Augmentation (ZA) and Waterfall Method (WM). From Table 5-12, an improvement is seen with the Pond and Grass dataset using WM; however, there is no improvement for the door dataset. The reduction in IOU score for door using WM is due to misclassified pixels being propagated along the pipeline.

Several image processing techniques such as: Sobel filtering and contrast enhancement using histogram equalization (HE) were used before applying WM. Histogram equalization provided a significant improvement in the door dataset by

reducing the number of misclassified pixels. However, no improvement was seen for the Pond and Grass dataset using HE as highlighted in Table 5-12. The experiments conducted in this chapter show that image processing techniques such as image scaling (ZA or WM) and histogram equalization are essential for improving prediction accuracy.

# Chapter 6

## Conclusion

### 6.1 Summary and Conclusion

The increased application of segmentation requires more robust Machine Learning (ML) algorithms that can handle variations of the input. Areas such as autonomous driving and robotics depend on consistent and reliable results from ML algorithms for them to perform optimally. Convolutional Neural Networks are the current state of the art in image recognition, they have excelled in the areas of classification, object detection, and segmentation.

However, objects at varying sizes pose a significant problem to convolutional neural networks; this is referred to as the scale issue. Researchers (van Nord & Postma, 2017) and (Boominathan, et al., 2016) mention that using images of varying scales or resolutions can make the network more robust to changes in scale. However, although image scaling methods can be used to circumvent this scale issue, it is not practical if the distance between the highest and lowest scale is significant as too much contextual information will be lost in this process.

This research has shown that incrementally stepping across scales using a waterfall approach provides more accurate predictions at lower scales. Despite a few issues, WM is the superior method in comparison to ZA for producing scale-invariant convolutional neural networks. Scale-invariant CNNs can be beneficial in areas such as segmenting agricultural images, automated search and rescue missions, and an improved drone-based delivery system.

The results from section 8.5 of the appendix show that WM provides an improved IOU score for the pond and grass dataset, but not the for the door dataset. Although, WM did not improve the IOU score for the door dataset, combining WM with image processing algorithms such as histogram equalization generated better results. Image processing techniques such as: Waterfall Method, Zoom Augmentation and Histogram Equalization are all essential for providing more accurate segmentation using CNNs.

The contributions of this research include the proposed Waterfall Method (WM), as well as the benefits of using image processing techniques such as Contrast Enhancement using Histogram Equalization to improve prediction accuracy.

## **6.2 Future Directions**

Several optimizers were tested prior to the experiments, see appendix 8.6. Adam was selected as the optimizer of choice. However, the results from the appendix indicate that RMSPROP may provide an improvement in the IOU score. Further testing of optimizers will be done.

Although WM produces more accurate predictions, it is not without fault. False positives in training images causes noisy or high variance training which sometimes result in failures during training or inaccurate predictions. Future work will be done into techniques to reduce the effects of false positives on WM.

Testing of the proposed methods Zoom Augmentation, Waterfall Method and Histogram Equalization on different CNN architectures for segmentation. The experiments for this research was performed using the U-Net architecture, see section 2.7.

Finally, a more accurate method for selecting a scaling factor will be researched. Scaling factors are approximations in the change of spatial resolution; therefore there is no perfect scaling factor. However, the more accurate the approximation, the more accurate the predictions will be. Drone images of various land cover were captured at the end of this research. Therefore, they were not included as part of the experiments. Future experiments and testing will be performed on these images.

# References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016, November). Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).
- Amara, J., Bouaziz, B., & Algergawy, A. (2017). A Deep Learning-based Approach for Banana Leaf Diseases Classification.
- Bhadane, G., Sharma, S., & Nerkar, V. B. (2013). Early pest identification in agricultural crops using image processing techniques. *International Journal of Electrical, Electronics and Computer Engineering*, 2(2), 77-82.
- Bora, D. J., & Gupta, A. K. (2016). A new efficient color image segmentation approach based on combination of histogram equalization with watershed algorithm. *International Journal of Computer Sciences and Engineering*, 4(6), 156-167.
- Chollet, F. (2015). Keras.
- Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul), 2121-2159.
- Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., & Garcia-Rodriguez, J. (2017). A Review on Deep Learning Techniques Applied to Semantic Segmentation. arXiv preprint arXiv:1704.06857.
- Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).



- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning (Vol. 1). Cambridge: MIT press.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
- Jones, H. G., & Sirault, X. R. (2014). Scaling of thermal images at different spatial resolution: the mixed pixel problem. *Agronomy*, 4(3), 380-396.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.
- Lemley, J., Bazrafkan, S., & Corcoran, P. (2017). Deep Learning for Consumer Devices and Services: Pushing the limits for machine learning, artificial intelligence, and computer vision. *IEEE Consumer Electronics Magazine*, 6(2), 48-56.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.

- Lo, S. L., Chiong, R., & Cornforth, D. (2015). Using support vector machine ensembles for target audience classification on Twitter. *PloS one*, 10(4), e0122855.
- Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).
- Maggiori, E., Tarabalka, Y., Charpiat, G., & Alliez, P. (2016, July). Fully convolutional neural networks for remote sensing image classification. In *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International* (pp. 5071-5074). IEEE.
- Mitchell, T. M., & Learning, M. (1997). *Mcgraw-hill science. Engineering/Math*, 1, 27.
- Mohammad, M. A. H. E. (2015). *High Performance Hyperspectral Image Classification using Graphics Processing Units* (Doctoral dissertation, Ain Shams University).
- Najibi, M., Rastegari, M., & Davis, L. S. (2016). G-cnn: an iterative grid based object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2369-2377).
- National Resources Canada. (2015, November 20). Spatial Resolution, Pixel Size, and Scale. Retrieved October 15, 2017, from <http://www.nrcan.gc.ca/node/9407>
- Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1520-1528).
- O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.
- Pratheba, R., Sivasangari, A., & Saraswady, D. (2014, March). Performance analysis of pest detection for agricultural field using clustering techniques. In *Circuit, Power and*

- Computing Technologies (ICCPCT), 2014 International Conference on (pp. 1426-1431). IEEE.
- Rahman, M. A., & Wang, Y. (2016, December). Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation. In International Symposium on Visual Computing (pp. 234-244). Springer International Publishing.
- Raj, A., Maturana, D., & Scherer, S. (2015). Multi-scale convolutional architecture for semantic segmentation. Technical Report CMU-RITR-15-21, Pittsburgh, PA.
- Rajan, P., Radhakrishnan, B., & Suresh, L. P. (2016, October). Detection and classification of pests from crop images using Support Vector Machine. In Emerging Technological Trends (ICETT), International Conference on (pp. 1-6). IEEE.
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).
- Rhineland, J. (2016, December). Feature extraction and target classification of side-scan sonar images. In Computational Intelligence (SSCI), 2016 IEEE Symposium Series on (pp. 1-6). IEEE.
- Richards, J. A (1999). Remote sensing digital image analysis (Vol. 3). Berlin et al.: Springer.
- Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

- Sallee, J., & Meier, L. R. (2010). Kite aerial photography as a tool for remote sensing. *Journal of Extension*, 48(3), n3.
- Sakthivel, K., Nallusamy, R., & Kavitha, C. (2015). Color Image Segmentation Using SVM Pixel Classification Image. *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 8(10), 1919-1925.
- Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Sola, J., & Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 44(3), 1464-1468.
- Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.
- Triantafyllidou, D., & Tefas, A. (2016, December). Face detection based on deep convolutional neural networks exploiting incremental facial part learning. In *Pattern Recognition (ICPR), 2016 23rd International Conference on* (pp. 3560-3565). IEEE.
- Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., ... & Yu, T. (2014). scikit-image: image processing in Python. *PeerJ*, 2, e453.
- van Noord, N., & Postma, E. (2017). Learning scale-variant and scale-invariant features for deep image classification. *Pattern Recognition*, 61, 583-592.

- Wang, R., & Xu, Z. (2015, August). A pedestrian and vehicle rapid identification model based on convolutional neural network. In Proceedings of the 7th International Conference on Internet Multimedia Computing and Service (p. 32). ACM.
- Zeiler, M. D. (2012). ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.
- Zhong, Z., Jin, L., & Xie, Z. (2015, August). High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In Document Analysis and Recognition (ICDAR), 2015 13th International Conference on (pp. 846-850). IEEE.

# Appendix

## 8.1 Appendix A – Experiment Images









Images				
Region of Interest				
	Scale 1	Scale 2	Scale 3	Scale 4

Figure 8-1: Training data for all scales from the pond dataset.




Images			
Region of Interest			
	Scale 1	Scale 2	Scale 3

Figure 8-2: Training data for all scales from the door dataset.




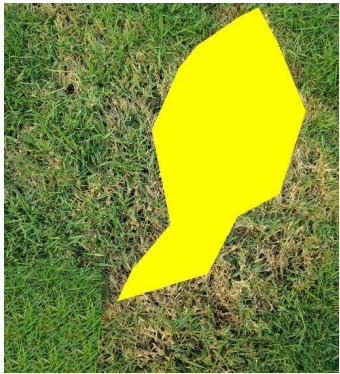

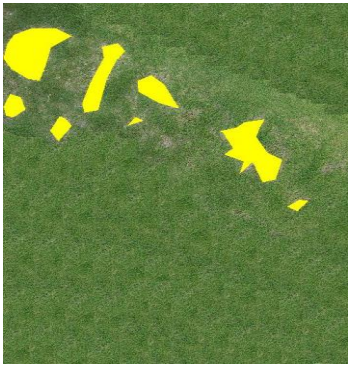
Images			
Region of Interest			
	Scale 1	Scale 4	Scale 8

Figure 8-3: Training data for all scales from the grass dataset.



## 8.2 Appendix B - Future work with drone images




Scale 1	
Scale 2	
Scale 3	

Figure 8-4: Images from the first drone footage.




Scale 1	
Scale 2	
Scale 3	

Figure 8-5: Images from the second drone footage.

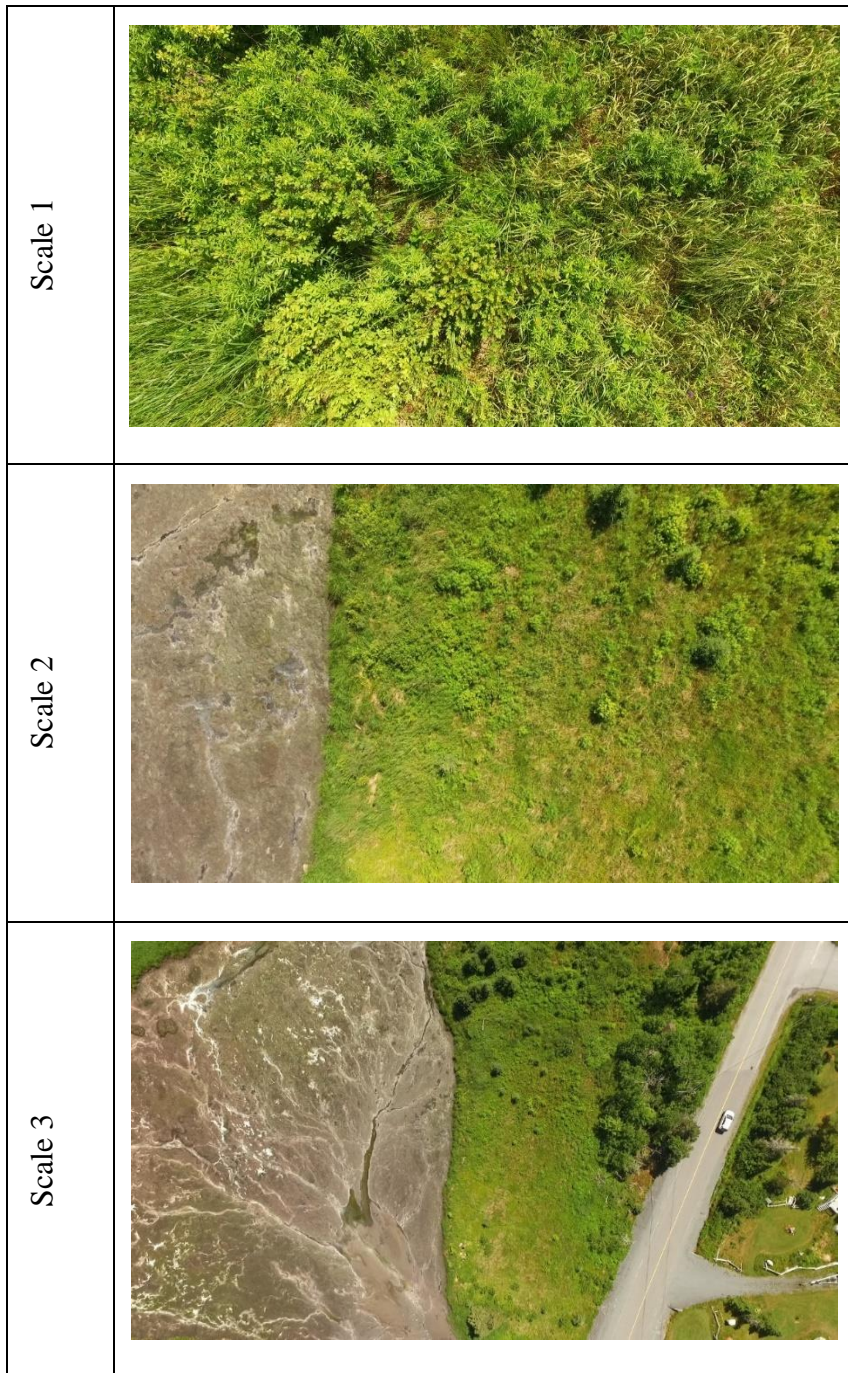
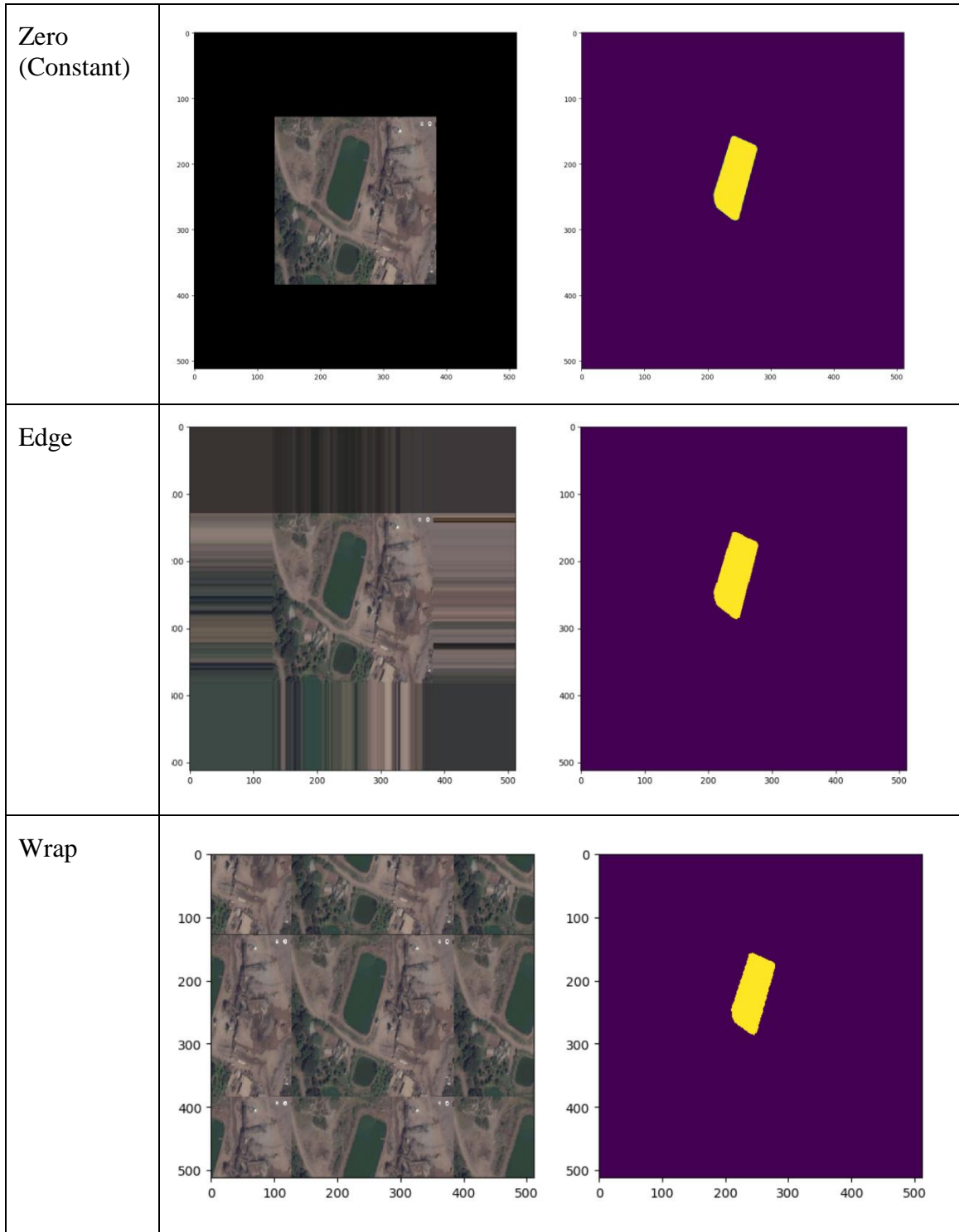


Figure 8-6: Images from the third drone footage.

### 8.3 Appendix C – Padding Methods



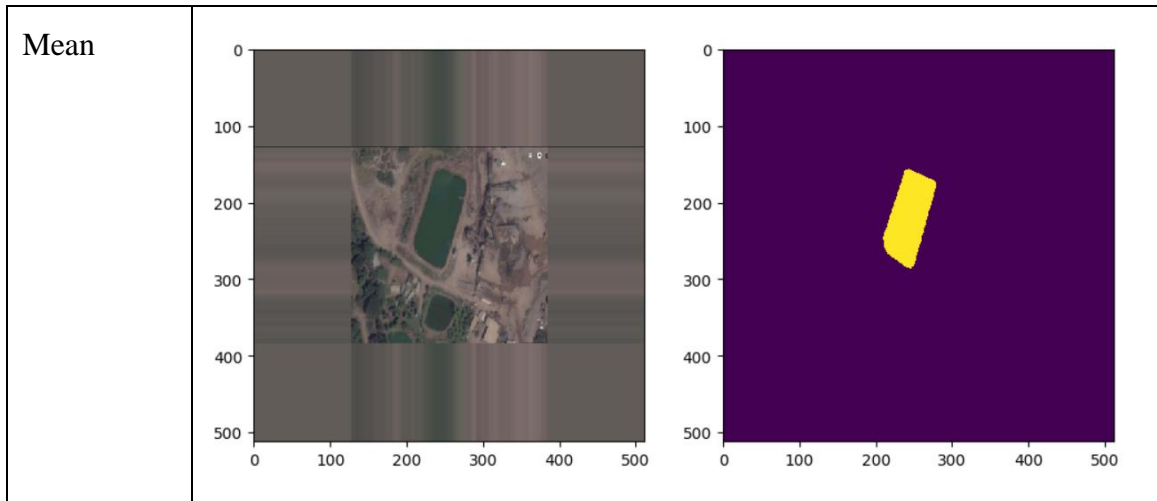


Figure 8-7: Shows four padding methods used after ZA.

Padding Methods	Pond Dataset			
	Zero	Edge	Wrap	Mean
1	0	0.618731	0.370771	0.804365
2	0	0.627839	0.533084	0.700548
3	0.021222	0.500443	0.346805	0.805383
4	0	0.633169	0.571436	0.640916
5	0	0.706453	0.204927	0.801603
6	0	0.497565	0.456299	0.710036
7	0	0.53384	0.576473	0.794956
8	0	0.021222	0.573096	0.735756
9	0	0.65653	0.495529	0.818502
10	0	0.560507	0.556784	0.788429
<b>AVERAGE</b>	<b>0.021222</b>	<b>0.53563</b>	<b>0.46852</b>	<b>0.760049</b>

Table 8-1: IOU scores from four padding methods used after ZA

The image at scale 1 for the pond dataset was zoom augmented to scale 2, and several padding methods were tested. Table 8-1 shows that mean padding is the most consistent and produces accurate IOU scores.

## 8.4 Appendix D – Overview of Training Procedure

Training Parameters	
Image Height	512
Image Width	512
Training size before augmentation	3-4
Training size after augmentation	8-12
Validation size	2-4
Test size	4
Batch size	2
Learning Rate	1e-4
Epochs	500

Table 8-2: Shows the training parameters used for experiments.

Training Size	Door Dataset	
Run	4 Original + 4 Rotation Augmented	10 Original
1	0.870978	0.879754
2	0.884398	0.865318
3	0.886643	0.893627
4	0.866125	0.864448
5	0.876918	0.886131
<b>AVERAGE</b>	<b>0.877012</b>	<b>0.877856</b>

Table 8-3: Results from increasing training size using data augmentation.

From Table 8-3, a CNN was trained on ten original images of the door data from scale 1, while another CNN was trained on four original images combined with four rotation augmented image. The results show that similar IOU scores can be attained using a smaller dataset combined with data augmentation.

## 8.5 Appendix E – Image Processing Results

### 8.5.1 Pond Dataset

<b>Original</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0.810359852652	0.006971678	0
<b>2</b>	0.839199884210	0	0
<b>3</b>	0.758837727831	0	0
<b>4</b>	0.848522024814	0	0
<b>5</b>	0.746960259329	0	0
<b>AVERAGE</b>	<b>0.800775949767</b>	<b>0.001394335512</b>	<b>0</b>

Table 8-4: Results from using no data augmentation on the Pond Dataset.

<b>Zoom Augment</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0.671888	0.298976958381	0.0
<b>2</b>	0.795376	0.256107045301	0.0
<b>3</b>	0.77811	0.372314150521	0.0111827351035
<b>4</b>	0.791835	0.372607861443	0.128241664819
<b>5</b>	0.77091	0.172828927757	0.0455051160734
<b>AVERAGE</b>	<b>0.7616238</b>	<b>0.294567</b>	<b>0.036986</b>

Table 8-5: Results from using ZA on the Pond Dataset.

<b>WM</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0.806615097199	0.436562838212	0.389409579147
<b>2</b>	0.570166713548	0.502918415152	0.261629749784
<b>3</b>	0.802865072335	0.527063030362	0.332860934666
<b>4</b>	0.803789139207	0.497134642537	0.439680070691
<b>5</b>	0.823985379601	0.319441976938	0.310481786915
<b>AVERAGE</b>	<b>0.76148428</b>	<b>0.456624</b>	<b>0.346812</b>

Table 8-6: Results from using WM on the Pond Dataset.

<b>WM wt HE</b>	<b>Pond Dataset</b>		
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>	<b>Scale 4</b>
<b>1</b>	0	0	0
<b>2</b>	0	0	0
<b>3</b>	0	0	0
<b>4</b>	0	0	0
<b>5</b>	0	0	0
<b>AVERAGE</b>	<b>0</b>	<b>0</b>	<b>0</b>

Table 8-7: Results from using WM with HE on the Pond Dataset.



## 8.5.2 Door Dataset

<b>Original</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.761571	0.579931
<b>2</b>	0.682769	0.441041
<b>3</b>	0.734601	0.645998
<b>4</b>	0.655317	0.564791
<b>5</b>	0.714909	0.584047
<b>AVERAGE</b>	<b>0.709833356</b>	<b>0.563161601</b>

Table 8-8: Results from using no data augmentation on the Door Dataset.

<b>Zoom Augment</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.755377986192	0.523978725137
<b>2</b>	0.785577779653	0.610888620163
<b>3</b>	0.716548674317	0.533183327087
<b>4</b>	0.768932974832	0.612936494778
<b>5</b>	0.784905106288	0.597264107258
<b>AVERAGE</b>	<b>0.762269</b>	<b>0.57565</b>

Table 8-9: Results from using ZA on the Door Dataset.

<b>WM</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.753696785647	0.535421992341
<b>2</b>	0.811700857853	0.418043224287
<b>3</b>	0.832152141446	0.378368088879
<b>4</b>	0.810602905292	0.441032139817
<b>5</b>	0.856163909258	0.511807573824
<b>AVERAGE</b>	<b>0.812863</b>	<b>0.456935</b>

Table 8-10: Results from using WM on the Door Dataset.

<b>WM wt HE</b>	<b>Door Dataset</b>	
<b>Run</b>	<b>Scale 2</b>	<b>Scale 3</b>
<b>1</b>	0.902338099	0.762798307992
<b>2</b>	0.902661542	0.799544890362
<b>3</b>	0.921064694	0.817921326584
<b>4</b>	0.93043777	0.726590352478
<b>5</b>	0.92677762	0.769605002381
<b>AVERAGE</b>	<b>0.916656</b>	<b>0.775292</b>

Table 8-11: Results from using WM with HE on the Door Dataset.

### 8.5.3 Grass Dataset

<b>Original</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.535388019	0.33788581
<b>2</b>	0.532847447	0.279958012
<b>3</b>	0.511806852	0.209943247
<b>4</b>	0.470944908	0.208243042
<b>5</b>	0.455336227	0.131505798
<b>AVERAGE</b>	<b>0.501264691</b>	<b>0.233507182</b>

Table 8-12: Results from using no data augmentation on the Grass Dataset.

<b>Zoom Augment</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.629594278	0.479933
<b>2</b>	0.626482495	0.429818
<b>3</b>	0.615708043	0.404324
<b>4</b>	0.585047077	0.321353
<b>5</b>	0.590333073	0.232538
<b>AVERAGE</b>	<b>0.609432993</b>	<b>0.373593</b>

Table 8-13: Results from using ZA on the Grass Dataset.

<b>WM</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.629594278	0.577266
<b>2</b>	0.626482495	0.57497
<b>3</b>	0.615708043	0.563166
<b>4</b>	0.585047077	0.543248
<b>5</b>	0.590333073	0.482413
<b>AVERAGE</b>	<b>0.609432993</b>	<b>0.548213</b>

Table 8-14: Results from using WM on the Grass Dataset.

<b>WM wt HE</b>	<b>Grass Dataset</b>	
<b>Run</b>	<b>Scale 4</b>	<b>Scale 8</b>
<b>1</b>	0.578595947	0.582058581
<b>2</b>	0.545683067	0.558873205
<b>3</b>	0.565429621	0.548106107
<b>4</b>	0.555230695	0.53414636
<b>5</b>	0.539012912	0.510226498
<b>AVERAGE</b>	<b>0.556790449</b>	<b>0.54668215</b>

Table 8-15: Results from using WM with HE on the Grass Dataset.

The results from this section of the appendix show the IOU score from several methods mentioned in this research across all datasets. Table 8-7 shows that combining several augmentation methods for a given dataset produces erroneous results. This was not observed for the other datasets.

## 8.6 Appendix F – Optimizer Results

<b>Door Dataset</b>					
<b>Run</b>	<b>ADADELTA</b>	<b>ADAGRAD</b>	<b>ADAMAX</b>	<b>ADAM</b>	<b>RMSPROP</b>
<b>1</b>	0.872221	0.741377	0.916406	0.912428	0.90055
<b>2</b>	0.87326	0.760499	0.906377	0.933782	0.867157
<b>3</b>	0	0.727241	0.882737	0.923811	0.783758
<b>4</b>	0.76478	0.777296	0.906693	0.892835	0.718127
<b>5</b>	0.927521	0.786313	0.878107	0.897167	0.876897
<b>6</b>	0.933112	0.801827	0.913926	0.933088	0.860732
<b>7</b>	0.02341	0.860883	0.933602	0.926129	0.937059
<b>8</b>	0.927267	0.780936	0.904363	0.903392	0.700712
<b>9</b>	0	0.748698	0.019588	0	0.831588
<b>10</b>	0.897261	0.829661	0	0.922527	0.930679
<b>AVERAGE</b>	<b>0.621883</b>	<b>0.781473</b>	<b>0.72618</b>	<b>0.824516</b>	<b>0.840726</b>

Table 8-16: Results from scale 2 of the Door Dataset using several optimizers.

## 8.7 Appendix G – Training time Results

<b>Pond Dataset</b>		
<b>Run</b>	<b>Elapsed Time</b>	<b>Final Epoch</b>
<b>1</b>	4m:7s	80
<b>2</b>	3m:54s	75
<b>3</b>	3m:42s	72
<b>4</b>	4m:6s	79
<b>5</b>	3m:47s	73

Table 8-17: Running Time and Final epoch from the Pond Dataset.

<b>Door Dataset</b>		
<b>Run</b>	<b>Elapsed Time</b>	<b>Final Epoch</b>
<b>1</b>	4m:14s	130
<b>2</b>	3m:56s	118
<b>3</b>	3m:38s	106
<b>4</b>	3m:39s	109
<b>5</b>	4m:11s	124

Table 8-18: Running Time and Final epoch from the Door Dataset.

<b>Grass Dataset</b>		
<b>Run</b>	<b>Elapsed Time</b>	<b>Final Epoch</b>
<b>1</b>	3m:32s	185
<b>2</b>	2m:3s	105
<b>3</b>	1m:46s	92
<b>4</b>	2m:31s	132
<b>5</b>	2m:13s	116

Table 8-19: Running Time and Final epoch from the Grass Dataset.

## 8.8 Appendix H – Confidence Level Results for Segmentation

Pond Dataset					
Run	0.99	0.95	0.9	0.8	0.5
1	0.83482	0.83164	0.83108	0.83145	0.83037
2	0.86753	0.86987	0.86987	0.86893	0.86734
3	0.81153	0.80208	0.79633	0.79100	0.78137
4	0.84971	0.85268	0.85237	0.85131	0.84998
5	0.80735	0.78979	0.78375	0.77794	0.76875
<b>AVERAGE</b>	<b>0.83419</b>	<b>0.82921</b>	<b>0.82668</b>	<b>0.82412</b>	<b>0.81956</b>

Table 8-20: Results from different Confidence levels on the Pond Dataset.

Door Dataset					
Run	0.99	0.95	0.9	0.8	0.5
1	0.79022	0.77600	0.76748	0.75492	0.73315
2	0.68540	0.65534	0.64740	0.63971	0.62931
3	0.68929	0.68385	0.68097	0.67799	0.67200
4	0.64823	0.63349	0.62734	0.62046	0.60778
5	0.79389	0.76578	0.75815	0.74987	0.73797
<b>AVERAGE</b>	<b>0.72141</b>	<b>0.70289</b>	<b>0.69627</b>	<b>0.68859</b>	<b>0.67604</b>

Table 8-21: Results from different Confidence levels on the Door Dataset.

<b>Grass Dataset</b>					
<b>Run</b>	<b>0.99</b>	<b>0.95</b>	<b>0.9</b>	<b>0.8</b>	<b>0.5</b>
<b>1</b>	0.38349	0.39365	0.39675	0.40027	0.40577
<b>2</b>	0.45765	0.46330	0.46582	0.46858	0.47250
<b>3</b>	0.36099	0.36882	0.37161	0.37525	0.38063
<b>4</b>	0.36163	0.36767	0.37026	0.37252	0.37674
<b>5</b>	0.34373	0.36623	0.37490	0.38288	0.39517
<b>AVERAGE</b>	<b>0.38150</b>	<b>0.39193</b>	<b>0.39587</b>	<b>0.39990</b>	<b>0.40616</b>

Table 8-22: Results from different Confidence levels on the Grass Dataset.

The results from these experiments indicate that there is a marginal difference in IOU score when using a different level of confidence. An image segmented using a confidence level of 0.99 and 0.5 can produce similar IOU scores.