Summer 2021

Deep Learning Approaches for Seagrass Detection in Multispectral Imagery

Kazi Aminul Islam
Old Dominion University, russelkazi102@gmail.com

DEEP LEARNING APPROACHES FOR SEAGRASS DETECTION IN MULTISPECTRAL IMAGERY

by

Kazi Aminul Islam
B.S. April 2010, Khulna University of Engineering and Technology
M.S. May 2016, Lamar University

A Dissertation Proposal Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

ELECTRICAL AND COMPUTER ENGINEERING

OLD DOMINION UNIVERSITY
August 2021

Approved by:

Jiang Li (Director)

Richard Zimmerman (Member)

Dimitrie Popescu (Member)

Chung Hao Chen (Member)

ABSTRACT

DEEP LEARNING APPROACHES FOR SEAGRASS DETECTION IN MULTISPECTRAL IMAGERY

Kazi Aminul Islam
Old Dominion University, 2021
Director: Dr. Jiang Li

Seagrass forms the basis for critically important marine ecosystems. Seagrass is an important factor to balance marine ecological systems, and it is of great interest to monitor its distribution in different parts of the world. Remote sensing imagery is considered as an effective data modality based on which seagrass monitoring and quantification can be performed remotely. Traditionally, researchers utilized multispectral satellite images to map seagrass manually. Automatic machine learning techniques, especially deep learning algorithms, recently achieved state-of-the-art performances in many computer vision applications. This dissertation presents a set of deep learning models for seagrass detection in multispectral satellite images. It also introduces novel domain adaptation approaches to adapt the models for new locations and for temporal image series. In Chapter 3, I compare a deep capsule network (DCN) with a deep convolutional neural network (DCNN) for seagrass detection in high-resolution multispectral satellite images. These methods are tested on three satellite images in Florida coastal areas and obtain comparable performances. In addition, I also propose a few-shot deep learning strategy to transfer knowledge learned by DCN from one location to the others for seagrass detection. In Chapter 4, I develop a semi-supervised domain adaptation method to generalize a trained DCNN model to multiple locations for seagrass detection. First, the model utilizes a generative adversarial network (GAN) to align marginal distribution of data in the source domain to that in the target domain using unlabeled data from both domains. Second, it uses a few labeled samples from the target domain to align class-specific data distributions between the two. The model achieves the best results in 28 out of 36 scenarios as compared to other state-of-the-art domain adaptation methods. In Chapter 5, I develop a semantic segmentation method for seagrass detection in multispectral time-series images. First, I train a state-of-the-art image segmentation method using an active learning approach where I use the DCNN classifier in the loop. Then, I develop an unsupervised domain adaptation (UDA) algorithm

to detect seagrass across temporal images. I also extend our unsupervised domain adaptation work for seagrass detection across locations. In Chapter 6, I present an automated bathymetry estimation model based on multispectral satellite images. Bathymetry refers to the depth of the ocean floor and contributes a predominant role in identifying marine species in seawater. Accurate bathymetry information of coastal areas will facilitate seagrass detection by reducing false positives because seagrass usually do not grow beyond a certain depth. However, bathymetry information of most parts of the world is obsolete or missing. Traditional bathymetry measurement systems require extensive labor efforts. I utilize an ensemble machine learning-based approach to estimate bathymetry based on a few in-situ sonar measurements and evaluate the proposed model in three coastal locations in Florida.

Dedicated to my wife, son, parents, and all my mentors for their continuous support.

ACKNOWLEDGEMENTS

I would like to offer my sincerest appreciation for the direction, assistance and guidance offered by Dr. Jiang Li, who is my advisor and whose support and patience have been invaluable. I would also like to sincerely thank my committee members, Dr. Richard Zimmerman, Dr. Dimitrie Popescu, Dr. Chung Hao Chen, and express my deepest gratitude for their valuable guidance. I would like to thank my fellow lab-mates. I would also like to acknowledge the support from my wife, my son, my parents, my in-laws and other family members, without which the dissertation would not have been possible. I also gratefully acknowledge the support of NVIDIA Corporation for the donation of the TESLA K40 GPU used in this research.

TABLE OF CONTENTS

Page

Chapter Page

Chapter Page

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

1.1 Background

Seagrasses create critically important marine ecosystems by reducing atmospheric carbon dioxide and mitigating climate change. Seagrasses also provide food and shelter for fish and marine organisms, protect ecological systems, stabilize sea bottoms, maintain water quality, and help the local economy [2–6]. Seagrasses can be found in coastal areas all over the world [7]. Seagrasses live in sediments, intertidal and subtidal areas of coastal shallow regions by forming a meadow [8]. There is a need to accurately identify seagrass due to its salient role in keeping the ecosystem intact. This will help researchers better understand the growth of seagrass all over the world [6, 9].

Coastal areas have been significantly impacted over the last several decades by activities (aquaculture, humans with propeller current) of nearby inhabitants and coastal visitors. Due to the growth of the human population and industrial evolution, the release of waste and polluted water have also increased significantly in coastal areas [2–5, 10]. These issues are causing the deterioration of water quality and a decrease in seagrass distribution. Seagrass distribution is also damaged by natural calamities such as typhoons, strong wind, rainfall, climate warming, sea level rise, and ocean acidification [2–5, 10]. Florida has lost 50% of its seagrass between the 1880s and the 1950s [4]. Therefore, improving water quality

to restore seagrass has been a priority during the last few decades. Tampa Bay, Florida had an increase of seagrass distribution between the 1980s to 2006 by 114.5 km^2 [4].

Although seagrass monitoring is an important topic for marine researchers, the preferable approach for seagrass detection is still a manual approach using in-situ measurement data. With the advancement of technology, researchers came up with various approaches for seagrass detection. Among them, remote sensing imagery is a cost-effective and efficient system for seagrass detection [4,5]. Besides multi-spectral satellite images, researchers utilized digital video and images, Lidar, sonar data to detect seagrass [8]. In most of the previous assessments of seagrass distributions in the remotely sensed imagery, domain experts manually drew the seagrass maps [8]. Recently, researchers utilized machine learning algorithms with multi-spectral images for seagrass detection in different parts of the world [11,12]. Deep learning is a sub-group of machine learning algorithms which achieved state-of-the-art performances in image classification [13], image segmentation [14–16], and object detection [17,18] domains. In this dissertation, we propose deep learning-based seagrass monitoring models based on multi-spectral satellite images.

To develop a deep learning-based seagrass detection system, we first consider the scenario where we have enough labeled samples to train a deep learning model. Deep learning models usually require a large number of labelled training data to achieve competitive results. For seagrass detection, these labeled data are obtained by in situ observations that are time-consuming and labor-intensive. We compare performances of deep capsule network (DCN) and deep convolutional neural network (DCNN) for seagrass detection with the labeled data in a cross-validation setting. We also compare the two methods with a baseline

support vector machine (SVM) model. We found that DCNN and DCN models achieve comparable performances whereas SVM performs poorly in the cross-validation setting.

A well trained deep learning model at one location may fail at others if seagrass density distribution shifts from the source domain to the target domain. We explore the scenario of adapting a trained model at one location to others using a few labeled samples from a target location as guidance. For seagrass detection, we usually have abundant unlabeled samples, and it is possible to obtain a few labeled samples from the target location. We propose a semi-supervised method that uses both unlabeled and labeled samples to adapt the trained models.

In this dissertation, we also explore the research of classifying seagrass without using any labeled samples from a target location. We divide domain shift into two categories: temporal shift and location shift. Temporal shift refers to change of seagrass distribution due to seasonal variation in the same location while location shift refers to change of seagrass distribution due to change of location. For temporal shift domain adaptation, our overall objective is to train a classifier on the first time-point image with enough labeled samples and then effectively classify all the future temporal images without using any labeled samples. To solve this challenge, we propose to use a pixel-wise semantic segmentation approach, which can learn the relationship among neighboring pixels. To adapt the classifier for future temporal images, we propose to develop an unsupervised domain adaptation approach.

We propose to explore further the unsupervised seagrass detection task across locations without using any labeled samples from target domains. Seagrasses usually look different across locations due to water quality, water depth, suspended sediments [19] and

colored dissolved organic matter (CDOM) [20]. If we do not have any labeled samples from a target location, the trained seagrass detection model may get fooled by the different influence factors. To adapt seagrass detection classifiers, we propose to use the generative adversarial network (GAN) based domain adaptation approach to detect seagrass in target domains.

Bathymetry represents sea depth of the ocean floor which is a popular way to detect underwater marine objects. Seagrasses usually do not grow under a certain depth. Using the bathymetry measurement, we can improve performances of seagrass detection models by removing false positives. Normally, researchers utilize manual approaches, e.g.: sonar (sound navigation ranging) and LIDAR to measure the depth of the ocean floor. In this dissertation, we propose a machine learning-based bathymetry estimation model using multi-spectral images. Our proposed approach uses sonar measurements and corresponding image information to learn the underlying relationship. As we have limited sonar measurements, we propose to use an ensemble machine learning-based regressor for bathymetry estimation.

1.2 Contribution

In this dissertation, we explore the research problem of detecting seagrass using deep learning techniques in multi-spectral images. First, we try to find the best method for detecting seagrass in selected regions using state-of-the-art deep learning models. Then, we work on domain adaptation for seagrass detection. Contributions of this dissertation are summarized below:

1.2.1 Seagrass Detection using Deep Capsule Network

We compare a deep capsule network with deep convolutional neural network to detect seagrass in Florida coastal areas based on multispectral satellite images. To generalize a trained seagrass detection model to new locations, we utilize the capsule network as a data augmentation method to generate new artificial data for fine-tuning the model. The main contributions of this task are:

1. A capsule network is developed for seagrass detection in multispectral satellite images.
2. A few-shot deep learning strategy is implemented for seagrass detection. and it may be applicable to other applications.

We published a conference paper using outcomes from this task [21, 22]:
Islam, Kazi Aminul, Daniel Perez, Victoria Hill, Blake Schaeffer, Richard Zimmerman, and Jiang Li. Seagrass detection in coastal water through deep capsule networks. In Chinese Conference on Pattern Recognition and Computer Vision (PRCV), pp. 320-331. Springer, Cham, 2018.

1.2.2 Semi-supervised Domain Adaptation for Seagrass Detection

For seagrass detection, we usually have large amount of unlabeled data for a given new location and it is possible to obtain limited labeled data from domain experts. We propose a novel domain adaptation approach that uses both unlabeled data and a few labeled samples to learn an effective classifier for new locations. First, we utilize an unsupervised adversarial domain adaptation approach to adapt target domain representation to mimic source domain representation so that the classifier trained in the source domain may work in the target domain. Second, we utilize a supervised approach with the contrastive semantic alignment

loss to learn domain invariant representations between source and target domains. The first step aligns marginal distribution between domains, and the second step aligns class specific distributions using a few labeled samples from the target domain. The proposed domain adaptation approach optimizes the target domain embedding function to create a simple classifier that can work effectively in the target domain. Contributions of our proposed approach are:

- A novel approach which uses both unlabeled and a few labeled samples in the target domain to learn a domain invariant embedding for domain adaptation. It can utilize a large amount of unlabeled data for efficient training.
- To the best of our knowledge, this is the first attempt and successful system that can generalize deep CNN models for seagrass detection from one location to another.

We published one conference paper and one journal paper using the results of this task [23, 24].

- Islam, Kazi Aminul, Victoria Hill, Blake Schaeffer, Richard Zimmerman, and Jiang Li. Semi-supervised Adversarial Domain Adaptation for Seagrass Detection in Multispectral Images. In 2019 IEEE International Conference on Data Mining (ICDM), pp. 1120-1125. IEEE, 2019.
- Islam, Kazi Aminul, Victoria Hill, Blake Schaeffer, Richard Zimmerman, and Jiang Li. Semiâsupervised Adversarial Domain Adaptation for Seagrass Detection Using Multispectral Images in Coastal Areas. Data Science and Engineering(DSE) Springer, 2020

1.2.3 Seagrass Detection using Unsupervised Domain Adaptation

To detect seagrass in an image with temporal-shift or location-shift, we propose to use an unsupervised domain adaptation method. The proposed method will modify the target domain representation in the embedding space using the unlabeled samples with a GAN-based loss so that the source classifier can effectively classify the target domain images. Our contributions are:

- We propose a pixel-wise image segmentation method for seagrass detection where we use an active learning based approach to generate a pixel-wise label for multi-spectral images.
- We propose an unsupervised domain adaptation approach to minimize domain shift in temporal shift or location shift in multi-spectral images, where we adapt the trained source segmentation model using a generative adversarial network (GAN) loss in the target domain images without labeled samples.

1.2.4 Bathymetry Estimation using Machine learning Approaches

We propose an automated machine learning-based bathymetry estimation approach, which uses a multi-spectral satellite image to accurately predict bathymetry information.

Our contributions are:

- We are the first to develop a bathymetry estimation model using a gradient boosting machine learning regressor with multi-spectral satellite images.
- Our proposed bathymetry estimation model effectively measures bathymetry in three coastal locations of Florida using a few sonar measurement data.

1.3 Outline of the Dissertation

This dissertation is arranged in the following chapters:

1. Chapter 1: Introduction to the problem, existing work, and our contributions.
2. Chapter 2: We discuss some related work for seagrass detection and deep learning.
3. Chapter 3: We propose a deep capsule network for seagrass detection.
4. Chapter 4: We propose a semi-supervised domain adaptation approach for seagrass detection.
5. Chapter 5: We propose an unsupervised domain adaptation approach for seagrass detection.
6. Chapter 6: We proposed an ensemble model for bathymetry estimation.
7. Chapter 7: We provide concluding remarks about the research.

CHAPTER 2

RELATED WORK

2.1 Seagrass Distributions Mapping

Many automated systems to map seagrass distribution in multispectral satellite images have been developed. Traganos *et al.* proposed a support vector machine (SVM) approach to map the Mediterranean seagrass distribution in Greece utilizing Sentinel-2 satellite imagery [12, 25]. Lions *et al.* utilized field survey data and multi-spectral image data from the QuickBird satellite with maximum likelihood supervised classification algorithm for seagrass mapping in shallow coastal water [26]. WorldView-2 multispectral images have been used for shallow-water Benthic identification using a maximum likelihood classifier [27]. Pasqualinia *et al.* have found the overall accuracies to be between 73% and 96% for identifying four classes: sand, photophilous algae on rock, patchy seagrass beds, and continuous seagrass beds, with two spatial resolutions of 2.5m and 10m [28]. Vela *et al.* used fused image of SPOT-5 and IKONOS in southern Tunisia near the Libyan border to detect four classes including low seagrass cover, high seagrass cover, superficial mobile sediments, and deep mobile sediments [29]. For the lagoon environment mapping, they have obtained 83.25% accuracy over the entire area, 85.91% accuracy over the testing area with SPOT-5 images, and 73.41% accuracy over the testing area with IKONOS images [29]. Dahdough *et al.* combined red, green and blue of visible bands with near infra-red band for seagrass and

algae detection [30]. Oguslu *et al.* used a sparse coding method for sea-grass's propeller scar detection in WorldView-2 satellite images [31].

Different data sources including Landsat [32], IKONOS [33–35], Quickbird [36] and WorldView-2 satellite image sensors [21, 22, 37–39], and different machine learning models such as decision trees, naive Bayes, SVMs [32], and maximum likelihood [33, 34, 38, 39] have been utilized for effective seagrass distribution mapping. However, no model can be directly applied to new locations successfully without adaptation and extensive local training.

2.2 Bathymetry estimation

2.2.1 Traditional Bathymetry estimation in multi-spectral Images

Recently, researchers widely utilized multi-spectral satellite images for bathymetry estimation. Lafon *et al.* used SPOT satellite imagery and a set of field measurements to develop bathymetry estimation models and found that water reflectance is directly linked to depth [40]. Legleiter *et al.* empirically pointed out high similarity between field measured reflectance values and depth values. They also observed that the optimal band ratio analysis (OBRA) based bathymetry mapping values produce high agreement with real depth values in Worldview-2, Landsat-7 (ETM+), MODIS, and ASTER sensors [41]. Doxani *et al.* used Worldview-2 satellite imagery to compare competence of the Lyzenga linear bathymetry model in dense and sparse seagrass regions [42]. Su *et al.* utilized IKONOS and Landsat satellite images with a geographically adaptive inversion model [43]. The same group also used the non-linear inversion model in 2008 [44]. Wei *et al.* utilized Landsat-8, Sentinel-3 and Suomi National Polar-orbiting Partnership (SNPP) sensor data to estimate bathymetry

in corals, seagrass, and sand shallow water regions using a physics based approach: temporal variation of water-column optical properties [45]. Ma *et al.* utilized new ICESat-2 bathymetric points and Sentinel-2 multispectral imagery to estimate bathymetry using a linear model and a band ratio mode in South China Sea, Acklins Island and Bahama [46].

Pattanaik *et al.* estimated the bathymetry using radiative transfer equations for the Indian ocean using IRS-1C/1D and LISS-III sensor satellite data [47]. Stumpf *et al.* also proposed a ratio transfer algorithm using IKONOS satellite imagery for bathymetry estimation [48]. Ratio transfer algorithm also estimated better bathymetry compared to the Lidar [48]. Pushparaj *et al.* used the same ratio transfer algorithm to measure bathymetry in Landsat-8 images [49]. In addition to multispectral images from CubeSat [50], Landsat-8 [51], QuickBird [52], radar [53], Sentinel-2 [54] and Worldview-2 [38] sensors were utilized for bathymetry measurement. Our proposed model utilizes multi-spectral satellite images to estimate bathymetry in the shallow coastal area.

2.2.2 Optical bathymetry estimation using Lidar

Light detection and ranging (LIDAR) uses a pulsed-laser beam to estimate the depth of shallow coastal water. LIDAR sends a green laser beam that penetrates through the water and hits the water surface. The laser beam then is reflected back through water and finally hits the LIDAR sensor [55,56]. The laser measures the distance once reflected from the water surface [55, 56]. It can be used to estimate the bathymetry for up to $70m$ in clear water although the LIDAR-based method has higher cost and coarser bathymetry estimation [57]. Kinzel *et al.* utilized LIDAR data to estimate bathymetry on the Trinity and Klamath River in California, and the Colorado River in Colorado [58]. Saylam *et al.* quantified Airborne

Lidar Bathymetry (ALB) in the Frio River in Texas [59]. LIDAR data were also used for Bathymetry estimation in Germany [60] and Pielach River in Austria [61].

2.2.3 Bathymetry estimation using machine learning approach

Machine learning methods are effectively utilized in disease diagnosis [62,63], medical imaging [64], cybersecurity [65], and border security [66] applications. Researchers are also using the machine learning-based method to develop bathymetry estimation models. Mishra et al. proposed a support vector machine (SVM) based approach for shallow water bathymetry mapping utilizing remote sensing images. Their proposed method achieved comparable or better results than the traditional methods: linear transform model and ratio transform model [67]. Gayman *et al.* used multi-spectral optical imagery Rapid-Eye to estimate the bathymetry on the Great Bahama Bank [68]. Gayman *et al.* proposed a cluster-based regression (CBR) algorithm which provided better performance than the traditional linear and ratio methods. The CBR based method also performed better than the state-of-the-art machine learning model support vector machine (SVM) [68].

Mohammed *et al.* proposed an ensemble-based method for bathymetry estimation utilizing high-resolution satellite imagery [69]. They also used ensemble regression algorithms bagging (BAG), least squares boosting (LSB), support vector regression algorithm (SVR), the neural network (NN) and the Lyzenga generalised linear model (GLM) [70] in Landsat-8 and Spot-6 satellite sensors. Jalilzadeh *et al.* used artificial neural network (ANN), adaptive neuro fuzzy inference system (ANFIS), and regression learner in Landsat-8 images [71]. Sagawa *et al.* used multi-temporal Landsat-8 satellite images and a random forest model to estimate the satellite-derived bathymetry (SDB) [72]. Tonion *et al.* empirically

determined random forest was performing better than the SVM in the Landsat-8, Sentinel-2 and Planet-scope image sensors [73]. Ai *et al.* used convolutional neural network (CNN) in three satellite sensors: Resources Satellite Three (ZY3), Gaofen-1 (GF1), and Worldview-2 (WV2) [74]. Dickens *et al.* utilized a segmentation based method U-Net [15] architecture in Orbview-3 satellite sensor [75]. Besides the multi-spectral image, deep Learning method– fully convolutional neural network (FCNN), principal component analysis (PCA)- deep neural network (DNN), deep autoencoder (AE), and deep variational autoencoder (VAE) – are used to estimate bathymetry in the Celeris wave model [76] and flow velocity data [77].

2.3 Deep Learning

Deep learning models are a subset of machine learning methods which were inspired to mimic mammals' vision systems. A typical deep learning model consists of multiple layers of feature extraction processing units called "neurons". During training, these neurons learn to extract useful features from data to perform classification or regression. Deep learning has been successfully applied in image classification [13, 78], image segmentation [15, 15], image super-resolution [79–81], hyperspectral images [82], object detection [83], speech recognition [84], audio classification [85], computer-aided medical diagnosis [62, 63], medical imaging [64, 86] and cybersecurity [65, 87–89]. Deep learning models include feature extraction in the optimization loop and achieve state-of-the-art performances in many applications [90]. However, one challenge of deep learning models is that they require a large amount of training data to achieve competitive performances, making adaptation of deep learning models between domains difficult.

2.3.1 Deep Convolutional Neural Network (DCNN)

Among different deep learning classification models, deep CNN is the most popular model, and more details are provided in a comprehensive survey by Alom *et al.* [90]. A deep CNN model scans an input image using a set of trained filters to search for matched patterns contained in the filters. Each layer in the deep CNN model contains a number of trained filters. A layer close to input searches for simple patterns such as edges with different orientations and layers adjacent to output try to match more class-specific patterns to conduct classification. This hierarchy feature extraction mechanism is key to the success of CNN. Popular deep vision CNN models include AlexNet [13], VGG-net [91], Resnet [78], Dense-net [92] and inceptionV3 [93]. Deep CNN models use multiple processing layers to learn new representations for better recognition and achieved state-of-the-art in many applications including image classification [13, 94], medical imaging [64, 86, 95, 96], speech recognition [84], cybersecurity [65,88], biomedical signal processing [97] and remote sensing [66].

2.3.2 Generative Adversarial Network (GAN)

Goodfellow *et al.* developed the Generative Adversarial Network (GAN) model in 2014 [98]. The GAN model contains two networks: generator and discriminator. The GAN network follows the Nash equilibrium theorem where the two networks compete against each other to generate more realistic images. The generator network uses Gaussian noise as the input and generates target images by learning a mapping between them. The discriminator network classifies whether the generated images are real or fake. The discriminator uses the

given target image to learn the relationship between real and fake images whereas the generator utilizes discriminator feedback to improve its performance. Over time both generator and discriminator try to improve their performances. The generator gets good at generating fake images, and the discriminator can better distinguish between real and fake samples. After many iterations, when the discriminator network fails to identify whether the sample is fake or real, the model converges. Then, the generator can be used to produce realistic fake images. Due to its success, GAN has been used in a wide range of applications: super-resolution [99], image to image translation [100], and domain adaptation [101]. Researchers are also improving the GAN model performance through improving the architectures, e.g.: progressive GAN [102] and Wasserstein GAN [103].

2.3.3 Transfer Learning

Transfer learning is a useful strategy to train a machine learning model if there is not enough labeled data to train a model from scratch for a target task. Transfer learning algorithms refer to the research area where we can re-use a source model to solve a new related task. Transfer learning tries to train a predictive model through adaptation by utilizing common knowledge between source and target data domains [104]. Oquab *et al.* have used transfer learning with CNN for small data set visual recognition tasks [105]. Transfer learning has been explored also in computer-aided detection [106], post-traumatic stress disorder diagnosis [63] and face representation [107]. We used a model parameter sharing based transfer learning algorithm for seagrass detection. In the model parameter sharing-based transfer learning algorithm, we first use the source model weight to learn a representation for the target data. Then, we use this representation and corresponding

label to train a classifier for the target domain. Once trained, we use the target classifier to classify the target samples.

2.3.4 Domain Adaptation

Domain adaptation (DA) utilizes a well-trained source domain model to perform tasks in the target domain where we do not have enough labeled data to train a deep learning model. We usually have three types of scenarios for domain adaptation in case of data availability: unsupervised, semi-supervised, and supervised. In the unsupervised domain adaptation, we do not have any labeled samples in the target domain. The target domain has fewer labeled samples and abundant unlabeled samples in the semi-supervised scenario whereas in the supervised domain adaptation scenario, the target domain has labeled samples but not enough to train a deep learning model. With the rise of the deep learning model, different approaches are utilized for domain adaptation.

Recent deep learning based domain adaptation methods can be divided into three categories: divergence, adversarial, and reconstruction based. In divergence based domain adaptation, source and target domains minimize a divergence measure such as maximum mean discrepancy (MMD) [108], CORrelation ALignment (CORAL) [109], or contrastive correlation based distances [110]. Reconstruction based domain adaptation methods use a source domain classifier and modify unlabeled target domain images to look like the source domain using GAN or cycle GAN [100] loss. In this approach, the source classifier can be directly utilized in the target domain to classify the target domain images. In adversarial based domain adaptation, distribution in the target domain is matched to source domain distribution by the GAN loss [98].

Tzeng *et al.* proposed an adversarial discriminate domain adaptation method where they used a GAN loss to perform domain adaptation. The discriminator classifies whether the input samples are coming from the source domain or target domain. The target domain changes its features so that the domain classifier can not correctly identify whether the input samples are coming from the source domain or target domain [101]. The target domain layers work as a generator which changes its representation to look like the source domain. This method is converged when the discriminator gets too confused to correctly identify the domains. At that point, we can use the classifier trained on the source domain to classify the target domain samples. We will explore the adversarial and contrastive correlation-based domain adaptation algorithms for seagrass detection.

2.3.5 Image Segmentation

Image segmentation is utilized in a wide range of applications, e.g.: biomedical imaging, scene understanding, video surveillance, and autonomous vehicle driving [111]. Image segmentation is a pixel-based classifier whereas image classification refers to patch-wise classification. A pixel-based classifier assigns a class label to each pixel whereas a patch-wise classifier denotes the center pixel's as the class label. This helps an image segmentation model to learn the relationship between neighboring pixels whereas the image classification algorithm treats neighboring pixels as the same label.

U-Net [15] is a pixel-based classifier which achieved the state-of-the-art performance in 2015 for medical image segmentation. It is a fully convolutional neural network which means it does not need any fully connected layers. This helps the segmentation model to keep the spatial information of the input image. In contrast, image classification model converts

an image into a vector and uses fully connected layers that lose the spatial relationships between pixels. This approach performs well in image classification tasks. However, it is usually difficult to get the original image back after converting it to vector due to loss of spatial information. U-NET consists of encoder and decoder layers. In the encoder layers, U-Net uses multiple convolutional layers to reduce the input dimension with increased number of kernels. This helps preserve the image information. After encoder, it uses a few convolutional layers where it does not change the image dimensions and works as a bridge between encoder and decoder layers. The difference between U-Net as compared to FCN [14] is that U-Net uses feature maps of encoding layers during up-sampling/decoder layers. During encoding, the U-Net model encodes the image into a lower dimension and learns the image features. Then, it uses the same learned features to get back to the original image dimensions. As a result, U-Net learns the in-variance property, can separate the touching objects, and performs detection from edge to edge of the object. In the final layer, U-Net uses pixel-wise soft-max and binary cross-entropy loss to perform the classification. The recently proposed High-Resolution Network (HR-Net) [16] has improved segmentation performances where the model maintains high resolution throughout the network.

2.3.6 Active Learning

Active learning is a useful algorithm to generate enough labeled samples to train a deep learning model. In an active learning algorithm, we utilize the available few labeled samples to train an initial model. Then, the trained model is used to predict labels of the available unlabeled samples. From the predicted samples, the algorithm selects a subset of samples based on priority metrics which can be corrected by human operators. The new

selected labeled samples are used to retrain the initial model. For seagrass detection, it is almost impossible to correctly label each pixel at a location manually. To tackle this issue, we first use the labeled regions from domain experts to train a DCNN classifier. Then, we use the trained classifier to label available unlabeled samples from the same location. After producing the classification maps, we verify the performance from the domain experts. Then, we use the classification map from the DCNN model to train a new seagrass detection model.

2.3.7 Data Augmentation

Deep learning has been popular for obtaining state-of-the-art performances in different domains although it requires a lot of data to train. If the model is not fed with enough labeled data, it memorizes training data, and this scenario is known as over-fitting. To solve the over-fitting problem, we need to feed the deep learning model with more training data. Data augmentation techniques are used generally to produce more training data. Traditional data augmentation techniques include affine transformation, color space augmentation, kernel filters, mixing images, random erasing, and feature space augmentation [112]. Recently, deep learning-based data augmentation techniques such as GAN [98], adversarial training, meta-learning, and style transfer have been very popular for data augmentation [112]. We use a capsule network-based [113] data augmentation technique to increase the number of data for seagrass detection. The capsule network uses a joint optimization of classification/regression layers and auto-encoder layers. We use the auto-encoder layers of the capsule network to perform data augmentation in the target domain.

CHAPTER 3

SEAGRASS DETECTION USING CAPSULE NETWORK

3.1 Motivation

In this chapter, we compare the deep capsule network with the deep convolutional neural network (DCNN) and support vector machine (SVM) [114] to detect seagrass in multi spectral images. The deep convolutional neural network achieved state-of-the-art performance in a large scale image dataset in 2012 [13] whereas SVM is considered as the baseline model for the image classification.

Sabour *et al.* proposed the capsule network for image classification [113] in 2017. It is more robust to affine transformation, and it has been considered a better method than CNN for identifying overlapping digits in MNIST [113]. In 2018, the same group improved the capsule network with matrix capsules, and the expectation maximization algorithm was used for dynamic routing [115]. The improved model achieved state-of-the-art performance on the smallNORB data set [115]. The capsule network has also been used in breast cancer detection [116] and brain tumor type classification [117]. For highly complex data sets such as CIFAR10, the capsule network has not achieved good performances [118].

First, we will compare the performance of these models in three multispectral images of Florida coastal areas. We use a cross validation scheme to compare the performances in the selected regions. We find comparable performance between the deep capsule network and the deep convolutional network whereas support vector machine performance is much worse.

Then, we propose a few shot learning approach to classify the seagrass in new locations. We also propose a novel capsule network based data augmentation approach to improve the performance. Using the data augmentation approach, we find comparable performance between the deep convolutional neural network and the deep capsule network.

3.2 Capsule Network

3.2.1 Formulation

We develop a capsule network for seagrass detection in this study by following the design in [113]. Let us assume that layer i output is u_i, then the prediction vector output for the next capsule j is [113]:

$$u_{j/i} = W_{ij} u_i. \tag{1}$$

The weight matrix W_{ij} is learned through back propagation. The coupling coefficient c_{ij} between layers is learned through the following equation:

$$c_{ij} = \frac{exp(b_{ij})}{\int_k exp(b_{ik})}. \tag{2}$$

Here, b_{ij} represents the probability of connection between layer i and j. All of the connections are initially set to equal probability of zero. Then, with an iterative process some of the connections are magnified more than others. The input vector to next layer j is:

$$s_j = \int_k c_{ij} \bar{u}_{j/i}. \tag{3}$$

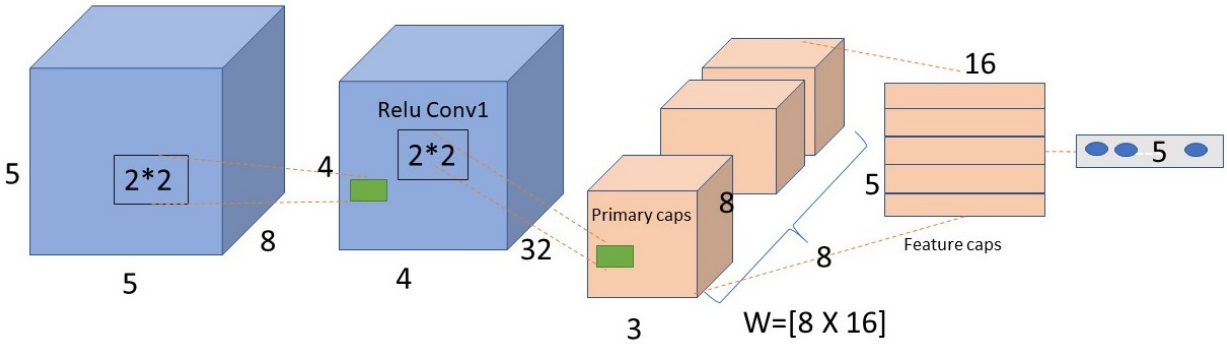
Then a squashing function is used to make the probability between 1 and 0.

$$v_j = \frac{\left\|s_j\right\|^2}{1+\left\|s_j\right\|^2}\frac{s_j}{\left\|s_j\right\|} \tag{4}$$

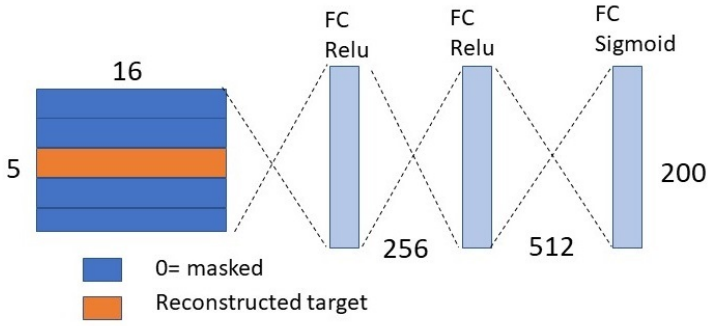
The loss function for the capsule network is as follows:

$$l_k = T_k max(0, m^+ - |v_k|)^2 + \lambda(1 - T_k)max(0, |v_k| - m^-)^2. \tag{5}$$

If a particular class is present, then T_k is 1 and 0 otherwise.



(a) Capsule Network Classification.



(b) Capsule Network Reconstruction.

Fig. 1: Capsule Network.

3.2.2 Architecture

The capsule network's model has two convolutional layers of 2x2 convolutional kernels with 32 filters each for extracting high level features. The features extracted by the convolutional layers are then fed to the capsule layers, in which a weight matrix of 8 by 16 is used to find the corresponding class representations. The last capsule layer, Feature-caps, stores a capsule per class, each capsule having a total of 16 features. The length of each capsule represents the probability for each class. Additionally, the features in each capsule of the feature-caps are used to reconstruct the original image class. The reconstruction architecture has 3 fully connected layers with a linear activation and a size of 256, 512 and 200 hidden units. The output of the decoder corresponds to the size of the input patches (5x5x8).

3.3 Experimental Setup

3.3.1 WorldView-2 Atmospheric Correction

We performed atmospheric correction in the three WorldView-2 satellite multispectral images by matching the images with *in situ* measurements collected at 22 stations across the images on the same day by a survey boat. At each station, the following measurements were obtained by two spectroradiometer systems in tandem:

- $[E_s(0^+)]$: downwelling spectral irradiance above the sea surface (395 to 795 nm, 2.5 nm bandwidth),

- $L_\mu(0.65, \lambda)$: upwelling spectral radiance 0.65 m beneath the sea surface where λ represents wavelength [HTSRB, Satlantic Instr.],

- $E_\mu(0.21)$ and $L\mu(0.21)$: upwelling irradiance and radiance 0.21 m beneath the sea surface [HyperPro, Satlantic Instr.].

With these measurements, we calculated the following attributes:

- Spectral upwelling diffuse attenuation coefficient,

$$K_{L_\mu} = -\frac{1}{z} \ln \frac{L_\mu(0.65)}{L_\mu(0.21)} \tag{6}$$

  where z was the difference in depth between the sensors placed at 0.65 m and 0.21 m.

- Upwelling radiance just beneath the air-water interface $L_\mu(0-, \lambda)$ was calculated using $KL_\mu(\lambda)$ to propagate $L_\mu(0.21, \lambda)$ to the surface using Beers Law [119].

- Remote sensing reflectance $[R_{rs}(\lambda)]$ was computed as $L_w(0+, l)/E_s(0+, \lambda)$.

We then reduced the spectral resolution of the field measurements to match the spectral bands of the WorldView-2 image based on the published spectral response functions (www.digitalglobe.com). Finally, we performed a linear regression between the 22 *in situ* measurements to their corresponding WorldView-2 spectra at the same location and created the gain and offset for each band to effectively remove atmospheric signals from the image.

3.3.2 Datasets

We collected three multispectral satellite images captured by the WorldView-2 (WV-2) satellite. These images have a wavelength between 400-1100 nm and spatial resolution of

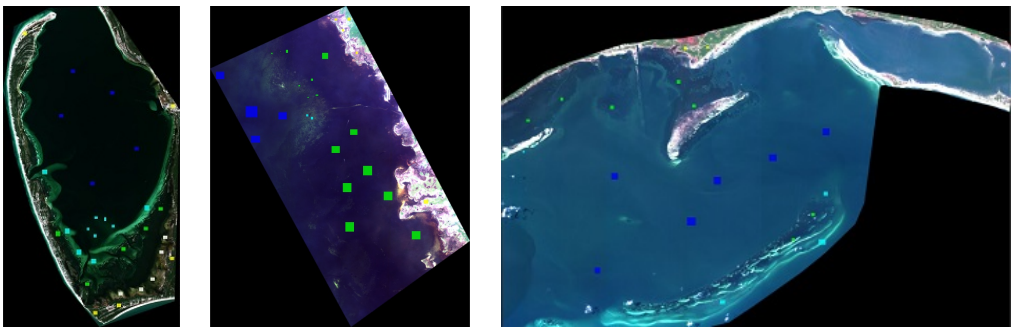
TABLE 1: Band and wavelength information for WV-2 multi-spectral images.

Bands	Wavelength (Nanometre)
Coastal Blue	400 - 450
Blue	450 - 510
Green	510 - 580
Yellow	585 - 625
Red	630 -690
Red Edge	705 - 745
Near-IR1	770 - 895
Near-IR2	860 - 1040

2m shown in Table 1. In this study, an experienced operator [1] selected several regions in each of the three images with highest confidence of the labeling. These regions have been identified as blue, cyan, green and yellow boxes, corresponding to sea, sand, seagrass and land respectively (Fig. 2). At Saint Joseph Bay, an intertidal class was added, and it is represented as white in Fig. 2(a).

3.3.3 Capsule Network

We develop a capsule network for seagrass detection by following the design in [113]. The model has two convolutional layers and 32 convolutional kernels with a size of 2x2x8 for extracting high level features. The extracted features are then fed into the capsule layers, in which a weight matrix of 8x16 is used to find the most similar capsule in the next layer. The



(a) (b) (c)

Fig. 2: Satellite images taken from Saint Joseph Bay(a), Keaton Beach(b) and Saint George Sound(c). The blue, cyan, green, yellow and white boxes correspond to the selected regions belonging to sea, sand, seagrass, land and intertide classes.

last capsule layer, Feature-caps, stores a capsule per class, and each capsule has a total of 16 features. The length of each capsule represents posterior probability for a class. Additionally, the features in Feature-caps are used to reconstruct original images. The reconstruction architecture has 3 fully connected layers with a sigmoid activation function, and the sizes of the layers are 256, 512 and 200, respectively. Output size of the reconstruction structure is the same as that of the input patch (5x5x8).

3.3.4 Transfer Learning

The ultimate goal of this study is to develop a deep learning model that is able to detect seagrass at any location in the world. However, there exists a significant amount of variations in seagrass representation from different satellite images. To resolve this issue, we propose a transfer learning approach such that only a small number of samples are needed

to adapt a trained deep model for predicting seagrass at a new location:

1. Train a capsule network using all the selected data from Saint Joseph Bay.
2. Feed the trained model with few labeled samples from Keaton Beach and extract features from the Feature-caps as new representations for the data.
3. Utilize the new representations to classify the entire Keaton Beach image based on the *1*-nearest neighbor (*1*-NN) rule.
4. Repeat the procedures for the image from Saint George Sound.

3.3.5 Capsule Network as a Generative Model for Data Augmentation

The capsule network has the capability of reconstructing input data from features in Feature-caps. We generate artificial labeled data at new locations to improve model adaptation as follows:

1. Train a capsule network with the selected patches at Saint Joseph Bay and fine-tune the model with a limited number of samples from Keaton Beach.
2. For each of the patches used for fine-tuning the model, extract the 16 corresponding features in the Feature-caps and compute mean (μ_C) and standard deviation (σ_C) for each of the 16 features.
3. For each patch from Keaton Beach, generate a total of 176 new artificial patches by varying each of the features 11 times within the range of $[\mu_C - 2\sigma_C, \mu_C + 2\sigma_C]$.
4. Fine-tune the trained capsule network with these artificial and original patches.

5. Repeat this procedure for 20 iterations, and repeat the same procedure for Saint George Sound.

For comparison purposes, we add random noise within the range of $[\mu_C - 2\sigma_C, \mu_C + 2\sigma_C]$ directly to the patches that are fed to the capsule network, and then we extract their features to classify all the patches from Keaton Beach and Saint George Sound using the *1*-NN rule.

3.3.6 Convolutional Neural Network

A similar method is implemented on CNN for comparison purposes. The CNN model has two convolutional layers with a ReLU activation function and 16 2x2x8 and 64 4x4x16 convolutional kernels, respectively. The convolutional layers are followed by one fully connected layer with 16 hidden units and a soft-max layer to perform classification. We utilize the dropout technique with a probability of 0.1 to reduce over-fitting [120].

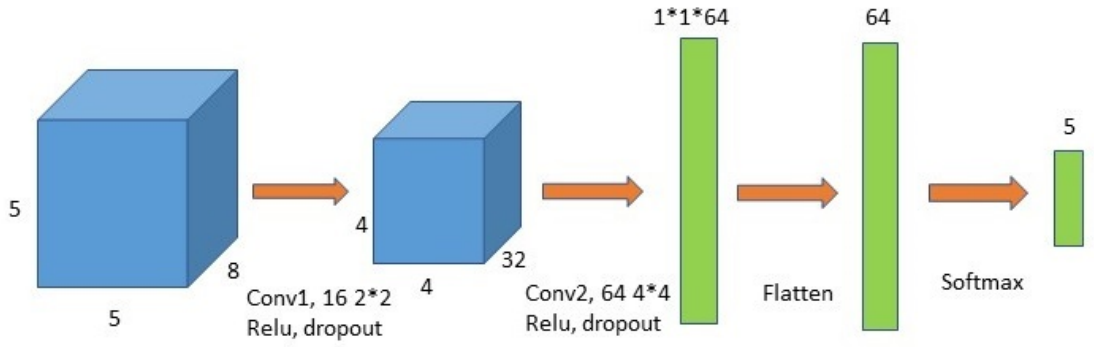


Fig. 3: Deep Convolutional Neural Network.

3.4 Results

3.4.1 Model Structure Determination

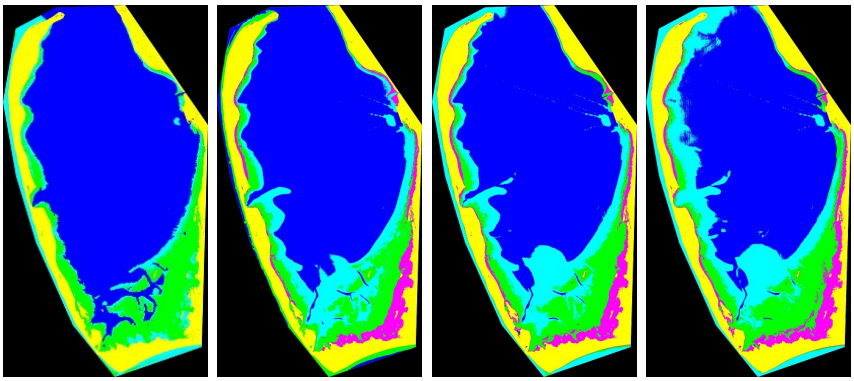
We select Saint Joseph Bay as the primary location to train deep models with the selected regions. To have a fair comparison of the performances between capsule network and CNN, we keep the same number of parameters, 9k, in convolutional layers for both models. In the capsule network, there are 46k parameters for routing and 254k parameters for reconstruction. We train 10 epochs for CNN and 50 epochs for the capsule network to roughly keep the same amount of training for both models.

3.4.2 Cross-validation Results in Selected Regions

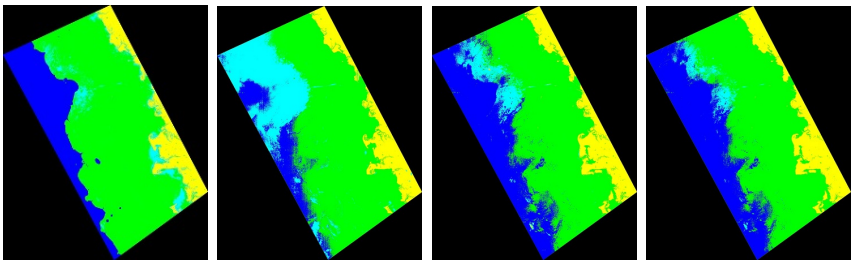
To validate our model, we perform *3*-fold cross-validation (CV) in the selected regions for the three locations separately. In 3-fold CV, we divide all the extracted training patches into three parts. Then, we use two parts for training and one part for testing. We repeat this process until all three parts are tested once. Table 2 shows the classification accuracies for each satellite image using SVM, CNN and capsule network. Additionally, each model is trained with all the patches from the selected regions and then applied to the corresponding whole image as shown in Fig. 4.

3.4.3 Transfer Learning

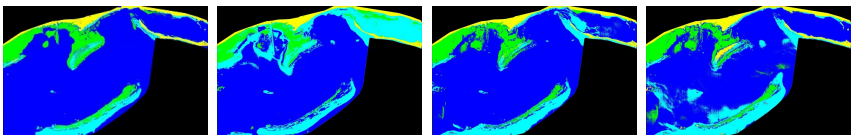
Table 3 shows the classification accuracies in the selected regions by transfer learning with a different number of labeled samples (shots) from new locations. Zero shot transfer learning means applying the deep learning model trained at Saint Joseph Bay directly to



(a)



(b)



(c)

Fig. 4: Three-fold CV results. From left to right, the classification map by the physics model [1], SVM, CNN and capsule network on Saint Joseph Bay (a), Keaton Beach (b) and Saint George Sound (c). The colors blue, cyan, green, yellow and magenta represent sea, sand, seagrass, land and intertide, respectively.

TABLE 2: Three-fold CV results of Saint (St) Joseph Bay, Keaton Beach and Saint (St) George Sound.

Location	SVM	CNN	Capsule Network
St Joseph Bay	90.20%	**99.99%**	99.94%
Keaton Beach	81.13%	97.20%	**99.97%**
St George Sound	76.27%	80.20%	**99.40%**
Mean	82.53%	92.46%	**99.77%**

Keaton Beach and Saint George Sound. It is observed that CNN has better performances in transfer learning.

3.4.4 Capsule Network as a Generative Model for Data Augmentation

We use the capsule network as a generative model to obtain new training data for model adaptation as described in Section 3.3.5. For comparison purposes, we have identified the following cases:

- Regular fine-tuning: We fine-tune the capsule network with a small number of labeled samples (shots) from the new locations. After fine-tuning, we use the transfer learning procedures to classify the rest of the patches.

- Random noise: We add some random noises into the labeled patches to generate artificial patches for transfer learning.

- Generative fine-tuning: We fine-tune the capsule network with a small number of

TABLE 3: Transfer Learning using CNN and capsule network (capsule net) for Keaton Beach and Saint George Sound.

Method	Location	0 Shot	1 Shot	10 Shots	50 Shots	100 Shots
CNN	Keaton Beach	**47.23**%	**56.90**%	**99.56**%	**99.63**%	**99.75%**
	St George Sound	**22.08**%	**75.26**%	**95.92**%	**98.66**%	**98.76%**
Capsule Net	Keaton Beach	38.74%	54.85%	94.45%	96.13%	97.27%
	St George Sound	21.88%	47.58%	89.36%	94.00%	92.96%

labeled samples (shots) from the new locations. After fine-tuning, we generated artificial patches as described in Section 3.3.5 and use the transfer learning procedures to classify the rest of the patches.

Table 4 shows the classification accuracies in the selected regions for each of these cases with a different number of fine-tuning shots. It can be observed that the best accuracies are obtained using generative fine-tuning for most of the cases.

The results displayed in Table 4 show the accuracies for only one iteration in generative fine-tuning. To investigate the effect of the number of iterations on performances, we run the generative fine-tuning method with a different number of iterations in 100 shots deep learning and show the results in Table 5, where the accuracies are obtained in Keaton Beach and Saint George Sound either with generated data only or combined with the original patches. Additionally, we show the classification maps of each method in Fig. 5. The figure shows classification maps of one shot and 100 shots by each of the methods previously

discussed. In the case of generative fine-tuning, we show the results after 20 iterations with the combination of generated and original data.

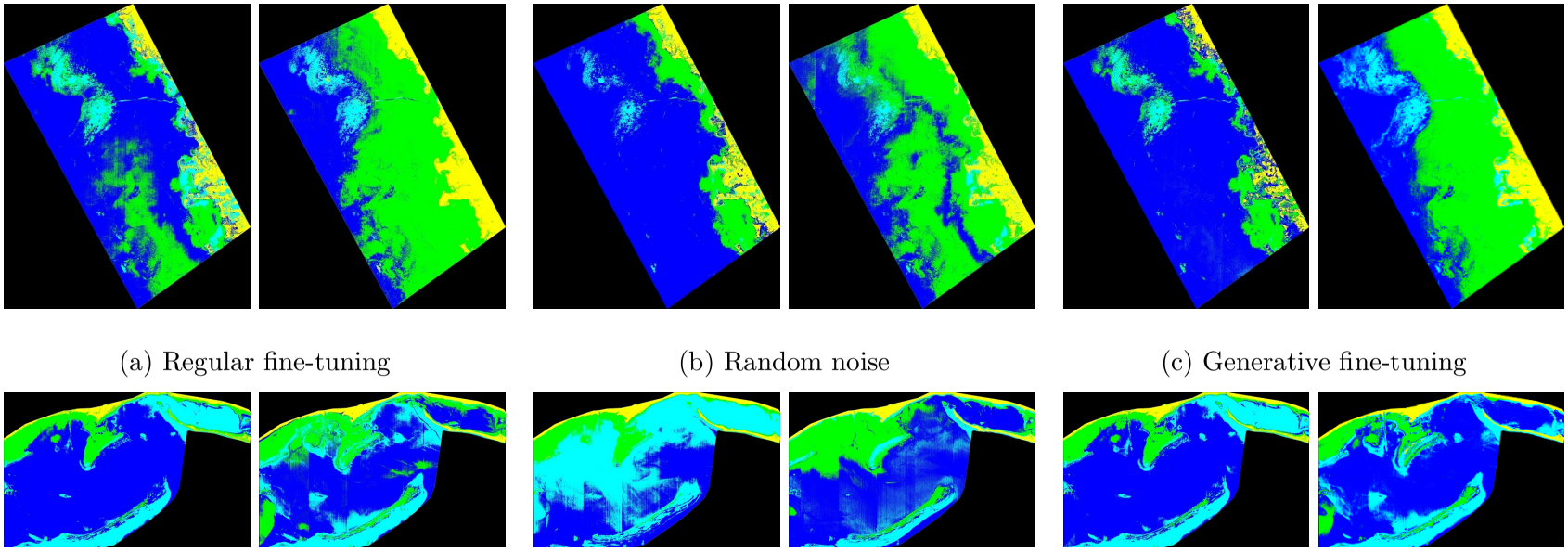
TABLE 4: Transfer Learning results with regular fine-tuning (FT), random noise and generative fine-tuning (FT) for Keaton Beach and Saint (St) George Sound locations. iter=iteration

Method	Location	1 Shot	10 Shots	50 Shots	100 Shots
Regular fine-tuning	Keaton Beach	69.66%	94.00%	96.79%	98.35%
	St George Sound	77.58%	87.39%	99.32%	99.37%
Random Noise	Keaton Beach	54.88%	94.52%	96.10%	97.22%
	St George Sound	47.58%	89.36%	94.00%	92.96%
Generative FT (1-iter)	Keaton Beach	53.11%	89.43%	98.70%	98.85%
	St George Sound	78.20%	95.13%	99.15%	99.42%

3.4.5 Changes in Feature Orientation

We investigated the feature orientation changes in the Feature-caps layer of the capsule network while using each of the fine tuning methods. Figure 6 shows the average values of the features in Feature-caps after each fine-tuning method. The plots in Figure 6 are generated through the following steps:

1. For each class in the data set, collect all image patches and extract the feature matrix computed by the Feature-caps layer in the capsule network, which contains 5 capsules (where 5 is the number of classes), each of them with a size of 16 features.



(a) Regular fine-tuning (b) Random noise (c) Generative fine-tuning

(d) Regular fine-tuning (e) Random noise (f) Generative fine-tuning

Fig. 5: Classification maps by the generative fine-tuning method after 20 iterations at Keaton Beach (a,b,c) and Saint George Sound (d,e,f) with 0 shot (left) and 100 shots (right). The colors blue, cyan, green and yellow represent the patches classified as sea, sand, seagrass and land, respectively.

TABLE 5: Classification results with different number of iterations by the generative fine-tuning method in 100 shots learning. iter=iteration

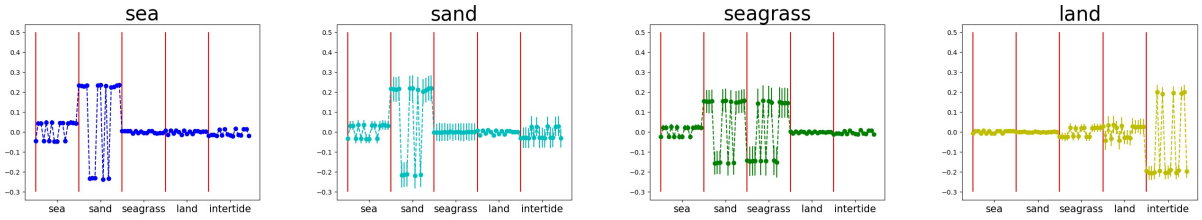
iter	Keaton Beach		Saint George Sound	
	Generated data only	Generated and original data	Generated data only	Generated and original data
5	87.15%	98.82%	86.56%	99.47%
10	85.69%	98.86%	**93.34**%	**99.73**%
15	91.88%	99.09%	78.00%	98.20%
20	**93.00%**	**99.16**%	89.57%	99.67%

2. Reshape each feature matrix into a 1-dimensional vector in which the first 16 numbers are the features corresponding to the first class, the next 16 are the ones corresponding to the second class and so on. This feature vector has a total size of $5 * 16$.

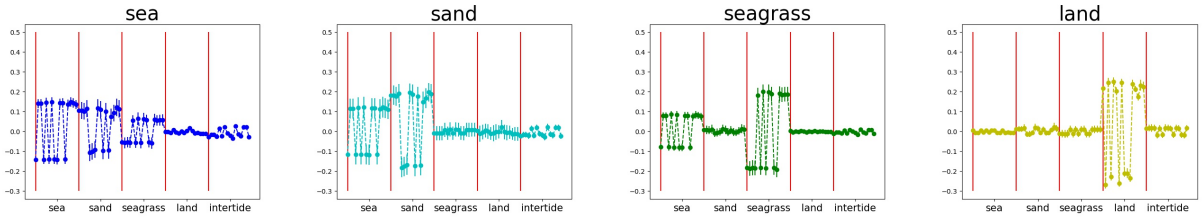
3. Average all the feature vectors belonging to each class and plot them in a 2D graph. Since the probability of an entity belonging to a class is measured by the length of its instantiation parameters (or features), the absolute value of the features belonging to a class should be significantly larger than the rest of the features.

3.5 Discussion

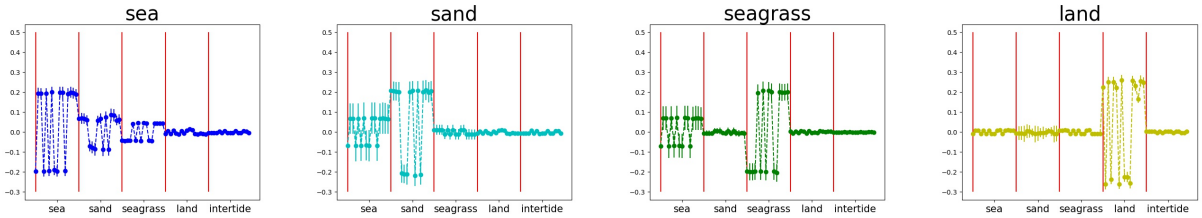
For cross validation results in Table 2, SVM, CNN and the capsule network perform better at Saint Joseph Bay location than at Keaton Beach and Saint George Sound. We



(a) Orientations produced by the model trained at Saint Joseph Bay



(b) Orientations after fine-tuning the model trained at Saint Joseph Bay with 100 shots from Saint George Sound



(c) Orientations after using Capsule Network as a Generative Model (for 20 iterations)

Fig. 6: Features orientation in Feature-caps at Saint George Sound.

select the Saint Joseph Bay as the source location in the experiment of transfer learning and Keaton Beach and Saint George Sound as the target locations. The capsule network outperforms SVM at all the three locations. The capsule network performs slightly better than CNN for two locations whereas in other locations CNN was better. In Fig. 4, the sea class is misclassified as sand in Keaton Beach and ST George Sound by SVM as compared to the physics based approach.

CNN and the capsule network have lower accuracies at Keaton Beach and St George Sound in zero shot and one shot learning. The model trained at Saint Joseph Bay performed poorly at the other two locations because of the variations of class orientation as shown in Fig. 6. One shot transfer learning is not enough to represent the entire orientation changes at different locations. However, with the increase of the number of samples/shots, the classification accuracies were significantly improved (Table 3). In Table 4 and Fig. 5, we have compared the generative fine-tuning approach with regular fine-tuning and random noise approaches. Random noises may not be related to original data and its performances were worse than the generative fine-tuning approach.

In Fig. 6, we have evaluated how the capsule's features are changing in different steps. In an ideal situation if one of the classes is used as input, the capsule representing that class should have higher feature values. In Fig. 6a, we show feature orientations of target location images (Saint George Sound) produced by the model trained with images from the source location (Saint Joseph Bay). For the sea class's capsule features in Fig 6a, the first 16 features should be large because sea patches were used as input. However, the second 16 features are larger because of the location variations between Saint Joseph Bay and Saint George Sound as shown in Fig. 6a. The source model predicted Saint George Sound's sea samples as sand class (the second 16 features) due to location change. Likewise, the source model predicted seagrass and land classes at Saint George Sound as sand and inter-tide classes, respectively. The source model predicted sand class samples correctly as they appears the same at both locations. The capsule feature's orientations also explain the poor zero shot results using capsule-network. After fine-tuning the network with generative

fine-tuning approach for 20 iterations, we can see that capsule features are representing correct classes (Fig. 6c).

We have achieved the best accuracy of 99.16% and 99.67% in Keaton Beach and Saint George Sound locations after 20 iterations in generative fine-tuning. Comparing Table 5 with Table 3, the accuracy is either comparable (99.16% vs. 99.75%) or better (99.67% vs. 98.76%) at both locations in transfer learning by CNN. Using generated data only for the 1-NN rule, the best accuracies we have achieved are 93.00% and 93.34% in Keaton Beach and Saint George Sound, respectively. If we compare the end to end classification map in Fig.s 4 and 5, the generative fine-tuning approach has produced the best results for both locations. In our companion paper [121], we studied seagrass quantification after identification.

CHAPTER 4

SEMI-SUPERVISED DOMAIN ADAPTATION FOR SEAGRASS DETECTION

4.1 Motivation

In this chapter, we propose a semi-supervised domain adaptation for seagrass detection. Our previous models degraded if directly applied to different locations for seagrass detection. A deep learning-based seagrass classifier properly trained on a location usually fails to classify if we feed a new scene to the model. Using a domain expert, it is possible to get a few labeled samples and many unlabeled samples from a new location. Here, we propose to use labeled and unlabeled samples from the target location to adapt the seagrass detection model using a novel semi-supervised domain adaptation method.

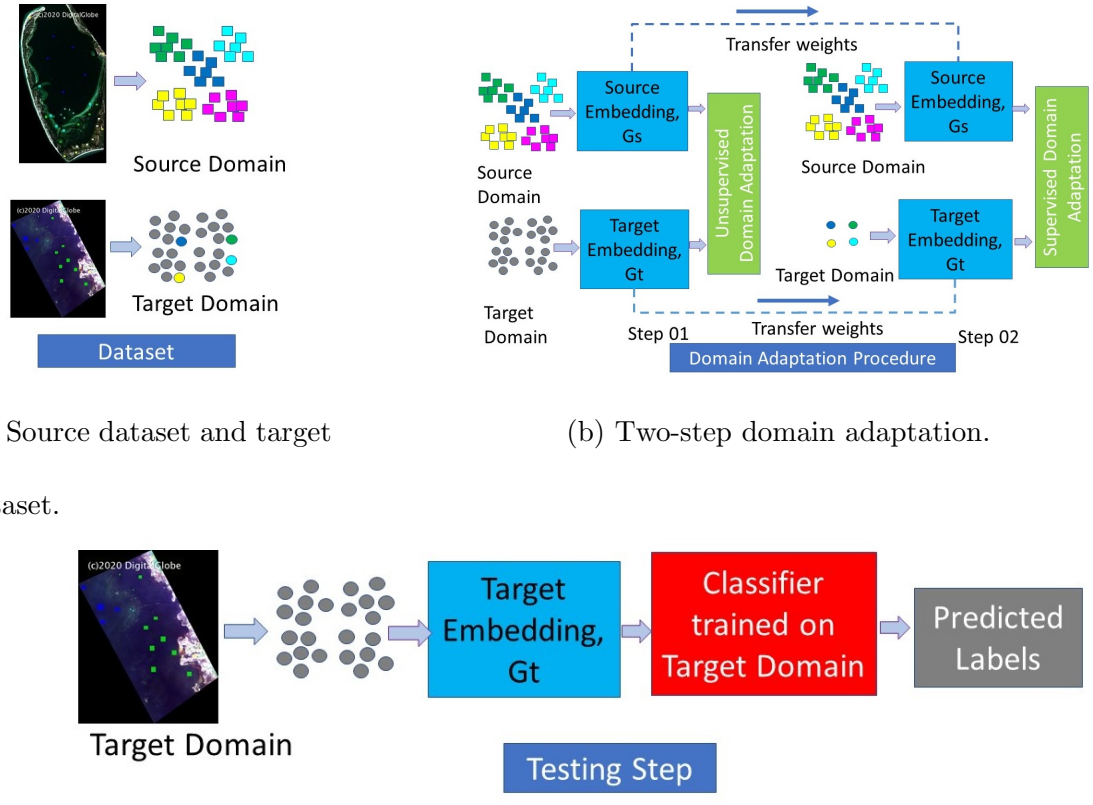
Domain adaptation techniques can be applied if there are not enough labeled data available to train a deep learning model from scratch in a new domain. In domain adaptation, a model in the source domain is first trained using an available large training dataset. A domain adaptation method is then applied to adapt the trained model to a new domain (target domain) without a few labeled samples from the target domain. Tzeng *et. al* proposed an unsupervised domain adaptation method that used the adversarial loss to match source and target domain distributions [101]. Motiian *et. al* proposed a supervised approach for domain adaptation which used the Siamese architecture with a few labeled for domain adaptation [110]. This model learned an embedding function for source and target

data where the two domains were semantically aligned and different classes were maximally separated using a contrastive semantic alignment loss. In this chapter, we propose a semi-supervised domain adaptation method that uses unlabeled samples from the target domain with a GAN loss and few-labeled samples from the target domain with a contrastive semantic alignment loss to detect seagrass in new locations.

4.2 Semi-Supervised Domain Adaptation

4.2.1 System Diagram

The diagram of the proposed domain adaptation method for seagrass detection is shown in Fig. 7. There are enough labelled data in the source domain to train a deep CNN model for seagrass detection while there are only a few labelled samples in the target domain as shown in Fig. 7a. The trained CNN model contains multiple convolutional layers for feature extraction and a fully connected layer for classification. These convolutional layers essentially learn an embedding function, and the fully connected layer takes its outputs for classification. Our proposed system uses two steps to adapt the embedding function trained in the source domain to the target domain as shown in Fig. 7b. In the first step, the proposed model uses unlabeled samples from both domains to modify the target embedding function while keeping the source embedding fixed so that the outputs from both embedding functions have a similar distribution. In the second step, the proposed model pairs labelled samples from the source domain, and a few labelled samples from target domain were used to align class specific distributions among both domains. Once the target embedding function is adapted, a simple classifier can be trained using the few labelled samples from the target



(a) Source dataset and target dataset.

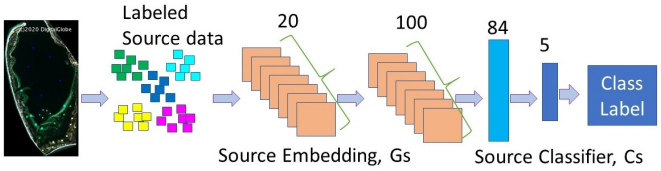
(b) Two-step domain adaptation.

(c) Testing.

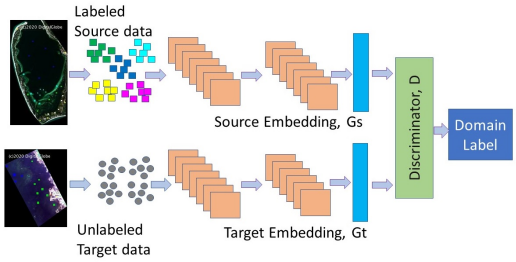
Fig. 7: Diagram of the proposed domain adaptation model for seagrass detection. (a) Datasets from both domains where colored samples are labelled while gray samples are unlabelled (b) Unsupervised adversarial adaptation and supervised contrastive semantic alignment between target and source domains (c) The adapted model used for seagrass detection in target domain.

domain to perform seagrass detection on the remaining target domain (Fig. 7c).

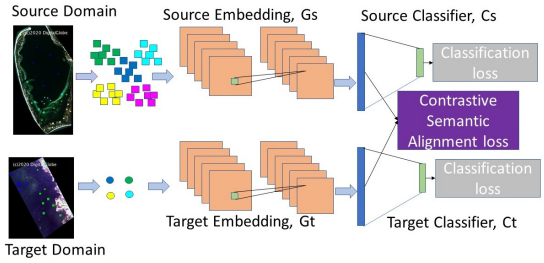
4.2.2 Model Architecture



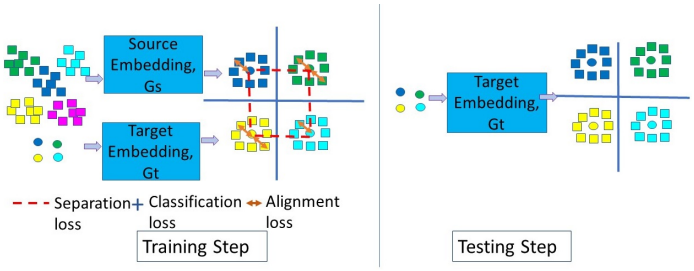
(a) Deep CNN model supervised training in source domain



(b) Unsupervised adversarial domain adaptation to learn embedding function in target domain



(c) Supervised joint domain adaptation based on CCSA loss



(d) CCSA loss training and testing steps

Fig. 8: Proposed semi-supervised domain adaption procedure. (a) Deep CNN model supervised training in source domain (b) Unsupervised adversarial domain adaptation to learn embedding function in target domain (c) Supervised joint domain adaptation based on CCSA loss and (d) CCSA loss training and testing procedures.

Fig. 8 shows the domain adaptation procedures of the proposed method. We first train a deep CNN model in the source domain with labeled data (Fig. 8a), where the CNN model learns an embedding function, G_s, called the source embedding function, and a simple classifier, C_s, for seagrass detection. In the target domain (Fig. 8b), we first use unlabeled data samples from both domains to adapt the target embedding function with a generative adversarial network (GAN) loss such that the discriminator cannot tell from which domain an embedding comes. This step will align marginal data distributions $p(G_s(x^s))$ and $p(G_t(x^t))$ of the source and target domains. In Fig. 8c, we utilize a few labeled samples from the target domain with a classification and a contrastive semantic alignment loss to further adapt the target embedding function such that the class specific data distributions $p(G_s(x^s)|y)$ and $p(G_t(x^t)|y)$ from the two domains are aligned after embedding. Fig. 8d illustrates the training and testing steps for class specific alignment. We will detail each of the steps in the following subsections.

4.2.3 Deep CNN Model Training in Source Domain

Let $D_s = \{X^s, Y^s\}$ and $D_t = \{X^t, Y^t\}$ denote source and target domain datasets, and we assume that there is a limited amount of labeled samples available in the target domain. A source domain deep CNN model is trained with the following classification loss (Fig. 8a),

$$L_c(f_s) = E[l(f_s(X^s), Y^s)] \tag{7}$$

where f_s is a classifier to be trained, E denotes the expectation function and l denotes any related loss functions.

A classifier, f, can be modeled as two functions as $f = G \circ C$, where G is the embedding function from the input image x to the embedding space and C is the function for predicting the class label from the embedding space. Thus, $f_s = G_s \circ C_s$ and $f_t = G_t \circ C_t$ denote the deep CNN model in the source domain and the target domain, respectively.

4.2.4 Adversarial Discriminative Domain Adaptation

By following the idea in Tzeng *et al.* [101], we utilize the GAN loss to adapt the embedding function G_s in the source domain to the target domain. It is assumed that we have source image x^s with label y^s from source domain distribution $p_s(x, y)$, and image x^t from the target domain where we do not have any label information. This unsupervised domain adaptation step tries to learn a target embedding function G_t based on G_s and unlabeled data from both domains. G_t and D in Fig. 8b are trained by MinMax optimization with the GAN loss $L_{adv_D}(X^s, X^t, G_s, G_t)$,

$$L_{adv_D}(X^s, X^t, G_s, G_t) = E_{x^s \sim X^s}[logD(G_s(x^s))] - E_{x^t \sim X^t}[log(1 - D(G_t(x^t)))] \quad (8)$$

where D is the discriminator used in the GAN model [98] and works as a classifier trained by the cross-entropy loss. The source domain samples are labeled as '1' and target domain samples are labeled as '0'. The discriminator, D, determines whether a sample belongs to the source domain or target domain. The target embedding function G_t modifies its parameters using the following generator loss,

$$Min_{G_t} L_{adv_G}(X^s, X^t, D) = -E_{x^t \sim X^t}[logD(G_t(x^t))] \quad (9)$$

This is similar to the standard GAN loss where G_t modifies its weights to mimic source domain sample embeddings to fool the discriminator, D. During training, we keep G_s fixed while changing G_t.

4.2.5 Classification and Contrastive Semantic Alignment

If there is a distribution shift between source and target domains, the source deep CNN model will not perform well in the target domain. We utilize a few labeled samples in the target domain and some labeled samples in the source domain to jointly adapt G_s and G_t using the classification loss and the contrastive semantic alignment (CCSA) loss proposed by Motiian *et at.* [110] as shown in Fig. 8c).

4.2.5.1 Classification Loss

We define the classification loss as,

$$L_C(G \circ C) = E[l(f(X), Y)] \tag{10}$$

This loss function is minimized in the source domain and target domain, respectively, with the selected labeled samples from the corresponding domain. This step will separate samples from different classes in both source and target domains, respectively.

4.2.5.2 Contrastive Semantic Alignment (CSA) Loss

To align class specific embedding between source and target domains, we use the CSA loss to jointly adapt G_t and G_s. The CSA loss in the target domain contains two

components and can be described as,

$$L_{CSA}(G_t) = L_{SA}(G_t) + L_{CS}(G_t) \tag{11}$$

where $L_{SA}(G_t)$ is the semantic alignment loss and $L_{CS}(G_t)$ is a class separation loss. $L_{SA}(G_t)$ is computed as,

$$L_{SA}(G_t) = \sum_{a=1}^{N_c} d(p(G_s(x_a^s)), p(G_t(x_a^t))) \tag{12}$$

where N_c is the number of class labels, and $x_a^s = X^s/\{Y = a\}$ and $x_a^t = X^t/\{Y = a\}$ are conditional random variables. d is a distance metric between the distribution of X_a^s and X_a^t. This semantic alignment loss tries to map source domain and target domain data samples as closely as possible if they carry the same class label. However, there is no guarantee that samples from different domains with different labels will be mapped as far as possible in the embedding space. To overcome this challenge, the class separation loss $L_{CS}(G_t)$ is computed as,

$$L_{CS}(G_t) = \sum_{a,b|a\neq b} k(p(G_s(x_a^s)), p(G_t(x_b^t))) \tag{13}$$

where k is a similarity matrix which adds a penalty when the distribution of x_a^s and x_b^t are close to each other. This encourages samples with different labels from different domains to be mapped as far as possible in the embedding space. Fig. 8d shows the working mechanism of the CSA loss.

During training, the semantic alignment loss (orange arrows) keeps the same class samples from different domains as close as possible. The class separation loss (red dashed line) tries to embed different class samples from different domains as far as possible. The classification loss (blue solid line) ensures high classification accuracy in the embedding space. During testing, we use the trained target mapping function to put the unseen target

samples into domain invariant space. The overall classification and contrastive semantic alignment loss becomes,

$$L_{CCSA}(G_t) = L_C(G_t \circ C_t) + L_{SA}(G_t) + L_{CS}(G_t) \tag{14}$$

Equations (11) to (14) are used to optimize G_t. A similar set of equations are used to optimize G_s such that both embedding functions are jointly adapted.

We paired each labeled sample in the target domain with randomly selected labeled samples in the source domain to compute the loss in Equ. (14), where $d(,)$ in Equ. (12) is Euclidean distance in the embedded space and $k(,)$ in Equ. (13) is a similarity measure defined between samples.

4.2.6 Loss Function Computation

The semantic alignment loss and class separation loss are defined as distance or similarity between distributions. It is not easy to estimate conditional distribution for each class given just a few labelled samples in the target domain. Following the method described in [110], we compute the semantic alignment loss as,

$$d(p(G_s(x_a^s)), p(G_t(x_a^t))) = \sum_{i,j} d(G_s(x_i^s), G_t(x_j^t))) \tag{15}$$

where (x_i^s, x_j^t) are all paired labelled samples in the source and target domains. Each labelled sample in the target domain is paired with many selected labelled samples of the same class in the source domain such that $y_j^t = y_i^s = a$. It helps a single labeled target sample to be paired with many source labelled samples and force target labelled samples to be mapped as close as possible to the same class samples in the source domain. The class separation

loss is calculated as,

$$k(p(G_s(x_a^s)), p(G_t(x_b^t))) = \sum_{i,j} k(G_s(x_i^s), G_t(x_j^t))) \tag{16}$$

where a and b denote class labels and $a \neq b$. Each labelled sample in the target domain is paired with many labelled samples from different classes in the source domain. The distance measure, $d(,)$, is defined as Euclidean distance in the embedded space,

$$d(G_s(x_i^s), G_t(x_j^t)) = \frac{1}{2} \left\| G_s(x_i^s) - G_t(x_j^t) \right\| \tag{17}$$

The similarity measure, $k(,)$, is calculated as,

$$k(G_s(x_i^s), G_t(x_j^t)) = \frac{1}{2} max(0, m - \left\| G_s(x_i^s) - G_t(x_i^t) \right\|)^2 \tag{18}$$

Here we use the Frobenius norm, and m is the margin that specifies the separability in the embedding space. The combination of $L_{SA}(G)$ and $L_{CS}(G)$ is also known as contrastive loss as defined in [110]. Note that we use the CCSA loss to jointly optimize G_t and G_s.

4.3 Experimental Setup

4.3.1 Dataset

We validated the proposed model on three multispectral images captured by the WorldView-2 satellite at three locations in the Florida coastal area: Saint Joseph's Bay (SJB), Keaton Beach (KB) and Saint George Sound (SGS). Each image has eight bands (Coastal Blue, Blue, Green, Yellow, Red, Red Edge, NIR-1 and NIR-2) with spatial resolution of 2 meters. An experienced domain expert labelled some regions for five classes in

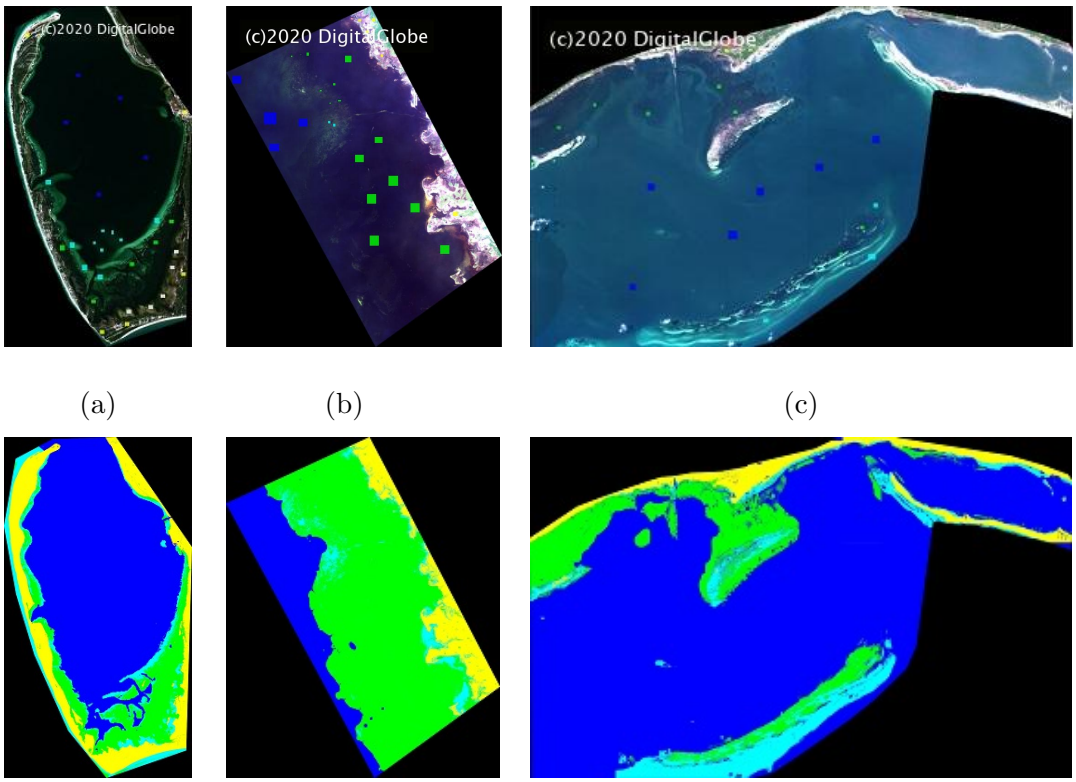


Fig. 9: World-View2 multispectral images collected in Florida at (a) SJB (b) KB and (c) SGS. Labelled region colormap: seagrass → green, sea → blue, sand → cyan, land → yellow and intertidal → magenta. Physics model [1] classification results are shown in (d) SJB (e) KB and (f) SGS.

each image: seagrass, sea, sand, land, and inter tidal as shown as green, blue, cyan, yellow and magenta in Fig. 9. Figs. 9 (d)-(f) show classification results by a physics model [1]. In this study, we trained a deep CNN model at one location and utilize the proposed domain adaptation model to generalize the model to other locations for seagrass detection.

4.3.2 Data Analysis

We compared the spectral signatures of each class in the multispectral WorldView-2 images taken at different locations. To better visualize the high-dimensional spectral information, we utilized the t-distributed stochastic neighbor embedding (t-SNE) algorithm [122] to compress high-dimensional data to 2 dimensions.

4.3.3 k-fold Cross-validation (CV) for Seagrass Detection

At each of the three locations, we performed cross-validation for seagrass detection in the labeled regions. The experimental results gave us performance upper limits for domain adaptation. In k-fold CV, we split data into k parts and kept one part for testing and the remaining parts for training. We repeated this experiment k times such that each part was tested once.

4.3.4 Domain Adaptation between Different Locations

In the domain adaptation experiments, each image was used as source image to train a deep CNN model, and it was then adapted to the other two locations guided by a few labeled samples from the new locations.

4.3.5 Models for Comparison

4.3.5.1 Source-Only

The source-only model used source domain samples to train a deep CNN model, and the model was then directly applied to new locations for seagrass detection.

4.3.5.2 ADDA

Adversarial discriminative domain adaptation (ADDA) [101] adapts the embedding function in the source domain to the target domain based on the GAN loss (Section 4.2.4) with all unlabeled samples in new locations, which was then combined with the classifier trained in the source domain to detect seagrass at the new locations.

4.3.5.3 Source+Target

We trained a deep CNN model in the source domain and used a few labeled data samples from the target domain to fine-tune the model. This is a baseline model for transfer learning.

4.3.5.4 CCSA

This model used the contrastive semantic alignment loss and classification loss to learn the embedding function and classification layers [110]. We used two separate embedding functions that were jointly optimized for the source and target domains (Section 4.2.5).

4.3.5.5 Proposed Model

We first used the GAN loss to adapt the embedding function trained in the source domain. Then the CCSA loss together with a few labeled samples from the target domain were utilized to further adapt the model to new locations as detailed in Section 4.2.5.

4.4 Results

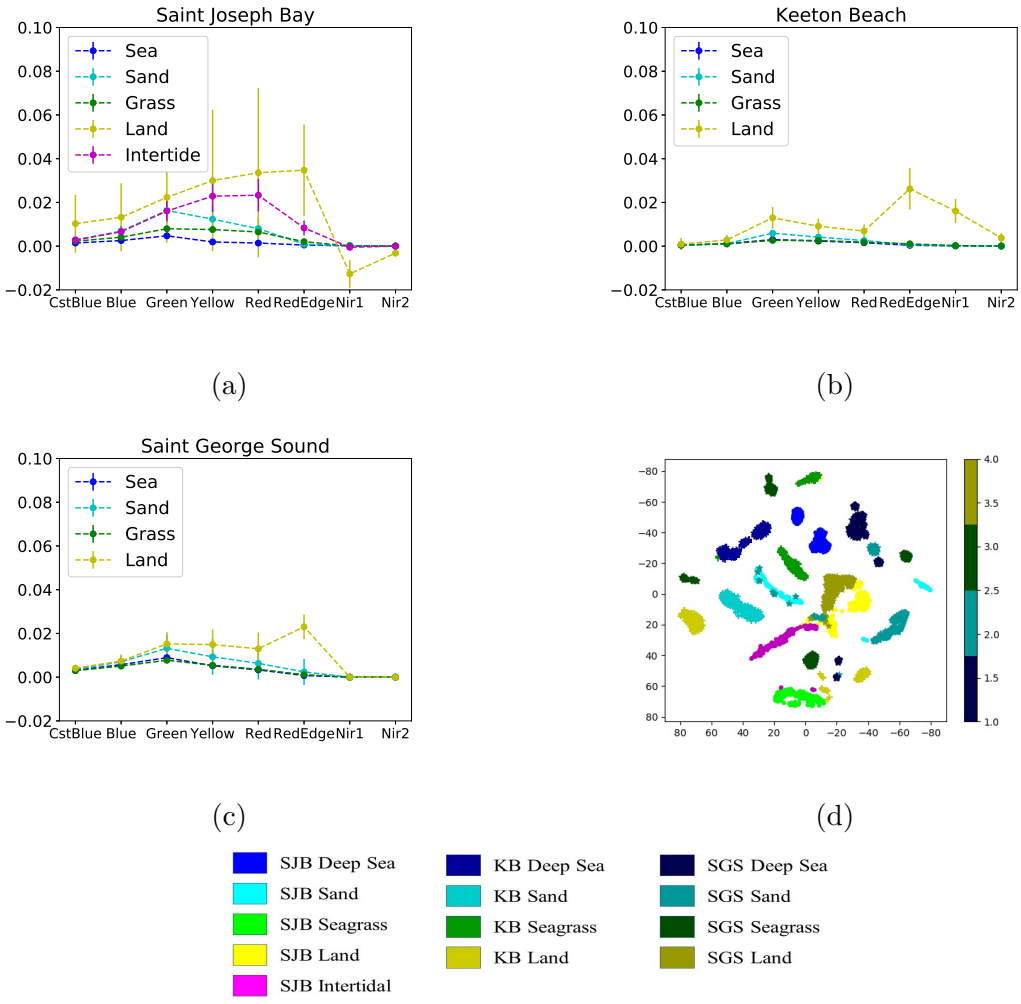


Fig. 10: Atmospherically corrected spectral signature, means and standard deviations in mutlispectral WorldView-2 images for different classes shown at (a) SJB (b) KB and (c) SGS. X-axis represents different bands and Y-axis represents spectral intensity mean and standard deviation. (d) t-SNE plotting of all three locations for different classes. Green, blue, cyan, yellow and magenta are used to represent seagrass, sea, sand, land and intertidal classes. For the t-SNE plotting, the bright shade, dark shade and shade between these two are used to represent SJB, SGS and KB samples, respectively. For seagrass class, we used three different green shades to represent three different locations, e.g.: green, dark green and bright green. Similarly, three shades of yellow, blue, and cyan were used to represent land, sea, and sand classes.

4.4.1 Data Analysis

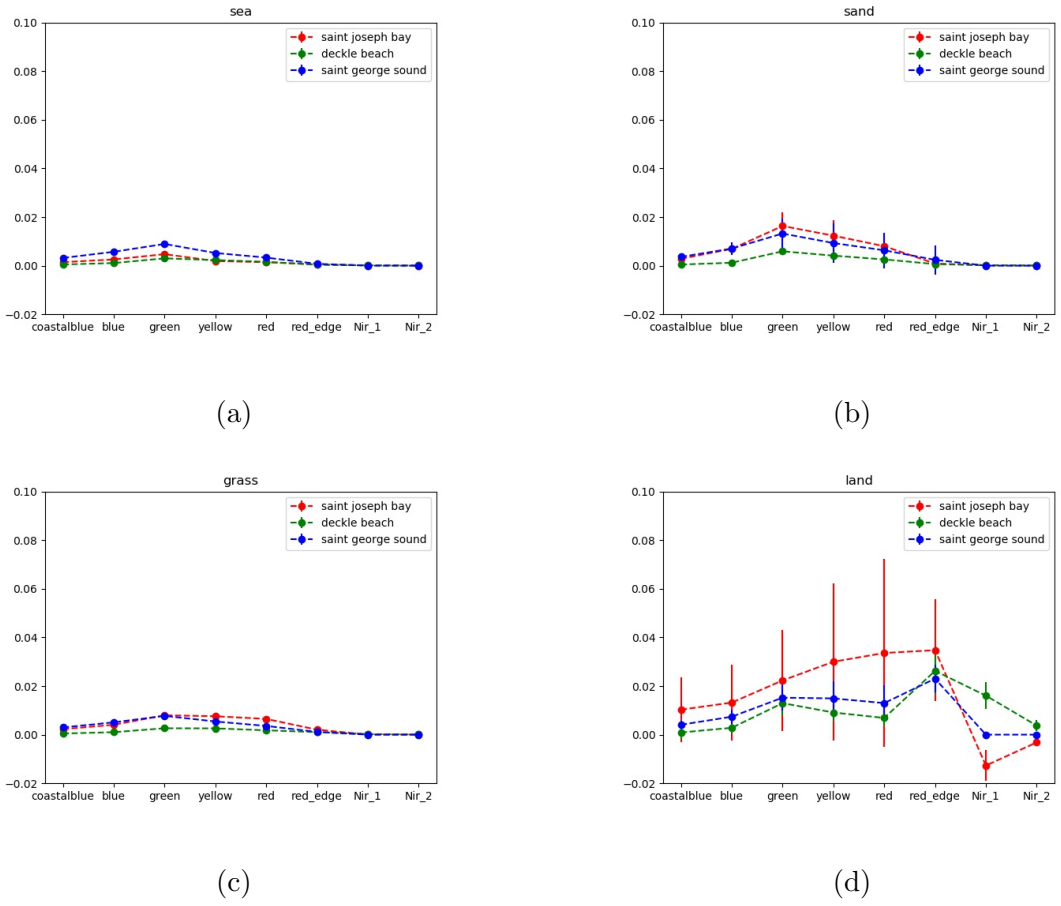


Fig. 11: Mutlispectral eight bands WorldView2 image intensity displayed using mean and standard deviation for Saint Joseph Bay, Keaton Beach and Saint George Sound location into separate classes, (a) sea, (b)sand, (c) seagrass, (d) land. X-axis represents different bands and Y-axis represents spectral intensity values.

Fig. 10 shows atmospherically corrected means and standard deviations of the eight WorldView-2 multispectral bands in the labelled regions at the three locations. Land had the highest mean spectral magnitude as it is located above water. The intertidal class is located between sand and land on spectral magnitude. These classes were followed by sand,

seagrass and sea in spectral magnitude.

It is also observed that spectral signatures of the same class at different locations have different shapes, indicating that there are distribution shifts among locations. In Fig. 10d, we show t-SNE representations for samples from all three locations. We use green, blue, cyan, yellow and magenta to represent seagrass, sea, sand, land and intertidal classes. We use three shades to represent three different locations: the most bright shade, most dark shade and shade between these two to represent SJB, SGS and KB samples, respectively (Fig.10d). Note that there are significant distribution shifts among different locations in different classes.

4.4.2 Hyper-parameter Determination

Deep CNN models take a patch from the multispectral image to predict a class label for the central pixel of the patch. A large patch may cause over-smoothing and requires higher computation power whereas a too small patch may degrade the performance. After some trial and error, we found that a 5x5x8 patch size produced the best results in the three-fold CV experiment. Other parameters were determined in the same way and are listed below.

4.4.2.1 Embedding functions G_s and G_t, in CNN models

Both contain two convolutional layers followed by a flatten layer. The first layer had 20 filters with a size of 2*2*8, and the second layer had 100 filters with a size of 4*4*20. All layers used a ReLu activation function.

4.4.2.2 Classifiers C_s and C_t, in CNN models

Both contained a fully connected layer with 84 hidden units, and the output layer had 5 units with a SoftMax activation function for classification.

4.4.2.3 Source and target data pairing

400 labeled samples from each class in the source domain were randomly selected to pair with the few labeled samples in the target domain to compute the loss function described in Section 4.2.5.

4.4.2.4 Training parameter settings

We trained the source CNN models 50 epochs with a batch size of 128. We trained the unsupervised adversarial domain adaptation step 300 epochs and the CCSA step 240 epochs in all experimentals.

4.4.2.5 Learning rate

We used 0.0002 as the learning rate in all experiments. No dropout layer was used.

4.4.3 Cross-validation (CV)

Table 6 shows 3-fold CV results at the three locations to find upper limits of domain adaption. We achieved 99.99% accuracy at SJB, 99.98% at KB and 99.71% at SGS, respectively. The low variances indicate that the results are very reliable. DCNN performs

similarly and is computationally more efficient as compared to the deep capsule network described in Chapter 3. Therefore, we decide to use DCNN as the backbone for the proposed semi-supervised approach.

TABLE 6: Three-fold cross validation results at SJB, KB and SGS.

Fold no.	SJB (%)	KB (%)	SGS (%)
1st Fold	99.99	99.98	99.83
2nd Fold	99.99	99.98	99.66
3rd Fold	99.99	99.97	99.64
Mean	99.99±0.00	99.98±0.01	99.71±0.10

4.4.4 Domain Adaptation

We conducted six domain adaptation experiments for the three WorldView-2 satellite images as KB → SJB, SJB → KB, SGS → SJB, SJB → SGS, SGS → KB and KB → SGS. Comparison of our proposed model with previous models and results in the selected regions are shown in Table 7 and Table 8. For each domain adaptation experiment, we implemented 6 scenarios including 1 to 5-shot and 10-shot cases (n-shot stands for having n labeled samples from each class). One "shot" means one labeled sample per class in the target domain is used to adapt the model. Each scenario was performed three times with randomly selected labelled samples from the target domain and means, and standard deviations are shown in Table 7 and Table 8. The proposed method achieved the best results in 28 out of 36 scenarios in Table 7 and Table 8. In the 10-shot domain adaptation scenario,

TABLE 7: Classification results in target domain by different methods (All numbers are in %).Source+Target ((f.t.) =ST, Source Only=SO,Proposed Model=PM, 3-fold CV=CV

Shots	Tasks	SJB → KB	KB → SJB	SJB → SGS	SGS → SJB	KB → SGS	SGS → KB
N/A	SO	34.75	45.00	25.08	74.04	15.91	64.14
N/A	ADDA	35.76	42.20	67.80	35.39	78.69	99.43
1-shot	ST	84.78±18.09	76.21±17.64	**79.98±15.05**	74.23±16.61	63.39±16.70	71.13±6.85
	CCSA	71.26±5.43	78.60±6.95	73.34±7.09	76.70±5.65	72.49±1.77	70.82±4.44
	PM	**98.84±0.29**	**86.12±3.55**	71.35±17.20	**80.23±3.04**	**93.32±1.75**	**99.35±0.09**
2-shot	ST	84.78±18.09	76.21±17.64	79.98±15.05	74.23±16.61	63.39±16.70	71.13±6.85
	CCSA	82.56±20.30	87.47±3.30	88.87±7.50	**90.79±1.68**	84.84±3.65	84.31±20.89
	PM	**99.30±0.14**	**91.72±5.88**	**89.65±6.85**	89.70±5.34	**91.55±6.98**	**99.45±0.08**
3-shot	ST	81.88±15.94	84.80±11.38	90.47±8.37	76.36±21.78	72.96±2.12	67.27±7.03
	CCSA	83.95±21.08	88.83±2.76	**90.84±8.39**	87.68±5.17	89.26±6.91	87.27±21.61
	PM	**99.32±0.72**	**94.28±1.90**	89.46±7.13	**92.22±4.68**	**95.20±1.23**	**99.42±0.07**

TABLE 8: Classification results in target domain by different methods (All numbers are in %).Source+Target ((f.t.) =ST, Source Only=SO,Proposed Model=PM, 3-fold CV=CV

4-shot	ST	87.17±19.10	85.60±11.86	67.49±31.52	71.14±18.34	75.78±18.24	65.15±0.10
	CCSA	96.82±3.76	95.26±4.24	**90.93±8.31**	94.79±5.24	91.19±7.72	98.67±1.05
	PM	**99.44±0.46**	**96.31±2.04**	90.92±8.00	**96.84±1.76**	**92.38±6.83**	**99.38±0.12**
5-shot	ST	**99.88±0.07**	**98.20±1.11**	67±30.39	92.40±5.45	71.07±0.40	64.58±0.66
	CCSA	99.72±0.30	95.48±4.22	**91.01±8.14**	95.26±5.56	91.38±7.99	99.43±0.33
	PM	99.07±0.33	95.50±2.84	**91.01±8.00**	**96.27±1.43**	**93.93±4.67**	**99.47±0.12**
10-shot	ST	**99.57±0.67**	86.01±22.57	89.02±15.09	80.91±15.91	71.03±0.10	76.08±20.30
	CCSA	99.42±0.44	99.04±0.42	97.71±0.82	**98.73±0.58**	97.67±1.19	99.56±0.25
	PM	99.34±0.31	**99.09±0.05**	**98.38±0.87**	98.69±0.93	**98.33±0.63**	**99.59±0.32**
N/A	CV	99.98±0.01	99.99±0.00	99.71±0.10	99.99±0.00	99.71±0.10	99.98±0.01

the proposed method approached model upper limits (3-fold CV performances). The second best model is the *Source+Target* (*f.t.*) that achieved the best results in 4 out of 36 scenarios in Table 7 and Table 8.

4.4.5 t-SNE Plotting

We demonstrate how the proposed model maps samples from different domains to the embedding space by utilizing the t-SNE algorithm with the following procedure:

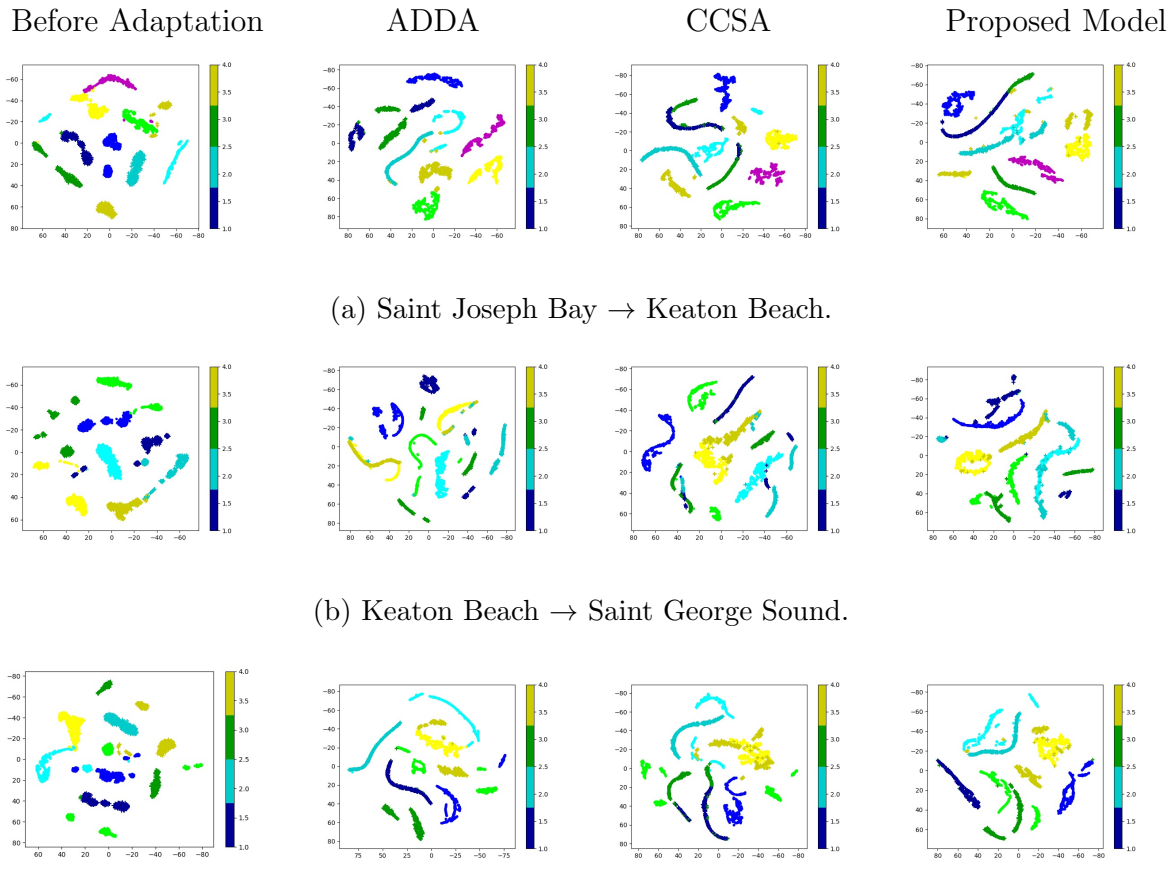
1. Compress the original samples from source and target domains (200 = 5*5*8 dimensions) to 2 dimensions using the t-SNE algorithm (before adaptation),

2. Feed original samples from the source and target domains to the embedding functions, G_s and G_t, respectively, to obtain new representations in the embedding space,

3. Compress the new representations to 2 dimensions using the t-SNE algorithm (after adaptation),

4. Plot the compressed data samples on the 2D plane using different colors for different classes. Use blue, cyan, green, yellow and magenta colors to represent sea, sand, seagrass, land and intertidal class. Utilize two different shades of the same color to denote target and source samples.

t-SNE results are shown in Fig. 12 for three domain adaptation scenarios: SJB → KB, KB→ SGS and SGS → KB. We used 400 samples in each class, respectively, from source and target domains. The proposed model achieved better embedding for sea and seagrass classes as compared to the CCSA model in the scenario of SJB → KB as shown in



(a) Saint Joseph Bay → Keaton Beach.

(b) Keaton Beach → Saint George Sound.

(c) Saint George Sound→ Keaton Beach .

Fig. 12: *t*-SNE plots in embedding space after 1-shot domain adaptation in target domain. (a) SJB → KB (b) KB → SGS and (c) SGS → KB. Green, blue, cyan, yellow and magenta are used to represent seagrass, sea, sand, land and intertidal, respectively. The most bright shade and the most dark shade are used to represent the source and target domain samples, respectively.

Figure 12a. In Fig. 12b and Fig. 12c, similar trends are observed for KB → SGS and SGS → KB cases. The CCSA model incorrectly mapped seagrass samples closer to sea samples and sand samples in the embedding space. The unsupervised domain adaptation method

performed poorly in all the cases as shown in Fig. 12.

4.4.6 Ablation Study

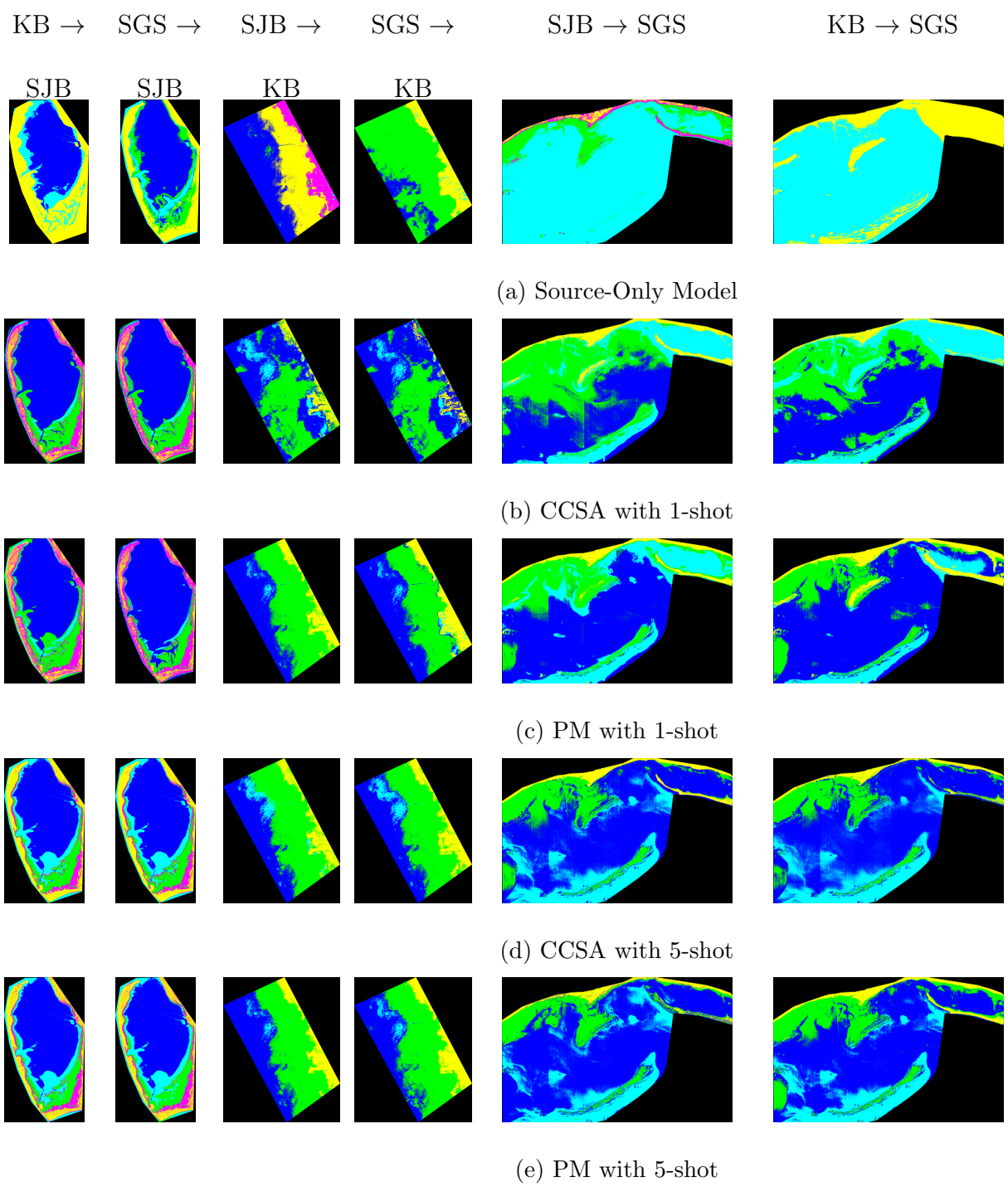
TABLE 9: Ablation study of the proposed method.

Shots	Methods	KB → SGS (%)
	ADDA	78.69
1-shot	CCSA	72.49±1.77
	Proposed Model w/o Joint Optimization	84.77±6.61
	Proposed Model	**93.32±1.75**
2-shot	CCSA	84.84±3.65
	Proposed Model w/o Joint Optimization	91.23±8.87
	Proposed Model	**91.55±6.98**
3-shot	CCSA	89.26±6.91
	Proposed Model w/o Joint Optimization	91.41±7.89
	Proposed Model	**95.20±1.23**
4-shot	CCSA	91.19±7.72
	Proposed Model w/o Joint Optimization	90.58 ±5.21
	Proposed Model	**92.38±6.83**
5-shot	CCSA	91.38±7.99
	Proposed Model w/o Joint Optimization	82.17 ±16.75
	Proposed Model	**93.93±4.67**

Our proposed model contained two loss functions: semantic contrastive alignment loss and GAN loss. If we remove the GAN loss from the proposed model and just use semantic contrastive alignment loss for domain adaptation, the model would be equivalent to the CCSA model. If we remove the contrastive semantic alignment loss from the proposed model then it will be equivalent to the unsupervised ADDA model. Our proposed model also used joint optimization for the source embedding function, G_s, and the target embedding function, G_t, in the supervised domain adaption step. We investigated the three components in the ablation study for KB $\rightarrow$ SGS, and results are shown in Table 9. Note that ADDA does not require labelled samples from the target domain, so only one scenario was performed. The proposed model with all three components achieved the best results.

4.4.7 Classification Maps

The classification maps produced by our proposed model, CCSA approach, and baseline model are shown in Fig. 13. The first row of Fig. 13 represents the base line classification maps where we directly applied classification models trained in source domains to classify target domain images without performing any adaptation. The baseline model performed poorly as compared to the physics model as shown in Fig. 9. The second and fourth rows of Fig. 13 represent classification maps produced by the CCSA model with 1-shot (Fig. 13b) and 5-shot (Fig. 13d), respectively. In this step, we used only contrastive semantic alignment loss to perform the domain adaptation task. The third and the last rows in Fig. 13 represents classification results by the proposed model with 5-shot. We used both the GAN loss and the contrastive semantic alignment loss for domain adaptation. The proposed model with 5-shot produced good classification results as compared to the physics model as



(a) Source-Only Model

(b) CCSA with 1-shot

(c) PM with 1-shot

(d) CCSA with 5-shot

(e) PM with 5-shot

Fig. 13: End to end classification maps produced by domain adaptation based on (a) source model (b) CCSA model with 1-shot (c) proposed model (PM) with 1-shot (d) CCSA model with 5-shot and (e) proposed model (PM) with 5-shot.

shown in Fig. 9. Note that the classification maps shown here are for visualization purposes only as the physics model has 10% error [1].

4.5 Discussion

Our proposed approach produced the best results for 28 out of 36 domain adaptation experimental scenarios as shown in Table 7 and Table 8. For KB → SGS and SGS → KB, our proposed method won all the scenarios. For SJB → KB, our model achieved 98.84% accuracy using just one labelled sample from the target domain, and it is much better than CCSA (71.26%), ADDA (35.76%) and *Source+Target* (*f.t.*) (84.78%). For KB → SGS, our proposed model with one labelled sample from the target domain achieved an accuracy of 93.32% as compared to CCSA (72.49%), ADDA (78.69%) and *Source+Target* (*f.t.*) (63.39%). Similar trends can also be found in SGS → KB in all the 1-shot domain adaptation cases except SJB → SGS, where all the methods achieved similar results. On average, our proposed method won by a large margin.

As we utilize more labeled samples from the target domain, the proposed method can still provide better domain adaptation, winning four or five out of the six experimental scenarios with 2-shot up to 10-shot cases. On average, however, the winning margin decreased as more labeled samples were used for adaptation. For the 10-shot scenario, CCSA and the proposed method achieved similar results, and the results were close to the 3-fold CV results, indicating that adding more labeled samples from the target domain did not provide more benefits.

For most of the scenarios, standard deviations of the proposed method were much

smaller than those of other methods. Our method first utilized a large number of unlabeled samples in both domains to perform domain adaptation. We then used a few labeled samples from the target domain to semantically align class specific distribution in the embedding space. The first step of the method aligned marginal distribution based upon a large number of unlabeled data and worked as a regularizer for the subsequent semantic alignment. Therefore, the proposed method can provide more stable performances.

Fig. 12 shows t-SNE plots for data samples or embeddings in source and target domains before and after domain adaptation. Before adaptation, we can see that data distributions in the source domain and target domain are not aligned. ADDA aligned distributions between the source and target domains, but there is no guarantee that the same class samples from different domains will be mapped closer in the embedding space. With the guidance of labelled samples, CCSA and the proposed model can do a better semantic alignment: the same class samples from different domains can be mapped closer, and the proposed method can do a better job as compared to CCSA.

We only performed the KB $\rightarrow$ SGS case study for ablation as shown in Table 9. All three components in the proposed model are important. With joint optimization, the proposed model became much more stable and achieved much smaller standard deviation in performances for all the scenarios. With more labeled samples from the target domain, CCSA can perform much better than ADDA.

As compared to the physics model classification maps in Figs. 9d, 38c and 9f, the classification maps produced by the proposed model with 5-shots were much better than those from the direct source domain model as shown in Fig. 13a. Classification maps

produced by CCSA with 5-shot (Fig. 13d) are good. However, those produced by CCSA with 1-shot (Fig. 13b) are much worse. Note that the physics model results have 10% error [1], and the classification maps are shown for visualization purposes only. For accurate quantitative assessment of these models, please see results in Table 7 and Table 8 where the accuracy was computed in the labeled regions.

CHAPTER 5

SEAGRASS DETECTION USING UNSUPERVISED DOMAIN ADAPTATION

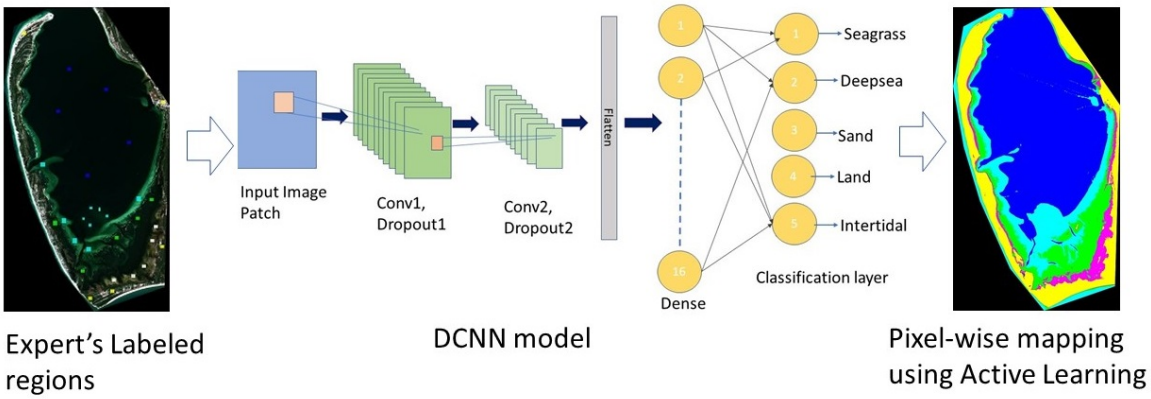
5.1 Background

To monitor the seagrass growth/decline over time, multi-spectral time-series imagery could be a cheap, effective, and automated approach. To train a monitoring model, we typically need to label each temporal image and constantly update the monitoring model, which is time-consuming, and potentially causes delay. In this chapter, we propose an unsupervised domain adaptation technique that will effectively classify time-series images without updating the monitoring model with labeled samples from new images. We first use a human expert's labeled samples from the first time-point image to train a deep learning model. Then, we use the proposed unsupervised domain adaptation technique to classify all future temporal images without new labeled samples. Our proposed model will adjust the seagrass detection model autonomously using unlabeled samples from future images.

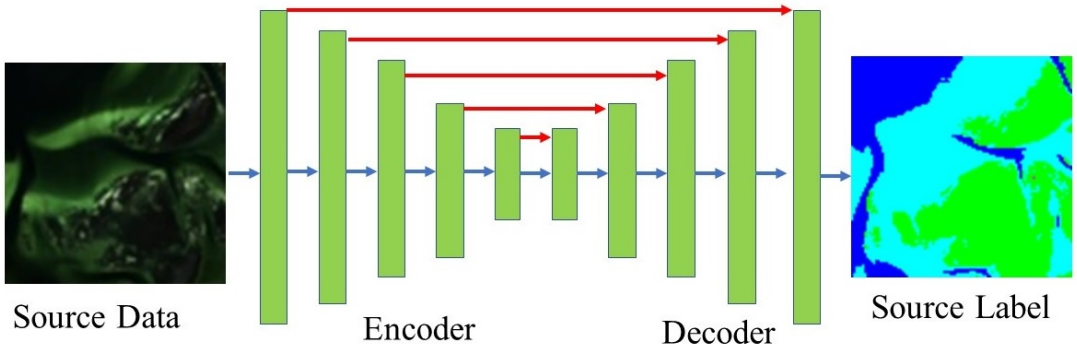
Previously, researchers used both the manual mapping approach [123] and machine learning-based approach [124] for seagrass detection in time-series images. Long *et al.* utilized Landsat imagery with the manual approach for seagrass mapping for 28 years between 1972 to 2000 in Australia [123]. They validated the performance using aerial imagery and in situ measurement data. Knudby *et al.* also utilized a series of Landsat images around Bawe and Chumbe islands in Zanzibar from 1984 to 2009 to detect seagrass [124]. They utilized a

supervised classification approach for seagrass detection [124]. Gullstrom *et al.* utilized the Landsat satellite images between 1986 to 2003 to detect seagrass in an area of Chwaka Bay, Zanzibar (Tanzania) [125]. They used a dark pixel subtraction atmospheric correction algorithm to remove the atmospheric effect in imagery. They utilized a supervised classification algorithm maximum likelihood to produce the seagrass maps [125]. Dekker *et. al* utilized the Landsat imagery between 1988 to 2002 in an Australian lake to detect seagrass using a radiative transfer model [126]. They found that their classification maps are consistent with past surveys and maps [126]. To the best of our knowledge, none of the previous approaches utilized any deep learning approach to detect seagrass in temporal images.

We further evaluate and extend our unsupervised domain adaptation method to address the issue of classifying seagrass in a new location without using any labeled samples from the target image. Domain shift due to the change of the location will be larger compared to the temporal shift because seagrass distribution changes from location to location due to change in water quality, seagrass species, water depth, suspended sediments [19], and colored dissolved organic matter (CDOM) [20]. In our previous experiments of the patch-based deep learning algorithm, we trained a deep learning model with enough labeled samples to effectively classify the same image. When we apply the model in a new location, the well-trained deep learning model usually fails to detect seagrass. In Chapter 4, we utilized a patch-based unsupervised domain adaptation (UDA) approach [101] to classify seagrass in the target location. However, the patch-based UDA method was not able to detect seagrass efficiently at new locations. In this chapter, we propose a pixel-based seagrass detection model in the source domain. The pixel-based classifier learns the relationship between



(a)



(b)

Fig. 14: Proposed Active Learning Approach (a) Pixel-wise mapping using active learning approach (b) Pixel-wise segmentation model training for seagrass detection.

neighboring pixels compared to the patch-based method. Then, we use a GAN-based loss to adapt the source segmentation model to the target domain.

5.2 Methods

5.2.1 Active Learning

We propose an active learning approach to train a pixel-wise seagrass detection method shown in Fig. 14. We first label a smaller percentage of a scene with the help of a domain expert for seagrass detection shown in Fig. 9. To generate the pixel-wise labels of the scene, we use the deep convolutional neural network (DCNN) classifier (Fig. 14a) in the loop. Detailed steps are as follows:

- First, we train a DCNN classifier using the extracted patches from the domain expert's labeled regions of seagrass, sand, deep-sea, intertidal, and land classes. In Fig. 14a, we use the green, cyan, blue, magenta, and yellow colors, respectively, to represent the labeled ROIs. The DCNN model uses a specific patch-size for the Landsat-8 and Worldview-2 images to extract image patches from the domain expert's labeled regions.

- After training, we predict the pixel-wise class label for the whole image using the trained DCNN model to generate a pixel-wise classification map shown in Fig. 14a, which is visually evaluated by the domain expert.

- If the result is satisfactory, we set the classification map as the new ground-truth to train an image segmentation model for seagrass detection as shown in Fig. 14b.

5.2.2 U-NET

Our proposed image segmentation network, U-Net, [15] for seagrass detection is shown in Fig. 15. U-Net learns multiple levels of features of the input image during encoding. During decoding, U-Net uses these learned features to up-sample the embedded representation back into the original image dimension. The U-Net model preserves the

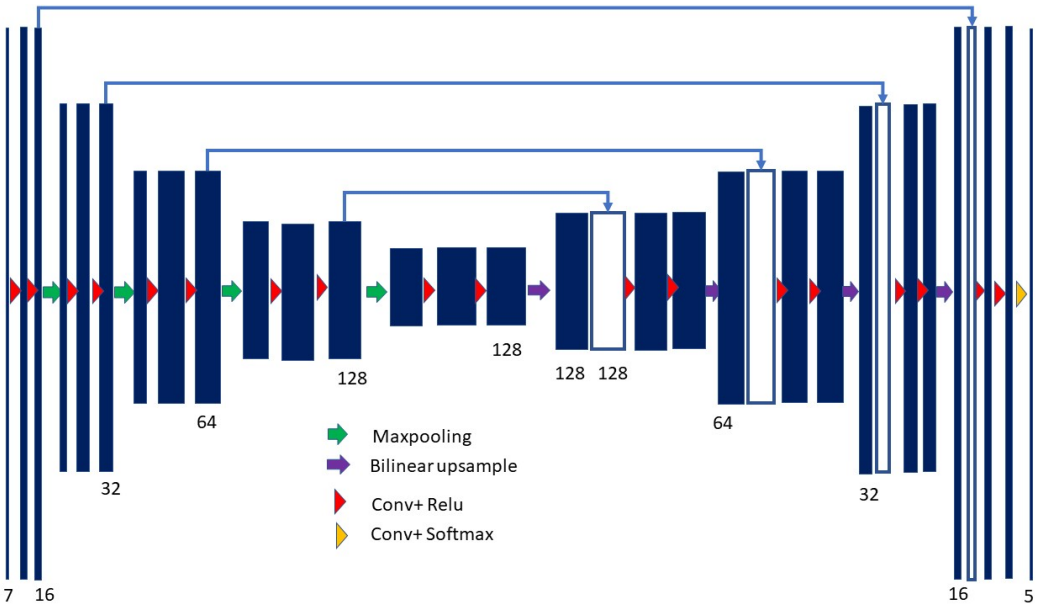


Fig. 15: U-NET architecture for seagrass detection

structural integrity of the input image through skip connections and concatenations of encoder and decoder layers. The U-Net model uses a cross-entropy loss at the last layer for the classification task, or linear loss at the last layer for regression.

Given the multi-spectral image dataset, $D = \{x_s^i, y_s^i\}_{i=1}^n$, where $x_s \in R^{H*W*b}$ is the image data and $y_s \in R^{H*W}$ is the corresponding pixel-based label, we train a segmentation model as shown in Fig. 15. The segmentation model takes an image as input and outputs its pixel-wise label. U-Net has two parts: Encoder, E_s and Decoder, F_s. The last layer of the decoder performs classification. The U-Net model can be defined as,

$$U_{Net} = E_s * F_s \tag{19}$$

The final layer of U-Net uses a soft-max cross-entropy loss to perform classification

defined as,

$$P_k(x) = \frac{exp(a_K(x))}{\sum_{k'=1}^{K}(exp(a_{k'}(x)))} \tag{20}$$

where k is the number of class label, $a_k(x)$ denotes activation output on the last layer of the feature map for class label and $P_k(x)$ is the approximated probability at pixel x. The probability $P_k(x)$ shows close to 1 for the correct class and close to 0 for all other classes.

We define the segmentation cross-entropy loss as follows,

$$\zeta_{seg}(x_s, y_s) = -\sum_{h,w}\sum_{c\in C} Y_s^{h,w,c} log(P_s^{h,w,c}) \tag{21}$$

where Y_s and P_s are the ground-truth and the model predicted output at each pixel.

5.2.3 HR-Net

High-resolution network (HR-Net) is a convolutional neural network that maintains high resolution information through the whole process. HR-Net achieved state-of-the-art performance in a human pose recognition competition in 2019 [16]. Since then HR-Net [16] has been utilized in many applications such as image classification, segmentation, and object detection. HR-Net is a pixel-based fully convolutional network where it assigns each pixel a class label. Previous pixel-based approaches including FCN [14] and U-Net [15] encode an input image into lower dimension embedding through convolutional and pooling layers. Then, the models use dilated convolution or the up-sampling layer to get back to the original image dimension. These encoding and decoding processes lose spatial information in the pooling layers. To tackle this problem, HR-Net keeps multi-scale resolution information through several parallel convolution layers with different numbers of pooling operations. The HR-Net diagram is shown in Fig. 16.

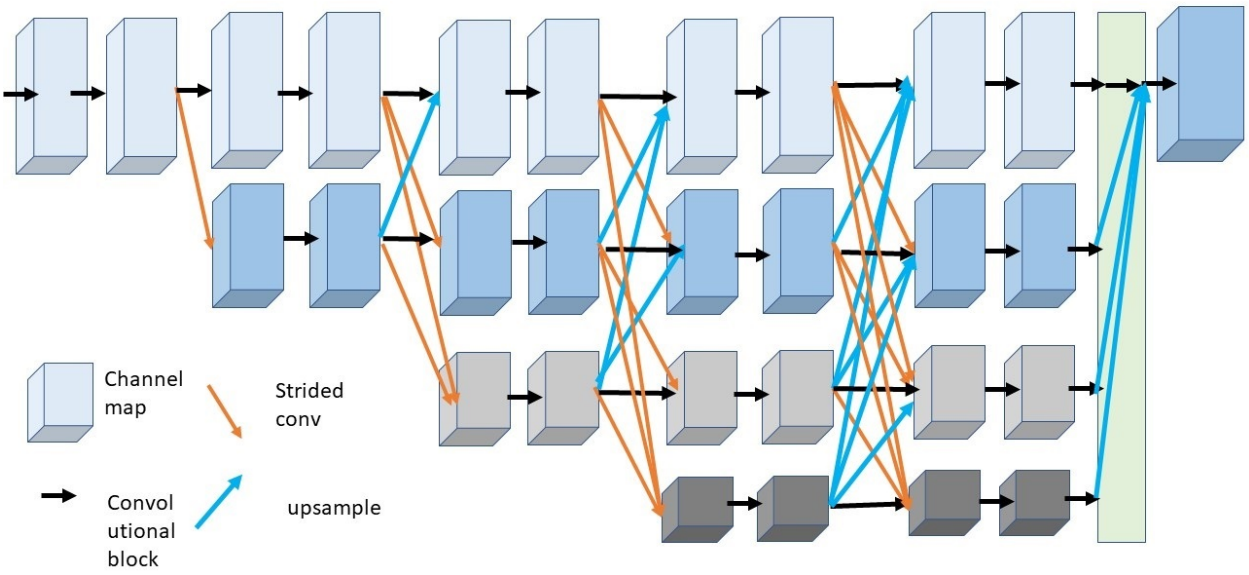
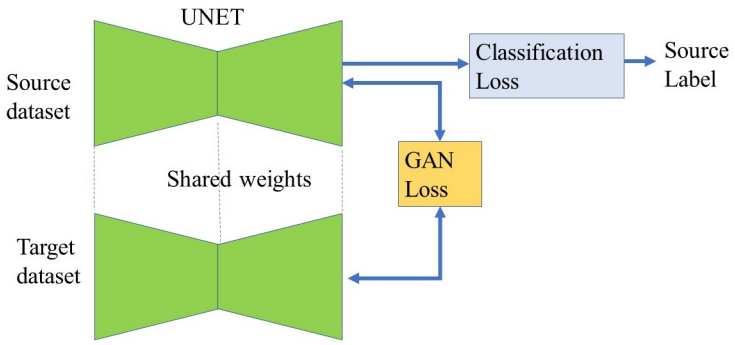


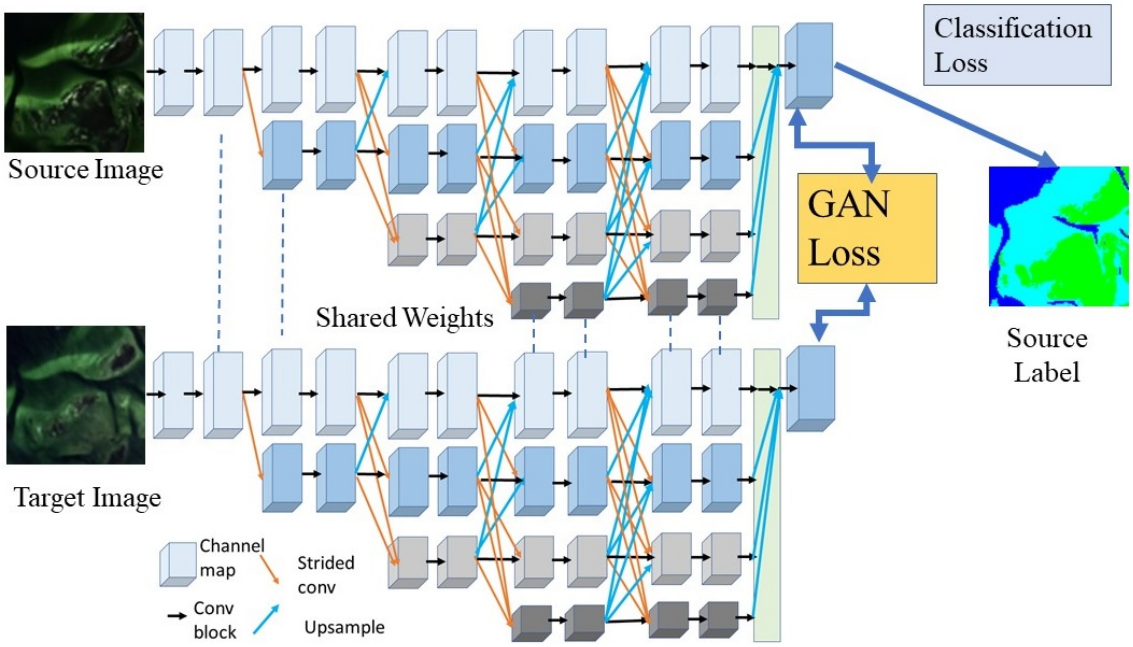
Fig. 16: High-resolution network (HR-Net) architecture for seagrass detection

The HR-Net model starts with a high-resolution image as input and applies multiple convolutional layers without reducing the image dimension as shown in Fig. 16. Then, the HR-Net model adds a high to low-resolution path using a strided convolution. From the second stage, each stage consists of layers from the previous stage in the same path and layers from an added lower resolution path. The parallel paths share information through a fusion mechanism. To perform a fusion, the lower resolution layers are up-sampled using a bi-linear interpolation operation followed by a 1-by-1 convolution. At the last stage in HR-Net, it fuses all paths by re-sampling with bi-linear interpolation followed by the 1-by-1 convolution. As a result, the HR-Net model can combine/concatenate all the four resolution representations of the input image as shown in Fig. 16.

5.2.4 UDA in the soft-max layer



(a) UDA for the U-Net model.



(b) UDA for the temporal images using HRNet model

Fig. 17: Unsupervised domain adaptation method in the soft-max layers for (a) U-Net (b) HR-Net

We also propose to use an unsupervised domain adaptation approach in the soft-max layer to adapt the segmentation network in the target domain shown in Fig. 17a. The segmentation network G is first trained on the source domain images x_s and its labels y_s

with a cross-entropy loss ζ_{seg} in equation 21. The segmentation soft-max output from the model for the source image is $P_s = G(x_s) \in R^{H*W*C}$, where c is the number of classes. For target images, we feed the samples to the segmentation model G and get the soft-max output $P_t = G(x_t)$. The overall objective is to make the segmentation prediction performance in both the source domain and target domain the same using the GAN loss. We feed the soft-max output of the source image P_s into a discriminator network D using a cross-entropy loss. We use the following loss to train the discriminator $\zeta_d(P)$,

$$\zeta_d(P) = -\sum_{h,w}(1-z)log(D(P_t)^{h,w,0}) + zlog(D(P_s)^{h,w,1}) \tag{22}$$

where $z = 1$ if the sample is from the source domain and $z = 0$ if from the target domain. To make the distribution of P_t closer to P_s, we use the following adversarial loss,

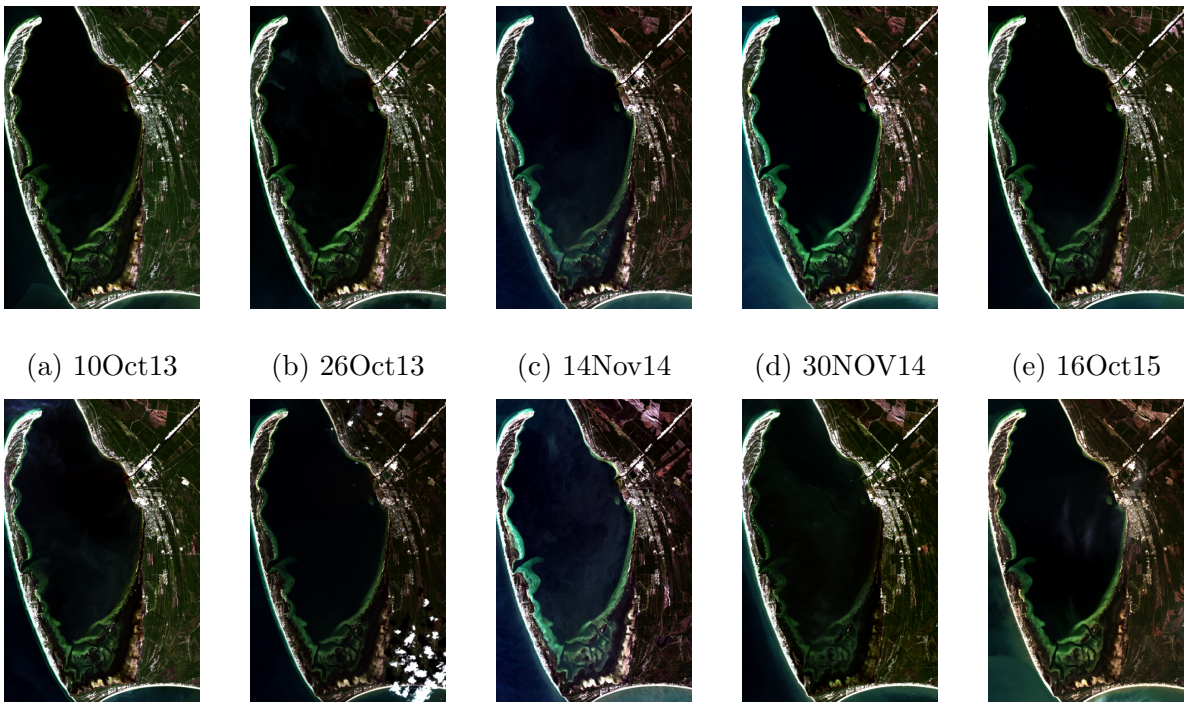
$$\zeta_{adv}(x_t) = -\sum_{h,w} log(D(P_t)^{h,w,1}) \tag{23}$$

Using this loss $\zeta_{adv}(x_t)$, the segmentation network G modifies its parameters to fool the discriminator D by maximizing the probability of target prediction P_t considered to be the source prediction. We learn the mapping function for the target domain to match the source domain features using a GAN loss, but we still want these layers to maintain the segmentation properties. Thus, we jointly optimize both segmentation loss and adversarial loss for the target domain. The overall loss for the UDA in the soft-max layer will be,

$$\zeta = \zeta_{seg}(x_s, y_s) + \lambda_{adv}\zeta_{adv}(x_t) \tag{24}$$

where λ_{adv} is the weight to balance the segmentation loss ζ_{seg} and adversarial loss ζ_{adv}.

5.3 Experiment Setup



(a) 10Oct13 (b) 26Oct13 (c) 14Nov14 (d) 30NOV14 (e) 16Oct15

(f) 02Oct16 (g) 06Nov17 (h) 14Mar18 (i) 25Sep19 (j) 28Nov19

Fig. 18: Landsat-8 times-series images for Saint Joseph Bay displayed using red, green, and blue bands: (a) 10Oct13 (b) 26Oct13 (c) 14Nov14 (d) 30Nov14 (e) 16Oct15 (f) 02Oct16 (g) 06Nov17 (h) 14Mar18 (i) 25Sep19 (j) 28Nov19. Where we use a date-month-year format in the sub-caption and Mar, Sep, Oct, Nov to represent the March, September, October, November months, respectively.

5.3.1 Temporal Landsat-8 Images

We develop an unsupervised domain adaptation-based seagrass detection method in time-series Landsat-8 images with a spatial resolution of $30m$. These images have coastal aerosol, blue, green, red, near infrared (NIR), short-wave infrared (SWIR)-1 and short-wave infrared (SWIR)-2 bands shown in Table 10, and were captured between 2013 and 2019

TABLE 10: Band and wavelength information for Landsat-8 multi-spectral images.

Bands	Wavelength(Micrometre)
Coastal aerosol	0.43-0.45
Blue	0.45-0.51
Green	0.53-0.59
Red	0.64-0.67
Near Infrared (NIR)	0.85-0.88
SWIR-1	1.57-1.65
SWIR-2	2.11-2.29

from Saint Joseph's Bay (SJB) location shown in Fig. 18. These images were processed with operational land imager (OLI) and collection-1 level-1 terrain and precision data (L1TP). To avoid seasonal variability, images collected in the autumn between September and November are considered. We utilize ten scenes between 2013 and 2019 from the Landsat-8 satellite. For the years 2018 and 2019, we use scenes from other times of the year due to cloud contamination in the autumn period. We use a date-month-year format and Mar, Sep, Oct, and Nov short forms to represent the March, September, October, and November months, respectively, in Fig. 18. These images were downloaded from the United States Geological Survey (USGS) Earth Explorer website. We utilize a dark object subtraction algorithm described in [127] to perform the atmospheric correction. The dark object pixels are retrieved using the NIR band. An expert operator labeled five classes in these images: seagrass, sand, deep-sea, intertidal and land shown in Fig. 19.

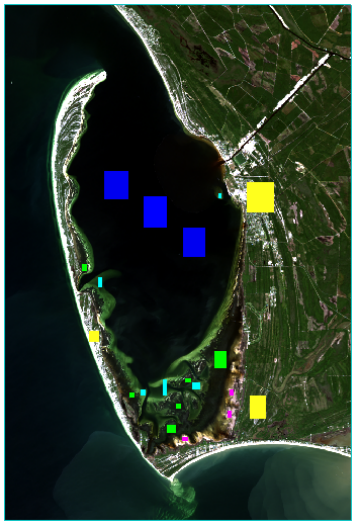


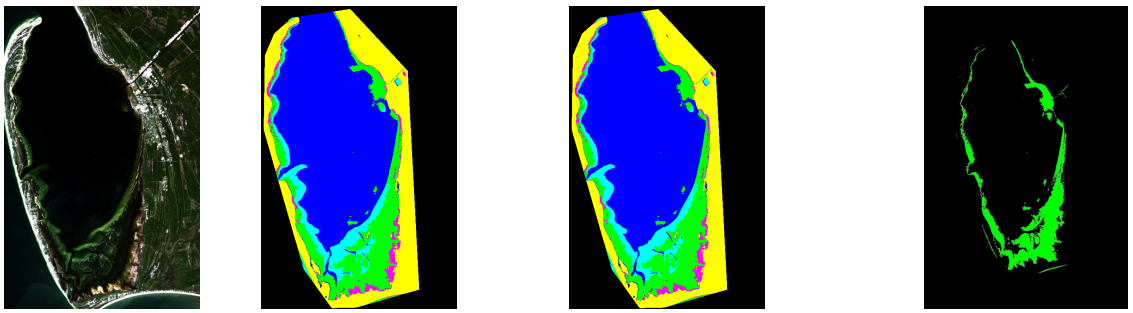
Fig. 19: ROIs defined in the 10 October 2013 Landsat-8 image by domain experts. ROI color-map: seagrass → green, sea → blue, sand → cyan, land → yellow and intertidal → magenta.

5.4 Temporal Image Results

We develop a seagrass detection method in the time-series images with temporal shift using the unsupervised domain adaptation (UDA) method described in Section 5.2.4. We first train a high resolution network (HR-Net) using an active learning approach in the source domain. Then, we utilize the unsupervised domain adaptation method for seagrass detection in all future temporal-shift images of the same location.

5.4.1 Active learning and segmentation model for source domain

In the source domain, we use the regions selected by domain experts to train a deep convolutional neural network (DCNN) for five classes. We use a $3 * 3 * 7$ input patch size to extract patches from the selected regions. The DCNN model has six layers: two

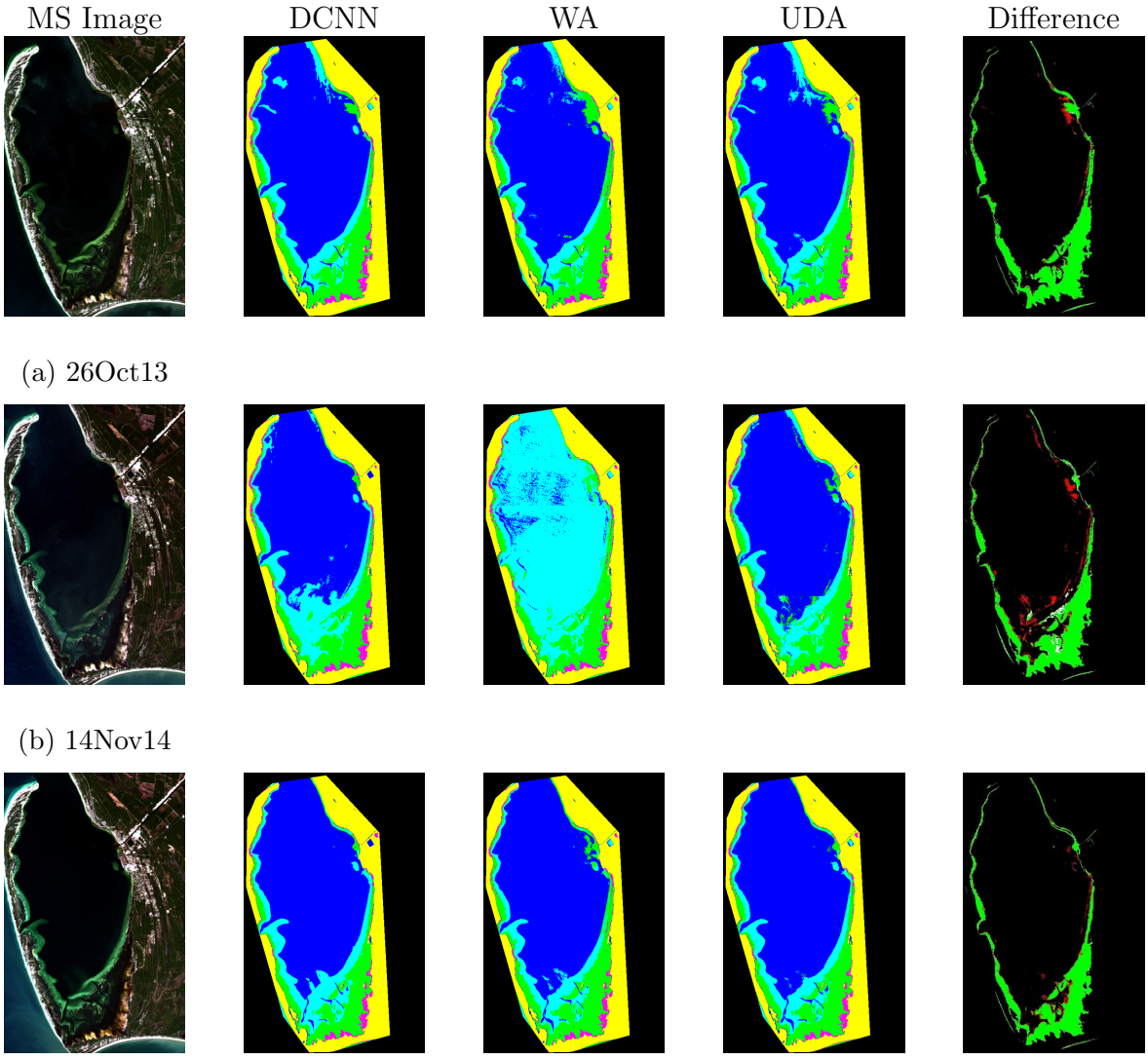


(a) 10Oct13 (b) DCNN (c) HRNet (d) difference

Fig. 20: Source image classification results using high-resolution network (HR-Net): (a) Source multispectral (MS) image 10 October 2013 (b) DCNN classification map (C) HR-Net classification map (d) seagrass difference map between DCNN and HR-Net (color map: green=agreement, red= false positive, white=false negative)

convolutional layers with a ReLU activation function, two dropout layers, one flatten layer and a softmax layer. The first convolutional layer uses 32 convolutional kernels of a size of 1-by-1 followed by a dropout layer with 0.01 dropout probability. The second convolutional layer uses 16 convolutional kernels of a size of 3-by-3 followed by a dropout layer with a probability of 0.01. Then, we use a flatten layer to convert it to vector features. At the last layer, we use a fully connected layer to perform classification for the five classes. We train the DCNN model for 500 epochs. The trained DCNN model is used to classify all the available unlabeled samples from the same image.

We use the trained DCNN to generate the pixel-wise classification map for the 10 October 2013 image using the active learning approach described in Section 5.2.1. Then, we use the active learning generated pixel-wise map to train the segmentation model in the source domain. We train a HR-Net model as the source segmentation model. We use an

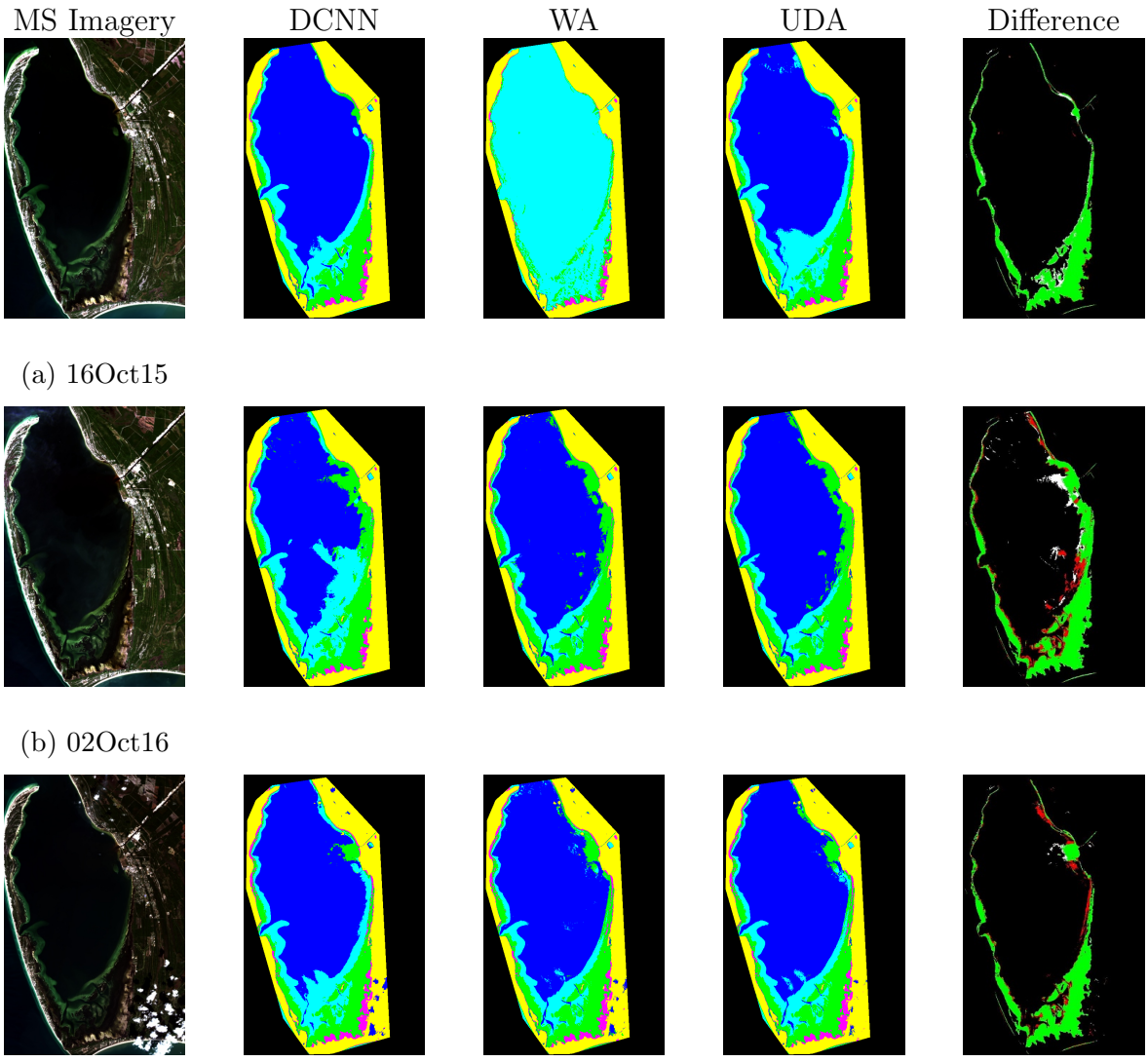


(a) 26Oct13

(b) 14Nov14

(c) 30NOV14

Fig. 21: Unsupervised domain adaptation (UDA) results at SJB for target time-series images : (a) 26 October 2013, (b) 14 November 2014, (c) 30 November 2014. Column list: MS imagery, DCNN classification maps, without adaptation classification map (WA), UDA classification map, seagrass difference map between DCNN and UDA (color map: green=agreement, red= false positive, white=false negative) chronologically.

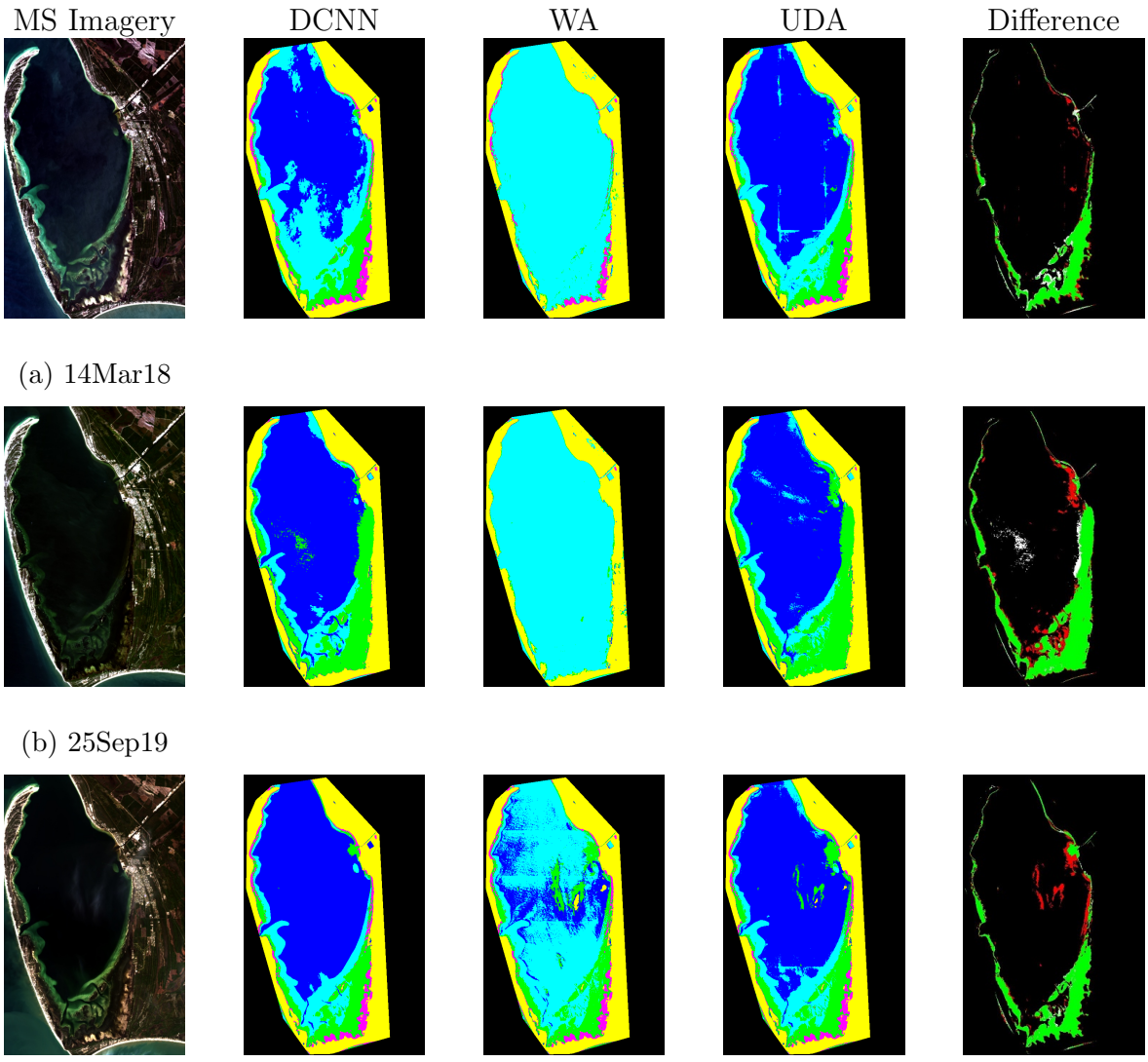


(a) 16Oct15

(b) 02Oct16

(c) 06Nov17

Fig. 22: UDA results for target images : (a) 16 October 2015, (b) 02 October 16, (c) 06 November 2017. Column list: MS imagery, DCNN classification maps, without adaptation classification map (WA), UDA classification map, seagrass difference map between DCNN and UDA (color map: green=agreement, red= false positive, white=false negative) chronologically.



(a) 14Mar18

(b) 25Sep19

(c) 28Nov19

Fig. 23: UDA results for target images : (a) 14 March 2018, (b) 25 September 2019, (c) 28 November 2019. Column list: MS imagery, DCNN classification map, Without Adaptation classification map (WA), UDA classification map, seagrass difference map between DCNN and UDA (color map: green=matched, red= FP, white=FN) chronologically

TABLE 11: HR-Net training results in 10 October 2013 Landsat-8 images. We compare HR-Net with DCNN based on overall accuracy (OA), mean F-1 score for five classes and recall, precision, F-1 score and seagrass area in pixels for seagrass class.

Location	10 October 2013
Overall Accuracy	99.89
Overall mean F-1score	0.9965
Precision seagrass	0.9981
Recall Seagrass	0.9972
F1-score seagrass	0.9976
Total seagrass area DCNN	39266
Total seagrass area HRNet	39301

input size of 256 by 256, batch size of 4, epochs of 110 and 1000 samples per class hyper-parameters to train the HR-Net model in the source domain. The training performance in the source model is shown in Fig. 20. We compare the HR-Net model with the DCNN model as shown in Table 11. We evaluate overall accuracy (OA), F-1 score metrics across five classes, and precision, recall, F-1 score, total seagrass area (TSP) for the seagrass regions in Table 11. We use a baseline model for the target domain where we directly use the trained source model to classify the target domain images (future temporal images) and label them as model without adaptation (WA) in Fig. 21, Fig. 22 and Fig. 23.

5.4.2 UDA in Target Domains

TABLE 12: UDA results in SJB's 2013 to 2019 Landsat-8 times-series images. We compare HR-Net with DCNN based on overall accuracy (OA), mean F-1 score for five classes and recall, precision, F-1 score and seagrass area in pixels for seagrass class. TSP: total seagrass pixels, OMF: Overall mean F-1 score, OA: Overall Accuracy, M: March, S: September, O: October, and N: November.

Location	26O13	14N14	30N14	16O15	02O16	06N17	14M18	25S19	28N19
OA	96.87	92.65	96.77	94.58	88.68	94.31	84.88	91.76	93.45
OM F-1	0.9450	0.8915	0.9291	0.9057	0.8407	0.9111	0.8311	0.7829	0.8983
Precision	0.9771	0.9309	0.9721	0.8906	0.8923	0.9585	0.8585	0.9108	0.9511
Recall	0.9114	0.8344	0.8816	0.9550	0.8052	0.8325	0.8228	0.8004	0.7719
F-1	0.9431	0.8800	0.9246	0.9217	0.8465	0.8910	0.8403	0.8520	0.8522
TSP DCNN	33295	32021	29375	31839	40315	30645	23788	41561	26584
TSP UDA	35697	35725	32392	29692	44675	35282	24820	47298	32756

We separately apply the UDA method for seagrass detection in each target temporal image. The proposed UDA approach for temporal images is shown in Fig. 35. We first initialize the target model with the learned source model weights. Then, we perform the proposed unsupervised domain adaptation method in the layer before the softmax layer mentioned in Section 5.2.4. We use a GAN-based loss to modify the target model's weight so that distribution of source and distribution of target data matches in the layer before the softmax layer. Then, we can utilize the source seagrass detection model in the target domain. The domain discriminator classifies whether the samples are coming from the source domain

or the target domain. The target model tries to fool the discriminator by changing its weights using the feedback/loss given by the discriminator. The discriminator tries to improve its performance so that it can correctly discriminate between the source and the target domain samples. The segmentation model and discriminator compete with each other to improve their performances. We also fine-tune the parameters of the target model's weights with the source image and corresponding label to maintain the classification capability. We repeat these steps for each temporal image from 2013 to 2019 to detect seagrass as shown in Fig. 21, Fig. 22 and Fig. 23.

In Fig. 21, Fig. 22 and Fig. 23, we represent the first column as multi-spectral (MS) images in red, green and blue bands after Gaussian stretching, the second column as the DCNN classification maps, the third column as the without adaptation (WA) classification maps for the source trained model (HR-Net), the fourth column as the UDA classification maps, and 5^{th} column as the difference maps between DCNN and UDA. In the without adaptation scenario, we directly apply the source HR-Net model to classify target time-series imagery. In the fifth column, we use the green color to display points where DCNN and UDA models both agree that a pixel is seagrass. The red color represents false positive (FP) points where UDA predicts seagrass while DCNN disagrees. The white color represents false negative (FN) where DCNN predicts seagrass while UDA disagrees. We utilize OA, mean F-1 score metrics for five classes and recall, precision, F-1 score and seagrass area pixels for the seagrass class to compare UDA with DCNN as shown in Table 12.

5.5 Discussion

We propose an active learning-based approach to train an image segmentation model

in the source domain. The active learning-based approach uses a DCNN model in the loop to label all pixels in a scene because it is impossible for humans to manually label all pixels. To evaluate the seagrass mapping performance of the DCNN model, expert operators visually assess the DCNN model performance. Once the expert operators are satisfied with the results, we use the classified maps as ground truth to train the image segmentation model. In Fig. 19, we show the initial selected regions used in the 10 October 13 image to train the DCNN model versus the model predicted pixel-wise classification map as shown in Fig. 20b. The DCNN model was able to produce accurate mapping along boundaries among different classes which is impossible for a human operator. This proves the DCNN model can perform better than an expert human operator. We also achieve a similar performance in the WV-2 imagery.

We train a HR-Net model using the 10 October 2013 Landsat-8 image of SJB and corresponding pixel-wise ground truth as shown in Fig. 19 and Fig. 20b respectively. We train the HR-Net model for 100 epochs and evaluate the training performance as shown in Fig. 20c, Fig. 20d and Table 11. We achieve an overall accuracy of 99.89% and overall mean F-1 score of 0.9965 across five classes as listed in Table 11 as compared against the DCNN predicted map. We achieve a precision value of 0.9981 , recall value of 0.9972 , and F1-score of 0.9976 in seagrass regions, and high agreement between HR-Net and DCNN results. We also visualize the difference in detected seagrass regions by DCNN and by the HRNet model using a color map where we use green to show agreement between DCNN and HR-Net, red to represent regions indicated by HR-Net as seagrass but not by DCNN, white to represent regions indicated as non-seagrass by HR-Net but seagrass by DCNN. The HR-Net model

shows a good training performance by reproducing the map generated by the DCNN model.

We perform UDA in the temporal images as shown in Fig. 21, Fig. 22 and Fig. 23. Our proposed model uses the learned weights from the source domain to perform adaptation in each individual future temporal image. We compare the performances of the UDA method in each temporal image with the DCNN model. For each individual temporal image, we compare the performance between UDA and DCNN using overall accuracy (OA), overall mean F-1 score (OM F-1), seagrass region's precision, recall, F-1 score and number of pixels detected as the seagrass area, and results are shown in Table 12. We also visualize the difference in seagrass regions between the DCNN and the UDA models using the similar color map described in the previous paragraph. We find that the UDA method performs the best in 26 October 13 with high agreement with the DCNN model. The UDA model achieves OA of 96.87%, OM F-1 0.9450, precision of 0.9771, recall of 0.9114, and F-1 score of 0.9431 in the seagrass regions compared with DCNN in the 26 October 2013 image. The UDA model achieves superb performance on the target 26 October 2013 image as shown in Fig. 21a because the domain difference between the source and target is small. The model trained in the source domain (10 October 2013) without adaptation (WA) also performs comparably in the target domain 26 October 2013 image due to the small domain difference between them. In contrast, when we directly apply the source model to the target image of 14 November 2014 in Fig. 21b, it fails to identify classes correctly as the domain difference becomes larger, but the proposed UDA approach achieves superb performance on the 14 November 2014 image without using any labeled samples. We achieve similar performances with all other temporal images using the proposed UDA model. If the without adaptation

(WA) model fails, the proposed UDA model achieves superb performances as shown in Fig. 21, Fig. 22, and Fig. 23. The performance metrics in Table 12 confirm the superb seagrass detection performance using the proposed UDA model for the temporal images. We achieve good overall accuracy and precision, recall, F-1 score for the seagrass class in Table 12 if the domain difference is small. The performance degrades over time due to the changes of properties in the image because of the seasonal variation shown in Table 12. We achieve the worst performance for the 14 March 2018 SJB image in Table 12 where the domain difference is largest. We also compared the proposed model using U-Net architecture as a backbone. The U-Net architecture performs similarly as HR-Net on the source image in the supervised learning, but it performs worse in the unsupervised domain adaptation learning as compared to HR-Net architecture.

In summary, we achieve superb results in time-series Landsat-8 images at Saint Joseph's Bay location using the proposed unsupervised domain adaptation (UDA) approach for seagrass detection. In the future, we plan to apply this approach to more locations and satellite sensors, e.g. WV-2, Rapid Eye.

CHAPTER 6

BATHYMETRY ESTIMATION USING MULTI-SPECTRAL SATELLITE IMAGE

6.1 Motivation

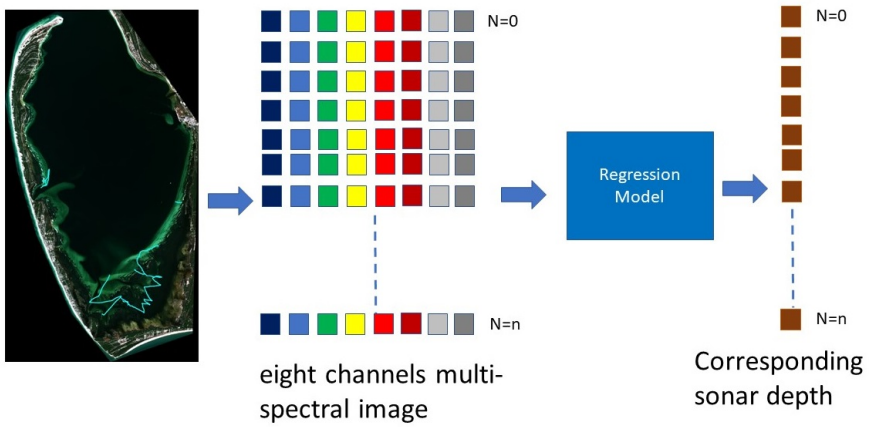
Aquatic bathymetry information abets an important part in marine, oceanographical research, and shipping vehicle routes. The word "bathymetry" represents the depth information in the underwater sea. Marine researchers use bathymetry metrics to identify/classify marine objects in shallow water, such as seagrass, and algae [1]. Bathymetry information is used in a wide range of applications ranging from fossil trace [128], benthic habitat mapping [129, 130], to foraging ecology of a southern elephant [131]. Bathymetry information is also required for sustainable management [57], hydrological modeling [69], monitoring sea-level rise due to climate change, flooding estimation, and sediment removal applications [132, 133]. For determining these objects, researchers need up-to-date bathymetry information of the seafloor.

Bathymetry maps look like topographic maps, but they represent the depth of water in ocean, river, or lakes [134]. Like a topographic map, it connects nearer depth points together and can represent a different region of the ocean, e.g.: ocean trench, seamount, or underwater mountain. In ancient times, bathymetry was measured using a heavy rope, which was thrown from the ship's surface. The rope's length was recorded after reaching the sea surface [134]. The old bathymetry measurement method acquires a single-point

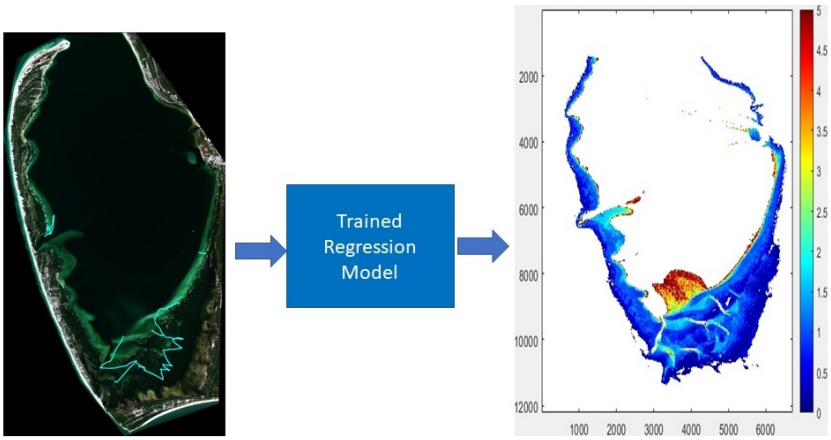
measurement at a time. Getting a clear idea of the underwater bathymetry, requires a lot of manpower, time, and many assessments [134]. Inaccurate and obsolete bathymetry estimation created a lot of danger to ship vehicles and crews in the past [134].

In recent years, sonar (sound navigation ranging) methods replaced the ancient rope bathymetry estimation system. An echo sounder sends out a sound pulse from the ship's surface which is bounced back from the sea bottom to the ship. The time it takes to travel to the sea bottom and back gives the estimate of bathymetry. Because the sonar method estimates bathymetry from a moving ship and records from a different point, when the sonar is bounced back, estimation is inaccurate. The estimated bathymetry may also contain faulty information due to underwater animals such as whales, sharks, and water quality. To remove these drawbacks, researchers used multi-beam echo sounders [135, 136] for performance improvement. This approach sends hundreds of very narrow beams with very high angular resolution. This gives bathymetry information from different points of view. It also removes the error of moving vehicles [134, 137]. In summary, most of the traditional approaches require in situ data measurement and result in a huge time delay to estimate bathymetry with a large labor cost.

To solve the inefficiency and time-consuming issues associated with the previous approaches, we propose an automated machine learning-based bathymetry estimation approach in this dissertation. The proposed bathymetry estimation model uses a multi-spectral satellite image to accurately predict bathymetry information. Our proposed bathymetry estimation model does not require visiting the site for data collection and can instantly estimate bathymetry without any human resource and time delay. Multi-spectral satellite



(a) Training



(b) Testing

Fig. 24: Proposed bathymetry estimation model using machine learning regressor: (a) Training phase (b) Testing phase. The collected sonar measurement tracks are shown in the SJB images in cyan color.

images have been used for bathymetry estimation in recent studies [67, 69].

6.2 Methods

6.2.1 Proposed Bathymetry Estimation Model

Our proposed bathymetry estimation model is shown in Fig. 24. The bathymetry estimation model learns the non-linear relationship between multi-spectral image data X_{train} with the corresponding sonar bathymetry value Y_{train} using a machine learning-based regression model during the training phase as shown in Fig. 24a. We select gradient-boosting as the regression model for bathymetry estimation after comparing with wide ranges of machine learning models, e.g: linear regression, random forest, gradient-boosting, ada-boost, support vector machine (SVM), deep convolutional neural network (DCNN), and deep transfer learning. We apply the trained bathymetry regression model to multi-spectral image data X_{test} in shallow regions to estimate the bathymetry information $Y_{predicted}$ as shown in Fig. 24b.

6.2.2 Linear Regression (LR)

The linear regression algorithm finds a linear relationship between two variables. The linear regression model tries to predict values of a dependent bathymetry variable Y based on independent multi-spectral reflectance data X. For the dataset $D = \{x^i_{train}, y^i_{train}\}^N_{i=1}$, where $i = 1, 2...N$, we can define a linear regression model to fit the data as,

$$y^i_{predicted} = c + m * x^i_{train} \tag{25}$$

This is a straight line equation where c is the intercept point and m is the slope. The linear regression method tries to find the optimal values for m and c that best defines the relationship between bathymetry and image data. It finds the best fit line by minimizing the cost:

$$S = \frac{1}{2N} \sum_{i=1}^{N} (y^i_{predicted} - y^i_{train})^2 \tag{26}$$

We use the scikit-learn [138] library to implement this model.

6.2.3 Support Vector Machines (SVM)

Support vector machine (SVM) was first presented by Vapnik *et al.* [139], which tries to find an optimal hyper-plane between two classes so that it can separate them with the largest margin. The SVM model uses linear, polynomial, Gaussian, Gaussian radial basis function (RBF) and sigmoid kernels to find the optimal hyper-plane. SVM is typically used for classification, but it also has been extended to handle regression problems.

We use two implementations for SVM in our study for performance comparison. The first one is a fast implementation named as $Linear - SVM$ from Scikit-learn library [138], and the second one is a popular SVM implementation known as $Lib - SVM$ [140]. In $Lib - SVM$, we use a radial basis function (RBF) kernel with $C = 1.0$ and $\epsilon = 0.2$. This model is relatively slower than the $Linear - SVM$ model. We also use the Scikit-learn [138] implementation for the $Lib - SVM$ model.

6.2.4 Random Forest (RF)

Random forest is an ensemble machine learning method which consists of many simple decision trees [141]. Decision trees separate data in feature space to identify different classes. Random forest uses the large number of decision trees in an ensemble manner. Each individual decision tree gives its prediction of the data, and random forest considers the decision from all the decision trees using the majority voting scheme. If one of the decision trees predicts an incorrect output then other decision trees can compensate for it. As a result, this method gives better performance with a lower variance. We utilized 100

trees with a maximum depth of 2 in the RF model for bathymetry estimation. We also utilized the mean square error (MSE) as performance metric. Random forest calculates the importance of features using the gini impurity/index. A large gini-index indicates that the feature is important. We used the scikit-learn [138] library to implement this model.

6.2.5 Ada-boost

Ada-boost is an adaptive ensemble machine learning method. It improves the performance iteratively by changing importance weights of training data points based on previous steps, with incorrectly classified data points in previous steps receiving larger weights. Ada-boost algorithm was introduced in 1997 by Freund *et al.* [142]. Ada-boost methods consist of a set of weak learners, and each of them performs slightly better than random guessing. The model combines these 'weak' predictions to form a strong final prediction. Initially, the combining weights are randomly initialized. After a few iterations, the model changes the weights adaptively to improve overall performance. If the model performs poorly on any training samples, it then gives larger weights to those samples in the next step. The final model is the combination of all previous models which produces a superb overall performance. We used 50 estimators, learning rate as 1.0, and linear loss hyper-parameters in the Ada-boost model [138] .

Dataset $D = \{x_i, y_i\}_{i=1}^{n}$ initialized with a same weights $w_i = \frac{1}{N}$. For iteration $t = 1, 2...T$, we repeat the following steps:

1. Train the weak classifier with weights w_i

2. Get weak hypothesis h_t with error $\epsilon_t = \sum_{i:h_t \neq y_i} W_t(x_i)$

3. Choose $\alpha_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t}$

4. Update: $w_{t+1}(i) = \frac{W_t(i)}{Z_t} = \{ \begin{smallmatrix} e^{-\alpha_t} & if\ instance\ i\ is\ correctly\ classified \\ e^{\alpha_t} & if\ instance\ i\ is\ not\ correctly\ classified \end{smallmatrix}$, where Z_t is a normalized factor (chosen so that $\sum_{i=1}^{m} D_{t+1} = 1$,)

Output the final model: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$

6.2.6 Gradient Boosting

We also used the gradient boosting (GB) method for bathymetry estimation which uses an ensemble of prediction models. This method trains many models in a gradual and sequential manner. It uses the loss function to evaluate the training performance of the learner with current coefficients. It identifies the initial weaker model based on the loss. The initial model may perform poorly on the training data. It uses an additive function in each iteration to reduce the residual error and improves performance. The final model uses an ensemble of previous models with a weight to predict the output. This model can be used for both classification and regression tasks.

Given a training data set $D = \{x_i, y_i\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$ and a number of iterations of M, the GB algorithm can be described as,

1. initialize the model with constant weight value,

$$F_o(x) = argmin_\lambda \sum_{i=1}^{n} L(y_i, \lambda) \tag{27}$$

2. From m=1 to M:

   (a) Compute pseudo residual

$$r_{im} = -[\frac{\delta L(y_i, F(x_i)}{\delta F(x_i)}]_{F(x)=F_{m-1}(x)} for, i = 1, .., n \tag{28}$$

(b) Fit a base learner $h_m(x)$ to pseudo residual. Then, train it using the training dataset $(x_i, r_{im})_{i=1}^{n}$,

(c) Compute multiplier λ_m by solving the following equations

$$\lambda_m = argmin_\lambda \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \lambda h_m(x_i)) \tag{29}$$

(d) Update the model

$$F_m(x) = F_{m-1}(x) + \lambda_m h_m(x) \tag{30}$$

3. Output, ensemble models $F_M(x)$

We used the sklearn library to implement the GB model with a learning rate as 0.1, number of estimators as 100, Friedman mean-square error, and a validation fraction of 0.1 [143].

6.2.7 Deep Convolutional Neural Network

Deep convolutional neural network (DCNN) is a popular machine learning method for computer vision tasks. DCNN-based models achieved state-of-the-art results in image classification [78], image segmentation [15], and object detection. Previously, we used a DCNN model for seagrass detection [21, 23]. We use a similar architecture for bathymetry quantification as shown in Fig. 25. The proposed DCNN regression model contains six layers: two convolutional layers, two dropout layers, one flatten layer, and one regression layer as shown in Fig. 25. The first and second convolutional layers have 32 convolutional kernels with a size of 1-by-1 and 16 convolutional kernels with a size of 1-by-1, respectively. After each convolutional layer, we utilize the dropout technique with a probability of 0.01

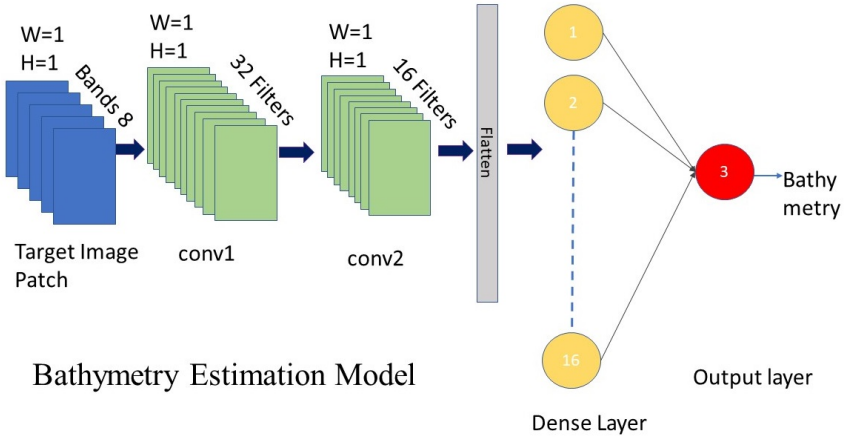
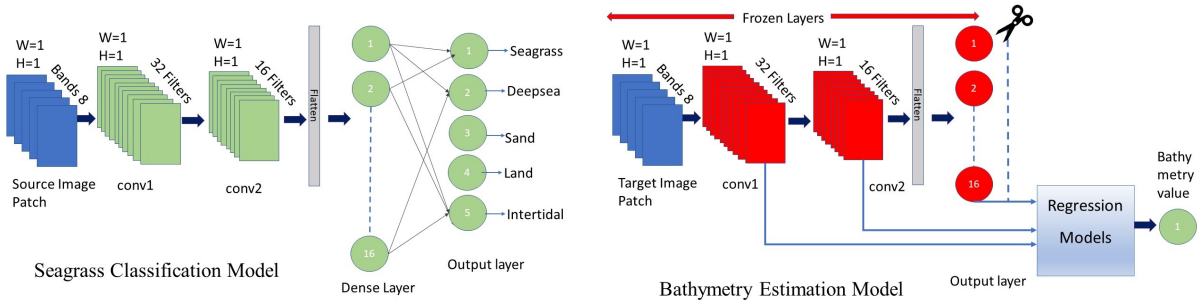


Fig. 25: Bathymetry model using deep convolutional neural network (DCNN).

to reduce over-fitting. We use a flatten layer to convert feature map into 16 hidden vector features. In the last layer, we use a fully connected layer to predict the bathymetry values. We use $1 * 1 * 8$ image patches as inputs for DCNN to predict bathymetry information.

6.2.8 Deep Model with Transfer Learning

It is possible to collect a small number of sonar data from these locations whereas a deep learning model requires many labeled data for training. Transfer learning algorithms can be used when we do not have enough labeled data to train a deep learning model from scratch. We use a model parameter sharing-based transfer learning approach for bathymetry estimation. We first train a DCNN model for the seagrass detection as a source model. Then, we utilized the learned weights from the source model to perform transfer learning in the target model.



(a) Source Model (b) Target Model

Fig. 26: Proposed transfer learning model for bathymetry estimation: (a) Source Model (b)Target Model.

6.2.8.1 Source Model

Our source DCNN model contains two convolutional layers with 32 and 16 $1 * 1$ convolutional kernels, respectively. Each convolutional layer is followed by a dropout layer with 0.01 dropout probability. The flatten layer converts features into a 16 dimension feature vector for classification as shown in Figure 26a. We utilize the same defined regions described in [21, 23] to train the DCNN model for seagrass detection. There are five classes in the image: seagrass, deepsea, sand, land and intertidal. We use a $1 * 1 * 8$ patch size to extract training samples for these five classes. We train three separate source models for three locations: SJB, KB and SGS.

6.2.8.2 Target Model

To perform transfer learning, we use the learned source model weights as the feature extractor. First, we freeze all layers before the $softmax$ layer in the source model as shown in Figure 26a. Then, we extract features from the flatten layer and use the extracted features

to train a linear regression model as described in Section 6.2.2. We repeat the same steps by replacing linear regression with random forest, gradient-boosting methods described in Section 6.2.4 and 6.2.6, respectively. We compare performances of these three methods through a 3-fold cross-validation (CV) evaluation method. We also repeat the same transfer learning procedure for the three methods (LR, RF, GB) using features extracted from the first and the second convolutional layers in the source model.

6.2.9 Log-Linear algorithm

As a baseline model, we use the log-linear algorithm between the multi-spectral image data and depth/bathymetry data [26, 48, 68, 144]. This algorithm assumes that there is a log-linear relationship between multi-spectral reflectance data and the bathymetry values. Geyman *et. al* used a multiple linear regression method to predict bathymetry value Y using reflectance values X. We define the log-linear model as follows:

$$Y = \sum_{i=1}^{N} m_i * log(X_i) + c_i \tag{31}$$

The multi-spectral reflectance value X has N bands. After taking logarithm of image reflectance, we estimate the linear relationship using the linear regression package of the scikit-learn toolbox [138].

6.2.10 Log-Ratio algorithm

The log-linear model usually does not consider the properties of sea bottom type: whether image reflectances are from grass or sand region. The log-ratio-based model can better estimate bathymetry values irrespective of bottom reflectances [48,68]. We define X_i

and X_j as reflectance values of band i and band j, respectively. We define a multiple linear regression method with the bathymetry value Y using the following equation:

$$Y = \sum_{i=1}^{N} \sum_{j=1}^{N} m_{ij} * \frac{log(X_i)}{log(X_j)} + c_{ij} \tag{32}$$

where $i \neq j$ and N is the number of band of multi-spectral image X. For eight-band multi-spectral images, we extracted 56 features in total. Then we use the scikit-learn [138] toolbox based linear regression algorithm to evaluate the performance.

6.2.11 Evaluation Metrics

We utilized root mean square error (RMSE), mean absolute error (MAE), and coefficient of determination R^2 metrics to evaluate the performances of different regression approaches. The R^2 metric can explain how much an independent variable depends on the dependent variables. The perfect value for the R^2 metrics is 1.0 while the RMSE and MAE are non-negative and the smaller the better.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Y_{true} - Y_{pred})^2}{N}} \tag{33}$$

$$MAE = \frac{\sum_{i=1}^{N} |Y_{true} - Y_{pred}|}{N} \tag{34}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(Y_{true} - Y_{pred})^2}{\sum_{i=1}^{N}(Y_{true} - Y_{pred}^{-})^2} \tag{35}$$

where Y_{true} is a labeled sonar measurement, Y_{pred} is model predicted bathymetry value, Y_{pred}^{-} is average predicted bathymetry value and N represents number of data sample.

6.3 Experiment Setup

6.3.1 Multi-spectral WorldView-2 Images

We utilized three multi-spectral satellite images captured by the Worldview-2 optical sensor in the coastal areas of Florida: Saint Joseph Bay (SJB), Keaton Beach (KB), and Saint George Sound (SGS). These images have eight bands: coastal-blue, blue, green, yellow, red, red-edge, near infrared (NIR)-1, and NIR-2 bands with a spatial resolution of $2m$. The wavelengths of Worldview-2 satellite sensor are between $400nm$ to $1040nm$. We performed atmospheric correction for these three images using in-situ data collected from 22 stations as mentioned in Section 3.3.1.

6.3.2 Sonar Bathymetry Measurement

We utilized in-situ multi-beam sonar bathymetry data and their corresponding multi-spectral images to train the bathymetry estimation models. We collected $1,635$ sonar bathymetry measurements from Saint Joseph Bay as shown in Fig. 24a and trained and evaluated these machine learning models using the collected measurements through 3-fold CV. Similarly, we collected $13,096$ and $9,189$ sonar bathymetry measurements from Keaton Beach and Saint George Sound locations, respectively. We applied $2.2m$ and $0.37m$ tidal correction to the measurements from Keaton Beach and Saint George Sound, respectively, to match the tidal hide of the multi-spectral images. These models were trained and validated similarly as for Saint Joseph Bay.

6.4 Results

6.4.1 Patch size determination

To find a suitable patch size for bathymetry estimation, we compared cross validation (CV) results at Saint Joseph Bay (SJB) with different numbers of patch size of $1*1*8, 3*3*8$ and $5*5*8$, and results are listed in Table 13. If we increase the patch size from $1*1*8$ to $3*3*8$ and $5*5*8$, the performance does not improve by a large margin. Therefore, we select $1*1*8$ patch size to design bathymetry regressor models.

TABLE 13: Performance Comparison for different patch sizes at Saint Joseph Bay using RF, ada-boost, GB and linear SVM based on MAE.

patch size	RF	Ada-boost	GB	Linear svm
1x1x8	0.4171	0.3771	**0.2636**	0.6970
3x3x8	0.3911	0.3618	**0.2534**	0.6940
5x5x8	0.3928	0.3634	**0.2501**	0.6838

6.4.2 Cross Validation (CV) Results

To identify the best regression model for bathymetry estimation, the 3-fold CV results of MAE, RMSE, and R^2 are listed in Table 14. We used the following procedure to perform the 3-fold CV:

- Extract sonar bathymetry data and the corresponding multi-spectral image data. Label image information as independent variable X and sonar bathymetry data as dependent variable Y.
- Divide image data X and Y into 3 parts and perform 3-fold CV.

- Compute RMSE, MAE and R^2 for LR, linear-SVM, lib-SVM, RF, ada-boost, and DCNN.

- Repeat the same procedure for KB and SGS locations.

TABLE 14: Three-fold CV results at SJB, KB, and SGS by GB, RF, ada-boost, Lib-SVM, linear SVM, LR and DCNN models.

Location	Metric	GB	RF	Adaboost	LibSVM	Linear SVM	LR	DCNN
SJB	MAE	**0.26**	0.42	0.37	0.45	0.69	0.65	1.02
SJB	RMSE	**0.41**	0.60	0.50	0.92	1.35	1.03	1.67
SJB	R^2	**0.90**	0.79	0.85	0.62	0.02	0.40	0.00
KB	MAE	**0.22**	0.32	0.30	0.25	0.54	0.32	0.27
KB	RMSE	**0.34**	0.45	0.41	0.38	0.72	0.44	0.41
KB	R^2	**0.76**	0.58	0.67	0.69	0.45	0.59	0.66
SGS	MAE	**0.29**	0.54	0.51	0.39	0.71	0.59	0.45
SGS	RMSE	**0.43**	0.68	0.61	0.55	0.94	0.78	0.69
SGS	R^2	**0.79**	0.48	0.65	0.66	0.10	0.31	0.53

The 3-fold CV results at SJB, KB and SGS locations by GB, RF, ada-boost, Lib-SVM, linear-SVM, LR, and DCNN are listed in Table 14. The GB method performs best as compared with other methods. We evaluated performances of log-linear based-method with baseline LR and other approaches including GB, adboost, lib-SVM, linear-SVM, RF and DCNN and results are listed in Table 15. We repeated the same procedure using the log-ratio based method, and results are listed in Table 16.

TABLE 15: Log-linear model CV results at SJB, KB, and SGS by GB, RF, ada-boost, Lib-SVM , linear-SVM, LR (baseline) and DCNN.

Location	Metric	GB	RF	Adaboost	LibSVM	Linear SVM	LR	DCNN
SJB	MAE	0.27	0.43	0.37	0.44	0.67	0.64	1.06
SJB	RMSE	0.41	0.62	0.50	0.89	1.32	0.99	1.68
SJB	R^2	0.90	0.79	0.87	0.66	0.25	0.44	0.00
KB	MAE	0.23	0.33	0.30	0.26	0.48	0.32	0.56
KB	RMSE	0.34	0.45	0.41	0.39	0.65	0.44	0.76
KB	R^2	0.76	0.59	0.67	0.71	0.59	0.61	0.01
SGS	MAE	0.29	0.53	0.51	0.39	0.64	0.57	0.63
SGS	RMSE	0.43	0.68	0.62	0.55	0.86	0.75	0.90
SGS	R^2	0.79	0.48	0.64	0.67	0.24	0.37	0.35

We also use a scatter plot to compare performances by different methods visually as shown in Fig. 27. The scatter plot of the gradient-boosting regression method at KB and SGS locations is shown in Fig. 28. To avoid dense scatter-plotting, we use an average value-based scatter plotting as shown in Fig. 29. We divide all available sonar bathymetry data into bins with a range of $0.01m$. Then, we group all the sonar bathymetry and model predicted bathymetry belonging to a particular bin and calculate the average values for the grouped sonar bathymetry and model predicted bathymetry. We repeat this procedure to estimate the average sonar bathymetry and model predicted bathymetry values for all bins. We show scatter plotting of the average values of sonar and model predicted bathymetry at

TABLE 16: Log-Ratio model (56 features) CV results at SJB, KB, and SGS by GB, RF, ada-boost, Lib-SVM, linear-SVM, LR (baseline) and DCNN.

Location	Metric	GB	RF	Adaboost	LibSVM	Linear SVM	LR	DCNN
SJB	MAE	**0.25**	0.40	0.32	0.44	0.59	0.40	1.09
SJB	RMSE	**0.38**	0.55	0.45	0.90	1.24	0.62	1.70
SJB	R^2	**0.92**	0.82	0.88	0.64	0.26	0.77	0.00
KB	MAE	**0.22**	0.30	0.30	0.26	0.34	0.26	0.47
KB	RMSE	**0.34**	0.42	0.40	0.39	0.50	0.59	0.67
KB	R^2	**0.77**	0.64	0.69	0.71	0.59	0.45	0.22
SGS	MAE	**0.28**	0.45	0.51	0.39	0.58	0.40	0.69
SGS	RMSE	**0.41**	0.60	0.61	0.56	0.83	0.59	0.99
SGS	R^2	**0.81**	0.59	0.69	0.66	0.25	0.62	0.24

the three locations as shown in Fig. 29.

6.4.3 Transfer Learning Results

In transfer learning, we extract features in the first convolution layer, the second convolution layer, and flatten layer of source model, and use GB, RF and LR regression models with the extracted features to predict bathymetry information. The 3-fold CV results are shown in Table 17.

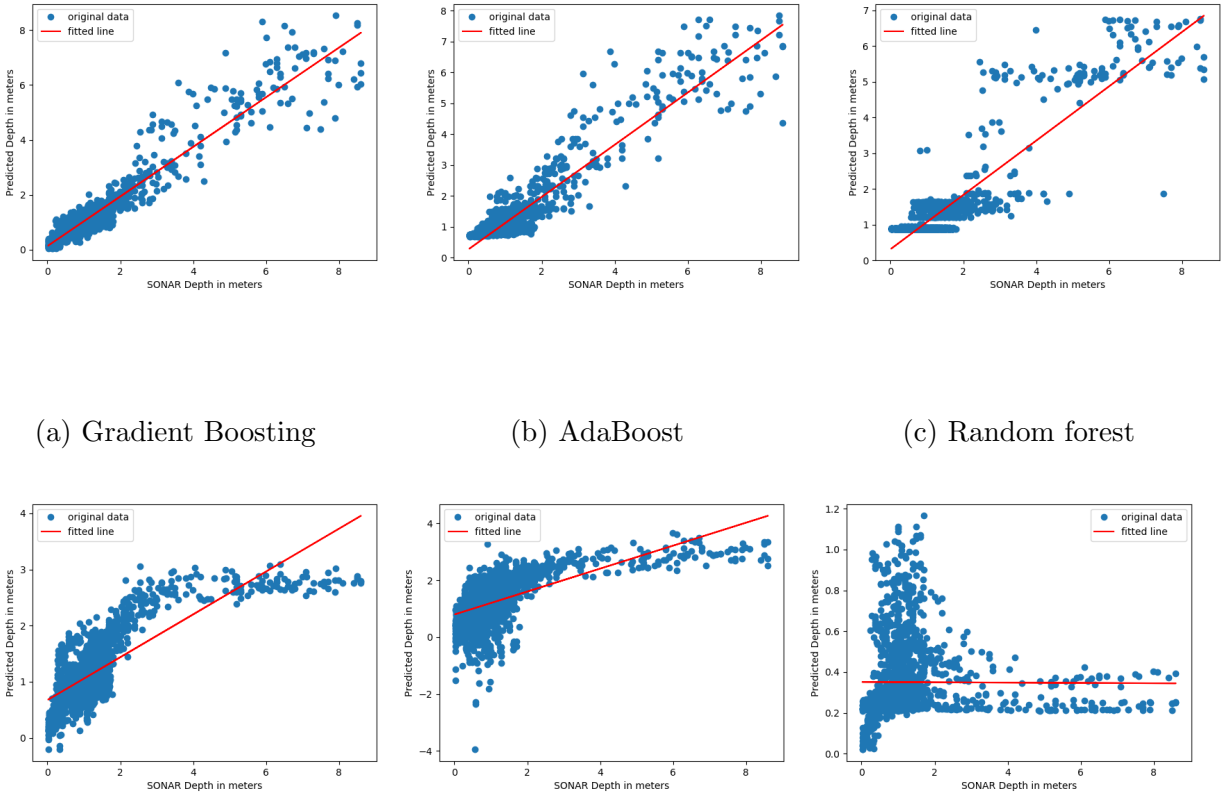
6.4.4 Regression Performance

TABLE 17: CV results of Transfer learning (TL) at SJB, KB, and SGS by GB, RF and LR. Loc: Location, C1: the 1st convolution layer, C2: the 2nd convolution layer, f: flatten

Loc	Metric	GB-f	GB-C2	GB-C1	RF-f	RF-C2	RF-C1	LR-f	LR-C2	LR-C1
SJB	MAE	0.27	0.26	**0.25**	0.39	0.40	0.39	0.47	0.47	0.44
SJB	RMSE	0.46	0.43	**0.40**	0.57	0.59	0.54	0.75	0.77	0.68
SJB	R^2	0.87	0.88	**0.90**	0.81	0.79	0.83	0.67	0.66	0.72
KB	MAE	0.26	0.26	**0.25**	0.31	0.30	0.31	0.31	0.31	0.32
KB	RMSE	0.38	0.38	**0.37**	0.42	0.42	0.42	0.44	0.42	0.43
KB	R^2	0.69	0.69	**0.71**	0.62	0.62	0.62	0.59	0.62	0.61
SGS	MAE	0.32	0.32	**0.29**	0.52	0.53	0.47	0.45	0.45	0.42
SGS	RMSE	0.45	0.45	**0.42**	0.69	0.69	0.61	0.59	0.60	0.56
SGS	R^2	0.76	0.76	**0.79**	0.46	0.45	0.56	0.59	0.59	0.62

After selecting the regression model based on 3-fold CV performance, we estimate bathymetry on three locations using the following procedure:

- Train the selected regression model using all available sonar bathymetry data and corresponding multi-spectral image data at SJB.
- Apply the trained model to predict bathymetry at SJB. Compare the predicted bathymetry to historical National Oceanic and Atmospheric Administration (NOAA) bathymetry values.
- Repeat the same procedure at KB and SGS.

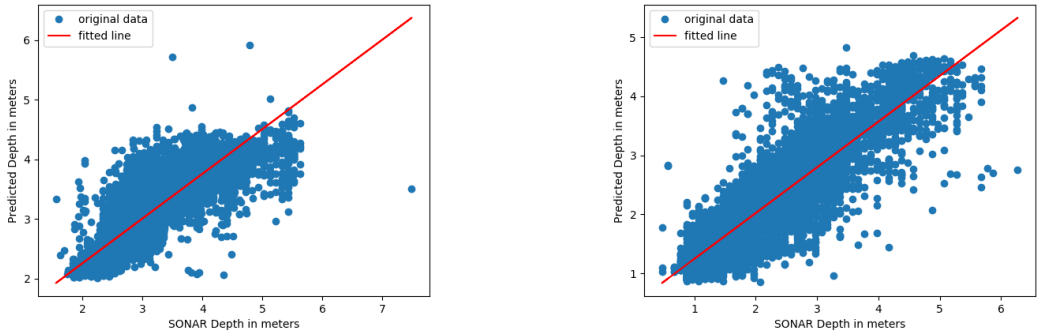


(a) Gradient Boosting (b) AdaBoost (c) Random forest

(d) LIB SVM (e) Linear regression (f) DCNN

Fig. 27: Scatter plotting of 3-fold CV results at Saint Joseph Bay location :(a) Gradient-Boosting (b) AdaBoost (c) Random Forest (d) Lib-SVM (e) Linear Regression and (f) DCNN. X-axis represents sonar bathymetry depth and Y-axis represents machine learning model predicted bathymetry.

The predicted bathymetry maps for SJB, KB, and SGS are shown in Fig. 30, Fig. 31, and Fig. 32, respectively. We use the Matlab color-map 'jet' to show the model predicted bathymetry values and NOAA bathymetry value clipped between 0 to 5. We use the blue to represent 0 bathymetry or lower and red to represent 5 bathymetry value or higher. We

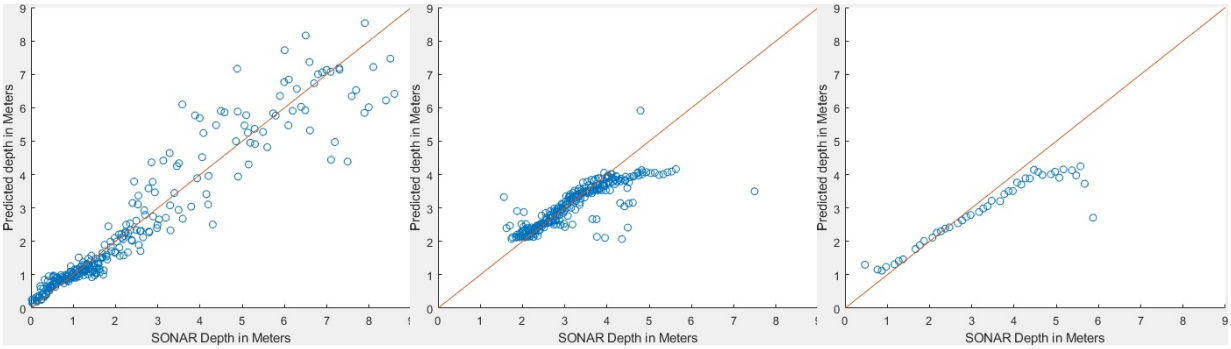


(a) Keaton Beach (b) Saint George Sound

Fig. 28: Scatter plotting of 3-fold CV results by Gradient Boosting at: (a) Keaton Beach and (b) Saint George Sound. X-axis represents the sonar bathymetry depth and Y-axis represents the machine learning model predicted bathymetry.

follow the same procedure for KB and SGS locations, and results are shown in Fig. 31 and Fig. 32, respectively.

To evaluate the bathymetry estimation performance visually, we show the difference between the model-predicted bathymetry and NOAA bathymetry using the 'jet' color-map with values between -1 to 1. Differences of -1 or lower are represented by the blue whereas differences of 1 or higher are represented by the red. We plot the histogram of difference values in Fig. 30d. We separate the bathymetry prediction performance in the seagrass region at SJB in Fig. 30e and corresponding histogram of difference values in Fig. 30f. The satellite imagery usually fails to get the correct bottom reflectance in the areas where seagrasses grow. Therefore, we add a $0.3m$ correction value to the seagrass region at this location. The histogram and difference values at KB and SGS are shown in Fig. 31e, f and Fig. 32d, e. The RMSE values between NOAA and model predicted bathymetry are shown



(a) SJB Average Values (b) KB Average Values (c) SGS Average Values

Fig. 29: Scatter plotting of 3-fold CV results by Gradient Boosting regression at (a) SJB (b) KB and (c) SGS. X-axis represents the sonar bathymetry depth and Y-axis represents the machine learning model predicted bathymetry.

in Table 18 for the three locations.

6.5 Discussion

To find the best method for bathymetry estimation, we compare the 3-fold CV performances of different regression methods as shown in Table 14. We find that the GB method achieved the best performance at all three locations. Linear SVM and DCNN performed worst relative to all other methods at these locations. At Saint Joseph Bay, GB achieved the best RMSE, MAE and R^2 with values of 0.26, 0.40 and 0.90, respectively. Similar performances can be observed at both KB and SGS locations. The DCNN method failed to estimate bathymetry due to lack of training samples for the SJB location. Typically, the deep learning model requires large datasets for training. Scatter plots of bathymetry

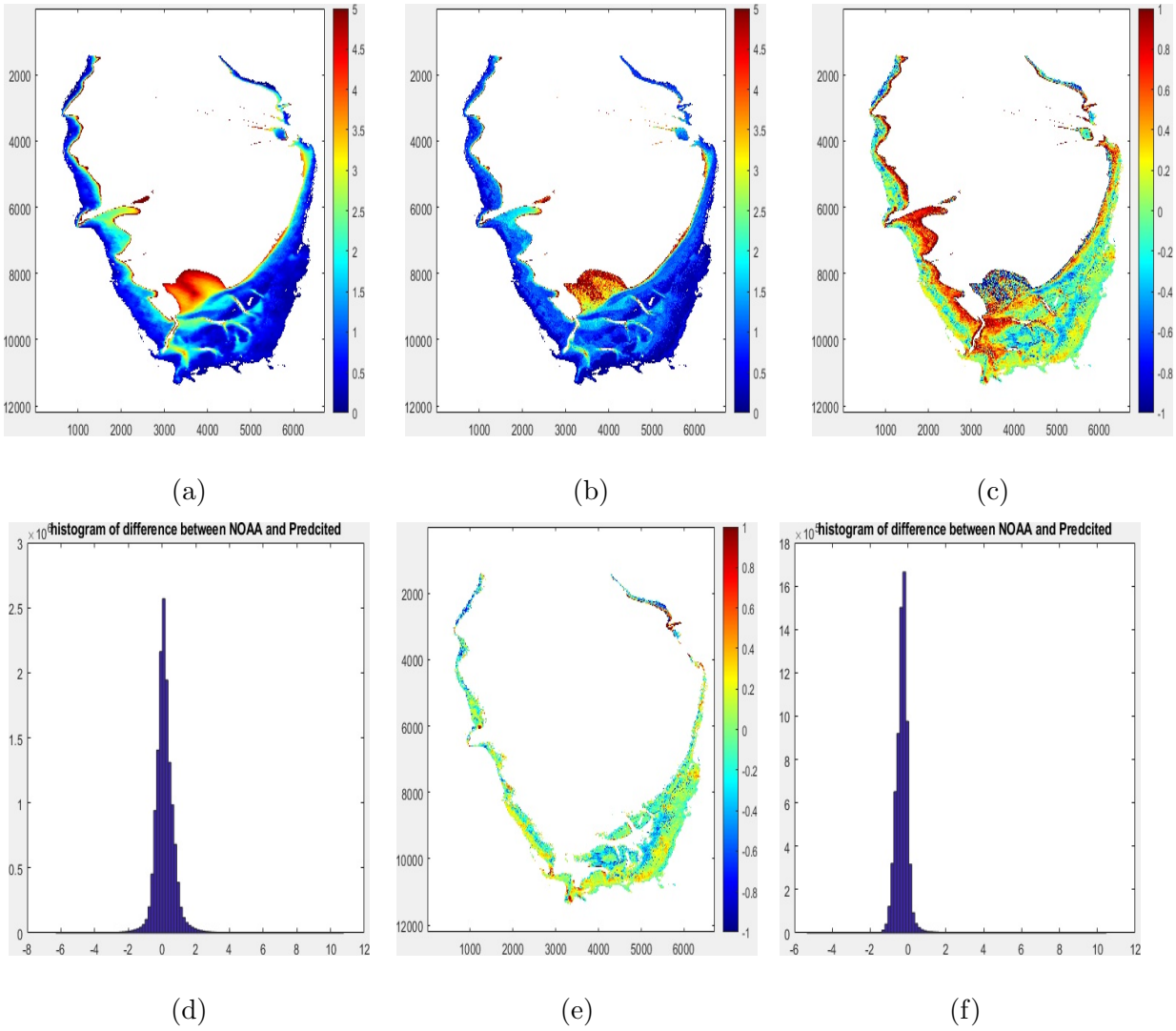


Fig. 30: Bathymetry estimation at SJB: (a) NOAA bathymetry, (b) Model predicted bathymetry, (c) Difference values between model predicted bathymetry and NOAA bathymetry, (d) Histogram of the difference values between model predicted bathymetry and NOAA bathymetry, (e) Difference values between the model predicted bathymetry and NOAA bathymetry in the seagrass regions, and (f) Histogram of the difference values between model predicted bathymetry and NOAA bathymetry in the seagrass regions

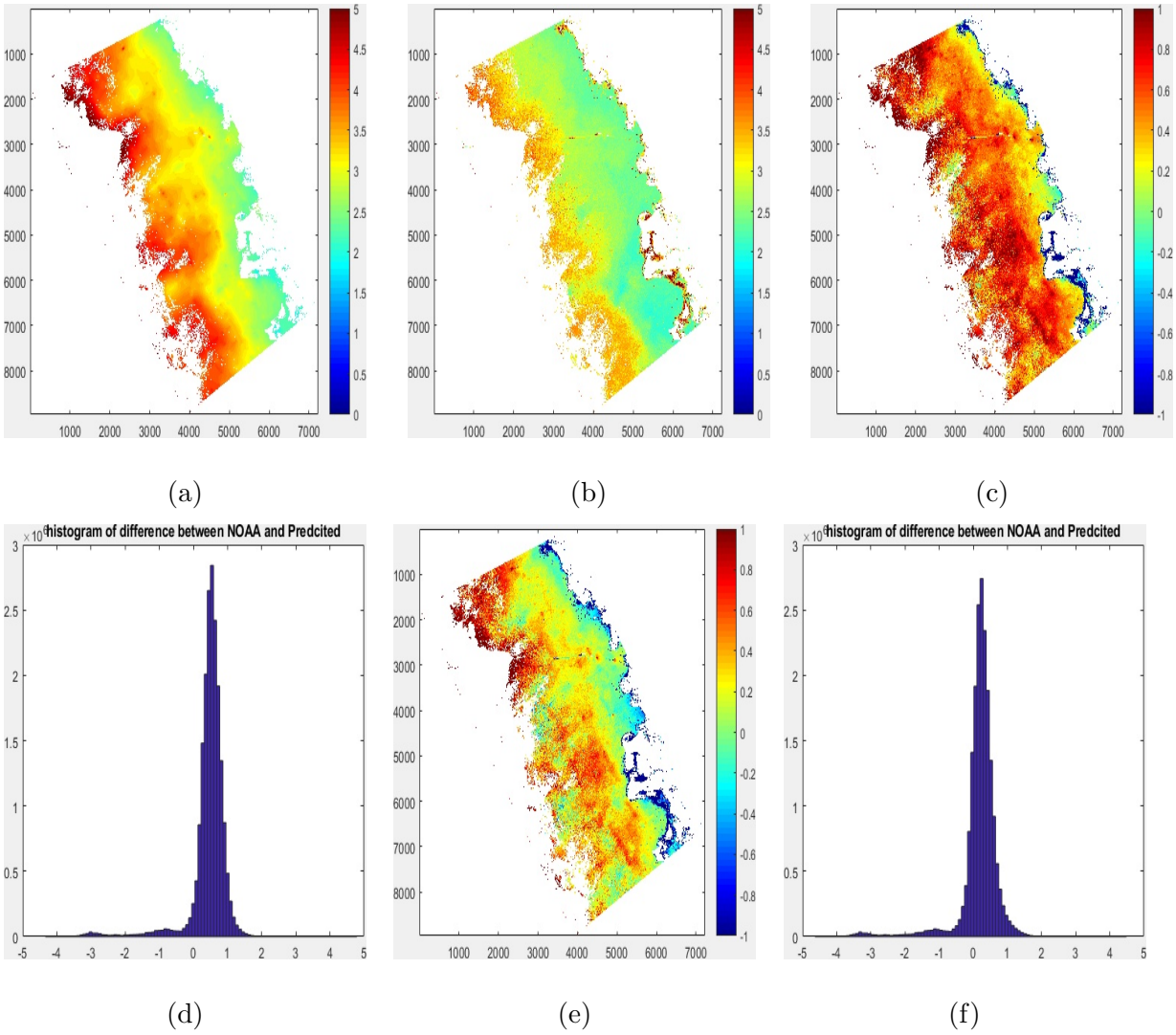


Fig. 31: Bathymetry estimation results at Keaton Beach (a) NOAA bathymetry, (b) Model predicted bathymetry, (c) difference between predicted bathymetry and NOAA bathymetry, (d) Histogram difference between predicted bathymetry and NOAA bathymetry, (e) difference between predicted bathymetry with seagrass correction and NOAA bathymetry, (f) histogram of difference between predicted bathymetry (seagrass corrected) and NOAA bathymetry

TABLE 18: RMSE between NOAA bathymetry and model prediction at SJB, KB, and SGS by GB.

Location	RMSE between NOAA and model predicted bathymetry	
	Without correction	With Seagrass correction
Saint Joseph Bay (SJB)	0.6205	0.6566
Keaton Beach (KB)	0.7107	0.5965
Saint George Sound (SGS)	0.8745	0.9045

estimation at the SJB location by different methods are shown in Fig. 27, which also proved that GB is the best method. Similar performance is found at KB and SGS locations as shown in Fig. 28. Therefore, we conclude that the GB method is the best approach for bathymetry estimation at the three locations under study, because The GB usually performs superbly with minimal feature selection effort and can scale the performance relative to other methods.

We compared baseline log-linear and log-ratio methods with the original multispectral (MS) reflectance based approach, and results are listed in Tables 14, 15 and 16. The original MS reflectance algorithm assumes a linear relationship between MS bands and bathymetry values whereas the other two methods assume log/log-ratio relationship. The performances by the log-ratio (MAE 0.40, RMSE 0.62, R^2 0.77) based linear regression method improved by a large margin as compared to log-linear (MAE 0.64, RMSE 0.99, R^2 0.44) and the original method (MAE 0.65, RMSE 1.03, R^2 0.40) at the SJB location. Similar performance improvements at KB and SGS locations are listed in Tables 14, 15 and

16. Larger performance improvements are observed by the Adaboost, linear-SVM methods whereas RF and lib-SVM have minimal performance improvements, and the performance by DCNN decreased for both log-linear and log-ratio based approaches. GB performed similarly with original reflectance, log-linear and log-ratio based approaches at SJB, KB and SGS locations, proving that GB is a robust model and requires minimal feature engineering. Both log-linear and log-ratio methods use a logarithm value of the MS reflectance whereas our GB-based proposed approach uses the original MS reflectance value and achieves superb performance.

Performances by the deep transfer learning method are shown in Table 17. The first convolution layer's features combined with the GB regression model is the best method. Because there might be domain shift between the source and target tasks that can be reflected by higher layer features, the initial layers learned low-level features more related to the target domain. The TL-based bathymetry estimation model achieved similar or slightly better results than the original multi-spectral based approach as shown in Table 14; however, it requires more computation resource, and the significance of transfer learning is trivial.

Since the GB method is the best model, we apply it to SJB, KB and SGS locations, and results are shown in Fig.s 30, 31, and 32 respectively. To assess the predicted bathymetry, we use the National Oceanic and Atmospheric Administration (NOAA) bathymetry as the reference values. It is worth noting that the NOAA bathymetry can not be considered as a ground truth since it is not updated regularly, and the sea depth has changed over the years. The GB achieved the smallest error in seagrass areas but large errors appeared in deep sand areas in Fig. 30 because NOAA bathymetry may contain more

incorrect measurements in those areas. Also, the sonar measurements are collected mostly from the shallow coastal areas. As a result, the model fails to estimate the deeper regions. In addition, the spectral attenuation is larger in the deep sea, and spectral reflectance may not correctly represent the actual depth. If we compare the histogram of the difference values between NOAA data and the model predicted bathymetry, most of the differences are close to 0, proving that the model estimated bathymetry is highly correlated with NOAA data for the SJB location. As an additional experiment, we add $0.3m$ correction to the seagrass region identified by a trained DCNN model [21], and it does not improve the bathymetry measurement results at the SJB location.

We repeat the same experiment at the KB location, and results are shown in Fig. 31. The difference values (between NOAA data and model-predicted bathymetry) and histogram of the difference values are shown in Fig. 31c and Fig. 31d, respectively. Similarly, we apply a $0.3m$ bathymetry correction to image pixels identified as the seagrass region by the trained DCNN classifier [21]. After the correction, RMSE is reduced and the difference value and its histogram are shown in Fig. 31e and Fig. 31f, respectively. The model predicted bathymetry at the SGS location shows a similar trend as that at the SJB location as shown in Fig. 32.

We show RMSE between NOAA bathymetry and model-predicted bathymetry in Table 18. We add a correction of $0.3m$ to the predicted bathymetry depth in the seagrass areas to correct sea bottom reflectance error. Due to the correction, the RMSE error at the KB location decreases. However, models perform better at SJB and SGS without the correction. We obtain the best RMSE values of 0.6205 and 0.8745 without the seagrass correction at SJB and KB, respectively, as listed in Table 18. We achieve the best RMSE

of 0.5965 at KB with the seagrass area correction as listed in Table 18.

In summary, the proposed GB-based bathymetry estimation model performs superbly at the three locations in Florida. The estimated bathymetry aligns with the NOAA bathymetry in the shallow seagrass regions relative to the sand regions. In the future, we plan to apply this model to more locations to further validate the proposed approach.

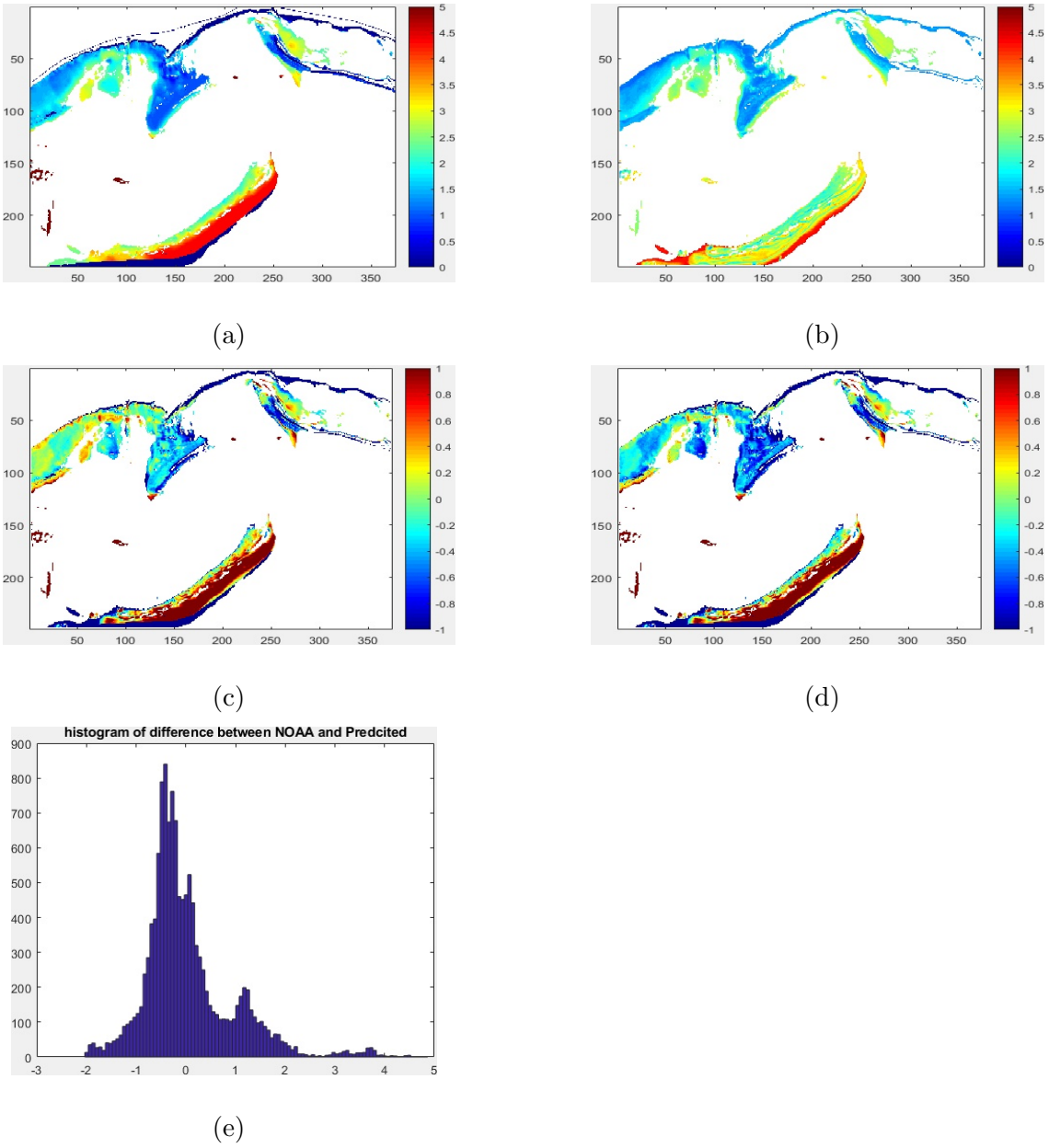


Fig. 32: Bathymetry estimation results at Saint George Sound, (a) NOAA bathymetry, (b) Model predicted bathymetry, (c) Difference between Model predicted bathymetry and NOAA bathymetry, (d) Difference between the model Predicted bathymetry (sea-grass corrected) and NOAA bathymetry, (e) Histogram difference between model predicted bathymetry and NOAA bathymetry after correction.

CHAPTER 7

CONCLUSION

Automatic seagrass detection systems in multispectral images are important tools for seagrass monitoring. This dissertation discusses different aspects of deep learning-based seagrass systems using multi-spectral satellite images. First, I develop a deep learning-based supervised approach using adequate labeled training data. Then, I develop a set of deep learning algorithms to handle the domain shift due to location or temporal change in different settings including few-shot, semi-supervised and unsupervised learnings. I also develop a bathymetry estimation model that will contribute towards the accurate seagrass monitoring system. In this section, I summarize and conclude each chapter based on the results and also develop future work strategies.

Capsule network for seagrass detection: In Chapter 3, I develop a deep capsule network for seagrass detection. To the best of our knowledge, this study represents the first work of designing a capsule network for seagrass detection. The proposed capsule network achieved better classification accuracy than the baseline models (SVM) and comparable results with the deep convolutional neural network (DCNN) in *3*-fold CV. Transfer learning proved to be a good technique to address the problem of model adaptation. In addition, the developed generative model is able to increase the classifier performance by iteratively generating new data from the capsule's features. It obtained accuracies of 99.16% and 99.67% at Keaton Beach and Saint George Sound, respectively. When I only used the generated data, I achieved accuracies of 93.00% and 93.34% at the two new locations,

respectively, proving the similarity between the original samples and generated samples. This dissertation also demonstrated the effectiveness of my developed method through a set of 2D plots that are able to display the capsule features. Since magnitudes of the capsule features determine probabilities of classes, the plots are able to visually assess performance of a trained capsule network in a significantly simple manner. To the best of my knowledge, this study is the first to offer this visualization tool for the evaluation of a capsule network's performance.

Semi-supervised learning for seagrass detection: Labelling atmospherically corrected multispectral images is labor intensive and time consuming. In Chapter 4, I developed a semi-supervised domain adaptation method for deep CNN models for seagrass detection. The proposed model first used unlabelled samples in both domains to adapt the source domain model to the target domain based on the GAN loss. Then it utilized contrastive semantic loss with a few labelled samples from the target domain to further adapt the model. In addition, the source model and target model were jointly optimized in the second step. The proposed model was evaluated in three atmospherically corrected WorldView-2 multispectral images taken in Florida and achieved the best results among 28 out of 36 experimental scenarios. The semi-supervised domain adaptation approach utilized one-tenth labeled samples to achieve the same performance as the transfer learning approach mentioned in the previous paragraph. Future work will evaluate the proposed model across broader regional areas such as the southeastern United States. In future, I plan to further improve the performance to detect seagrass across location using the unsupervised domain adaptation algorithm.

Unsupervised seagrass detection in time-series images: Time-series multi-spectral images are considered an effective data modality for monitoring seagrass over the years which typically requires data labeling by human experts. In Chapter 5, I propose an unsupervised domain adaptation approach (UDA) to remove this constraint. In the source domain, I use a pixel-wise image segmentation method to detect seagrass pixels so that it learns the relationship among neighboring pixels. The proposed unsupervised domain adaptation algorithm achieved superb results in the Landsat-8 multi-spectral time-series images. I plan to further improve the model to tackle location-shift.

Bathymetry estimation: Bathymetry plays an important role in detecting marine species in the underwater environment. However, bathymetry measurements are obsolete or absent in many regions. In Chapter 6, I propose a gradient boosting model for bathymetry estimation from multi-spectral images. The proposed model automatically learns the non-linear relationship between sonar measurement and its corresponding multi-spectral satellite image information. The gradient boosting model performs best compared with other competing machine learning model and is able to effectively quantify bathymetry information with RMSE of 0.6205, 0.5965 and 0.8745 against the NOAA bathymetry at Saint Joseph's bay, Keaton beach and Saint George Sound locations, respectively.

Although the developed deep learning algorithms are specifically designed for seagrass detection, they can easily be applied to other domains. We usually need several thousand to millions of labeled samples from a domain to train a deep learning model. In reality, it is not always possible to label such a large amount of data for every new domain. My developed seagrass detection model re-uses a well trained deep learning model from one

domain to detect seagrass in the target domain. I developed these algorithms based on the target data availability. Researchers can choose few-shot learning, semi-supervised learning, or an unsupervised learning approach based on the labeled or unlabeled data availability for their target domain. The proposed algorithms achieved superb results in the target location or temporal images with minimal human supervision.

In this dissertation, I developed a set of deep learning methods while developing algorithms for seagrass detection. In Chapter 3, I developed a novel data-augmentation method from the capsule's features and offered a new visualization tool for the evaluation of the capsule network's performance. In Chapter 4, I developed a semi-supervised domain adaptation method where I utilized GAN-based loss to match marginal distribution using the unlabeled samples and classification contrastive semantic alignment loss to match class-specific distribution using the labeled samples. This semi-supervised domain adaptation method achieved superb results as compared with traditional transfer learning approaches. In the unsupervised domain adaptation (UDA) method described in Chapter 5, I developed a semantic segmentation based UDA approach where we did not use any labeled samples. These developed algorithms can be used to handle domain shift in similar settings for deep learning-based applications.

REFERENCES

[1] V. J. Hill, R. C. Zimmerman, W. P. Bissett, H. Dierssen, and D. D. Kohler, "Evaluating light availability, seagrass biomass, and productivity using hyperspectral airborne remote sensing in saint josephâs bay, florida," *Estuaries and coasts*, vol. 37, no. 6, pp. 1467–1489, 2014.

[2] "myfwc: Florida fish and wildlife conservation commission." `http://myfwc.com/research/habitat/seagrasses/information/importance/`. (Date last accessed October 20, 2017).

[3] "floridadep: Florida coastal office." `https://floridadep.gov/fco`. (Date last accessed October 20, 2017).

[4] R. Li, J.-K. Liu, A. Sukcharoenpong, J. Yuan, H. Zhu, and S. Zhang, "A systematic approach toward detection of seagrass patches from hyperspectral imagery," *Marine Geodesy*, vol. 35, no. 3, pp. 271–286, 2012.

[5] D. Yang and C. Yang, "Detection of seagrass distribution changes from 1991 to 2006 in xincun bay, hainan, with satellite remote sensing," *Sensors*, vol. 9, no. 2, pp. 830–844, 2009.

[6] J. W. Fourqurean, C. M. Duarte, H. Kennedy, N. Marbà, M. Holmer, M. A. Mateo, E. T. Apostolaki, G. A. Kendrick, D. Krause-Jensen, K. J. McGlathery, *et al.*, "Seagrass ecosystems as a globally significant carbon stock," *Nature geoscience*, vol. 5, no. 7, pp. 505–509, 2012.

[7] L. C. Cullen-Unsworth, B. L. Jones, R. Lilley, and R. K. Unsworth, "Secret gardens under the sea: What are seagrass meadows and why are they important?,"

[8] M. Moniruzzaman, S. Islam, P. Lavery, M. Bennamoun, and C. P. Lam, "Imaging and classification techniques for seagrass mapping and monitoring: A comprehensive survey," *arXiv preprint arXiv:1902.11114*, 2019.

[9] C. M. Duarte, I. J. Losada, I. E. Hendriks, I. Mazarrasa, and N. Marbà, "The role of coastal plant communities for climate change mitigation and adaptation," *Nature Climate Change*, vol. 3, no. 11, pp. 961–968, 2013.

[10] R. C. Zimmerman, "Scaling up: Predicting the impacts of climate change on seagrass ecosystems," *Estuaries and Coasts*, vol. 44, no. 2, pp. 558–576, 2021.

[11] K. Topouzelis, D. Makri, N. Stoupas, A. Papakonstantinou, and S. Katsanevakis, "Seagrass mapping in greek territorial waters using landsat-8 satellite images," *International journal of applied earth observation and geoinformation*, vol. 67, pp. 98–113, 2018.

[12] D. Traganos and P. Reinartz, "Mapping mediterranean seagrasses with sentinel-2 imagery," *Marine pollution bulletin*, vol. 134, pp. 197–209, 2018.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

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

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

[16] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5693–5703, 2019.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[19] R.-R. Li, Y. J. Kaufman, B.-C. Gao, and C. O. Davis, "Remote sensing of suspended sediments and shallow coastal waters," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 3, pp. 559–566, 2003.

[20] Z. Ling, D. Sun, S. Wang, Z. Qiu, Y. Huan, Z. Mao, and Y. He, "Remote sensing estimation of colored dissolved organic matter (cdom) from goci measurements in the

bohai sea and yellow sea," *Environmental Science and Pollution Research*, vol. 27, no. 7, pp. 6872–6885, 2020.

[21] K. A. Islam, D. Pérez, V. Hill, B. Schaeffer, R. Zimmerman, and J. Li, "Seagrass detection in coastal water through deep capsule networks," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pp. 320–331, Springer, 2018.

[22] D. Pérez, K. Islam, V. Hill, R. Zimmerman, B. Schaeffer, and J. Li, "Deepcoast: Quantifying seagrass distribution in coastal water through deep capsule networks," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pp. 404–416, Springer, 2018.

[23] K. A. Islam, V. Hill, B. Schaeffer, R. Zimmerman, and J. Li, "Semi-supervised adversarial domain adaptation for seagrass detection in multispectral images," in *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 1120–1125, IEEE, 2019.

[24] K. A. Islam, V. Hill, B. Schaeffer, R. Zimmerman, and J. Li, "Semi-supervised adversarial domain adaptation for seagrass detection using multispectral images in coastal areas," *Data Science and Engineering*, vol. 5, pp. 111–125, 2020.

[25] D. Traganos, D. Cerra, and P. Reinartz, "Cubesat-derived detection of seagrasses using planet imagery following unmixing-based denoising: Is small the next big?," *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 42, 2017.

[26] M. Lyons, S. Phinn, and C. Roelfsema, "Integrating quickbird multi-spectral satellite and field data: mapping bathymetry, seagrass cover, seagrass species and change in

moreton bay, australia in 2004 and 2007," *Remote Sensing*, vol. 3, no. 1, pp. 42–64, 2011.

[27] M. D. M. Manessa, A. Kanno, M. Sekine, E. E. Ampou, N. Widagti, and A. R. Assyakur, "Shallow-water benthic identification using multispectral satellite imagery: investigation on the effects of improving noise correction method and spectral cover," *Remote Sensing*, vol. 6, no. 5, pp. 4454–4472, 2014.

[28] V. Pasqualini, C. Pergent-Martini, G. Pergent, M. Agreil, G. Skoufas, L. Sourbes, and A. Tsirika, "Use of spot 5 for mapping seagrasses: An application to posidonia oceanica," *Remote Sensing of Environment*, vol. 94, no. 1, pp. 39–45, 2005.

[29] A. Vela, A. Djellouli, H. Langar, V. Pasqualini, G. Pergent, C. Pergent-Martini, R. Mrabet, and H. Djabou, "Use of spot 5 and ikonos satellites for mapping biocenoses in a tunisian lagoon," 2005.

[30] F. Dahdouh-Guebas, E. Coppejans, and D. Van Speybroeck, "Remote sensing and zonation of seagrasses and algae along the kenyan coast," *Hydrobiologia*, vol. 400, pp. 63–73, 1999.

[31] E. Oguslu, K. Islam, D. Perez, V. Hill, W. Bissett, R. Zimmerman, and J. Li, "Detection of seagrass scars using sparse coding and morphological filter," *Remote Sensing of Environment*, vol. 213, pp. 92–103, 2018.

[32] F. Li and B. Xiao, "Aquatic vegetation mapping based on remote sensing imagery: An application to honghu lake," in *2011 International Conference on Remote Sensing, Environment and Transportation Engineering*, pp. 4832–4836, IEEE, 2011.

[33] E. J. Hochberg, S. Andréfouët, and M. R. Tyler, "Sea surface correction of high spatial resolution ikonos images to improve bottom mapping in near-shore environments," *IEEE transactions on geoscience and remote sensing*, vol. 41, no. 7, pp. 1724–1729, 2003.

[34] D. R. Mishra, S. Narumalani, D. Rundquist, and M. Lawson, "High-resolution ocean color remote sensing of benthic habitats: a case study at the roatan island, honduras," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 7, pp. 1592–1604, 2005.

[35] A. Fornes, G. Basterretxea, A. Orfila, A. Jordi, A. Álvarez, and J. Tintoré, "Mapping posidonia oceanica from ikonos," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 5, pp. 315–322, 2006.

[36] D. Yang, Y. Yang, C. Yang, J. Zhao, and Z. Sun, "Detection of seagrass in optical shallow water with quickbird in the xincun bay, hainan province, china," *IET image processing*, vol. 5, no. 5, pp. 363–368, 2011.

[37] S. Cerdeira-Estrada, T. Heege, M. Kolb, S. Ohlendorf, A. Uribe, A. Müller, R. Garza, R. Ressl, R. Aguirre, I. Mariño, *et al.*, "Benthic habitat and bathymetry mapping of shallow waters in puerto morelos reefs using remote sensing with a physics based data processing," in *2012 IEEE International Geoscience and Remote Sensing Symposium*, pp. 4383–4386, IEEE, 2012.

[38] F. Eugenio, J. Marcello, and J. Martin, "High-resolution maps of bathymetry and benthic habitats in shallow-water environments using multispectral remote sensing

imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 7, pp. 3539–3549, 2015.

[39] J. Marcello, F. Eugenio, F. Marqués, and J. Martín, "Precise classification of coastal benthic habitats using high resolution worldview-2 imagery," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 2307–2310, IEEE, 2015.

[40] V. Lafon, J. Froidefond, F. Lahet, and P. Castaing, "Spot shallow water bathymetry of a moderately turbid tidal inlet based on field measurements," *Remote sensing of Environment*, vol. 81, no. 1, pp. 136–148, 2002.

[41] C. Legleiter, M. Tedesco, L. Smith, A. Behar, and B. Overstreet, "Mapping the bathymetry of supraglacial lakes and streams on the greenland ice sheet using field measurements and high resolution satellite images," 2014.

[42] G. Doxani, M. Papadopoulou, P. Lafazani, C. Pikridas, and M. Tsakiri-Strati, "Shallow-water bathymetry over variable bottom types using multispectral worldview-2 image," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 39, no. 8, pp. 159–164, 2012.

[43] H. Su, H. Liu, L. Wang, A. M. Filippi, W. D. Heyman, and R. A. Beck, "Geographically adaptive inversion model for improving bathymetric retrieval from satellite multispectral imagery," *IEEE transactions on Geoscience and Remote Sensing*, vol. 52, no. 1, pp. 465–476, 2013.

[44] H. Su, H. Liu, and W. D. Heyman, "Automated derivation of bathymetric information from multi-spectral satellite imagery using a non-linear inversion model," *Marine Geodesy*, vol. 31, no. 4, pp. 281–298, 2008.

[45] J. Wei, M. Wang, Z. Lee, H. O. Briceño, X. Yu, L. Jiang, R. Garcia, J. Wang, and K. Luis, "Shallow water bathymetry with multi-spectral satellite ocean color sensors: Leveraging temporal variation in image data," *Remote Sensing of Environment*, vol. 250, p. 112035, 2020.

[46] Y. Ma, N. Xu, Z. Liu, B. Yang, F. Yang, X. H. Wang, and S. Li, "Satellite-derived bathymetry using the icesat-2 lidar and sentinel-2 imagery datasets," *Remote Sensing of Environment*, vol. 250, p. 112047, 2020.

[47] A. Pattanaik, K. Sahu, and M. Bhutiyani, "Estimation of shallow water bathymetry using irs-multispectral imagery of odisha coast, india," *Aquatic Procedia*, vol. 4, pp. 173–181, 2015.

[48] R. P. Stumpf, K. Holderied, and M. Sinclair, "Determination of water depth with high-resolution satellite imagery over variable bottom types," *Limnology and Oceanography*, vol. 48, no. 1part2, pp. 547–556, 2003.

[49] J. Pushparaj and A. V. Hegde, "Estimation of bathymetry along the coast of mangaluru using landsat-8 imagery," *The International Journal of Ocean and Climate Systems*, vol. 8, no. 2, pp. 71–83, 2017.

[50] D. Poursanidis, D. Traganos, N. Chrysoulakis, and P. Reinartz, "Cubesats allow high spatiotemporal estimates of satellite-derived bathymetry," *Remote Sensing*, vol. 11, no. 11, p. 1299, 2019.

[51] A. Pacheco, J. Horta, C. Loureiro, and Ó. Ferreira, "Retrieval of nearshore bathymetry from landsat 8 images: A tool for coastal monitoring in shallow waters," *Remote Sensing of Environment*, vol. 159, pp. 102–116, 2015.

[52] A. Muzirafuti, G. Barreca, A. Crupi, G. Faina, D. Paltrinieri, S. Lanza, and G. Randazzo, "The contribution of multispectral satellite image to shallow water bathymetry mapping on the coast of misano adriatico, italy," *Journal of Marine Science and Engineering*, vol. 8, no. 2, p. 126, 2020.

[53] C. Calkoen, G. Hesselmans, G. Wensink, and J. Vogelzang, "The bathymetry assessment system: efficient depth mapping in shallow seas using radar images," *International Journal of Remote Sensing*, vol. 22, no. 15, pp. 2973–2998, 2001.

[54] D. Poursanidis, D. Traganos, P. Reinartz, and N. Chrysoulakis, "On the use of sentinel-2 for coastal habitat mapping and satellite-derived bathymetry estimation using downscaled coastal aerosol band," *International Journal of Applied Earth Observation and Geoinformation*, vol. 80, pp. 58–70, 2019.

[55] S. Peeri, J. V. Gardner, L. G. Ward, and J. R. Morrison, "The seafloor: A key factor in lidar bottom detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 3, pp. 1150–1157, 2010.

[56] M. S. Andersen, Á. Gergely, Z. Al-Hamdani, F. Steinbacher, L. R. Larsen, and V. B. Ernstsen, "Processing and performance of topobathymetric lidar data for geomorphometric and morphological classification in a high-energy tidal environment.," *Hydrology & Earth System Sciences*, vol. 21, no. 1, 2017.

[57] J. Gao, "Bathymetric mapping by means of remote sensing: methods, accuracy and limitations," *Progress in Physical Geography*, vol. 33, no. 1, pp. 103–116, 2009.

[58] P. J. Kinzel, C. J. Legleiter, and J. M. Nelson, "Mapping river bathymetry with a small footprint green lidar: applications and challenges 1," *JAWRA Journal of the American Water Resources Association*, vol. 49, no. 1, pp. 183–204, 2013.

[59] K. Saylam, J. R. Hupp, J. R. Andrews, A. R. Averett, and A. J. Knudby, "Quantifying airborne lidar bathymetry quality-control measures: A case study in frio river, texas," *Sensors*, vol. 18, no. 12, p. 4153, 2018.

[60] N. Corti Meneses, S. Baier, J. Geist, and T. Schneider, "Evaluation of green-lidar data for mapping extent, density and height of aquatic reed beds at lake chiemsee, bavariaâgermany," *Remote Sensing*, vol. 9, no. 12, p. 1308, 2017.

[61] G. Mandlburger, C. Hauer, M. Wieser, and N. Pfeifer, "Topo-bathymetric lidar for monitoring river morphodynamics and instream habitatsâa case study at the pielach river," *Remote Sensing*, vol. 7, no. 5, pp. 6160–6195, 2015.

[62] K. A. Islam, D. Perez, and J. Li, "A transfer learning approach for the 2018 femh voice data challenge," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 5252–5257, IEEE, 2018.

[63] D. Banerjee, K. Islam, G. Mei, L. Xiao, G. Zhang, R. Xu, S. Ji, and J. Li, "A deep transfer learning approach for improved post-traumatic stress disorder diagnosis," in *Data Mining (ICDM), 2017 IEEE International Conference on*, pp. 11–20, IEEE, 2017.

[64] F. Li, L. Tran, K.-H. Thung, S. Ji, D. Shen, and J. Li, "A robust deep model for improved classification of ad/mci patients," *IEEE journal of biomedical and health informatics*, vol. 19, no. 5, pp. 1610–1616, 2015.

[65] R. Ning, C. Wang, C. Xin, J. Li, and H. Wu, "Deepmag: Sniffing mobile apps in magnetic field through deep convolutional neural networks," in *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 1–10, IEEE, 2018.

[66] D. Perez, D. Banerjee, C. Kwan, M. Dao, Y. Shen, K. Koperski, G. Marchisio, and J. Li, "Deep learning for effective detection of excavated soil related to illegal tunnel activities," in *Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2017 IEEE 8th Annual*, pp. 626–632, IEEE, 2017.

[67] A. Misra, Z. Vojinovic, B. Ramakrishnan, A. Luijendijk, and R. Ranasinghe, "Shallow water bathymetry mapping using support vector machine (svm) technique and multispectral imagery," *International journal of remote sensing*, vol. 39, no. 13, pp. 4431–4450, 2018.

[68] E. C. Geyman and A. C. Maloof, "A simple method for extracting water depth from multispectral satellite imagery in regions of variable bottom type," *Earth and Space*

Science, vol. 6, no. 3, pp. 527–537, 2019.

[69] H. Mohamed, A. Negm, M. Zahran, and O. C. Saavedra, "Bathymetry determination from high resolution satellite imagery using ensemble learning algorithms in shallow lakes: case study el-burullus lake," *International Journal of Environmental Science and Development*, vol. 7, no. 4, p. 295, 2016.

[70] H. Mohamed, M. Salah, K. Nadaoka, M. Zahran, *et al.*, "Assessment of proposed approaches for bathymetry calculations using multispectral satellite images in shallow coastal/lake areas: a comparison of five models," *Arabian Journal of Geosciences*, vol. 10, no. 2, p. 42, 2017.

[71] A. Jalilzadeh and S. Behzadi, "Machine learning method for predicting the depth of shallow lakes using multi-band remote sensing images," *Journal of Soft Computing in Civil Engineering*, vol. 3, no. 2, pp. 54–64, 2019.

[72] T. Sagawa, Y. Yamashita, T. Okumura, and T. Yamanokuchi, "Satellite derived bathymetry using machine learning and multi-temporal satellite images," *Remote Sensing*, vol. 11, no. 10, p. 1155, 2019.

[73] F. Tonion, F. Pirotti, G. Faina, and D. Paltrinieri, "a machine learning approach to multispectral satellite derived bathymetry," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 3, pp. 565–570, 2020.

[74] B. Ai, Z. Wen, Z. Wang, R. Wang, D. Su, C. Li, and F. Yang, "Convolutional neural network to retrieve water depth in marine shallow water area from remote sensing

images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 2888–2898, 2020.

[75] K. Dickens and A. Armstrong, "Application of machine learning in satellite derived bathymetry and coastline detection," *SMU Data Science Review*, vol. 2, no. 1, p. 4, 2019.

[76] A. Collins, K. L. Brodie, S. Bak, T. Hesser, and M. Farthing, "Nearshore bathymetric inversion and uncertainty estimation from synthetic imagery using a 2d fully convolutional neural network," in *AGU Fall Meeting 2020*, AGU, 2020.

[77] H. Ghorbanidehno, Y. Qian, J. H. Lee, M. Farthing, T. Hesser, P. K. Kitanidis, E. F. Darve, and M. Forghani, "Deep learning techniques for nearshore and riverine bathymetry estimation using water-surface observations," in *Ocean Sciences Meeting 2020*, AGU, 2020.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[79] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[80] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11065–11074, 2019.

[81] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0, 2018.

[82] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, 2015.

[83] Y. Liu, Y. Wang, S. Wang, T. Liang, Q. Zhao, Z. Tang, and H. Ling, "Cbnet: A novel composite backbone network architecture for object detection," *arXiv preprint arXiv:1909.03625*, 2019.

[84] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[85] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pp. 131–135, IEEE, 2017.

[86] F. Li, L. Tran, K.-H. Thung, S. Ji, D. Shen, and J. Li, "Robust deep learning for improved classification of ad/mci patients," in *International Workshop on Machine Learning in Medical Imaging*, pp. 240–247, Springer, 2014.

[87] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, 2019.

[88] M. M. U. Chowdhury, F. Hammond, G. Konowicz, C. Xin, H. Wu, and J. Li, "A few-shot deep learning approach for improved intrusion detection," in *Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2017 IEEE 8th Annual*, pp. 456–462, IEEE, 2017.

[89] R. Ning, J. Li, C. Xin, and H. Wu, "Invisible poison: A blackbox clean label backdoor attack to deep neural networks," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, IEEE, 2021.

[90] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019.

[91] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[92] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[93] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[94] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[95] D. P. Ibanez, J. Li, Y. Shen, J. Dayanghirang, S. Wang, and Z. Zheng, "Deep learning for pulmonary nodule ct image retrievalâan online assistance system for novice radiologists," in *Data Mining Workshops (ICDMW), 2017 IEEE International Conference on*, pp. 1112–1121, IEEE, 2017.

[96] R. Li, W. Zhang, H.-I. Suk, L. Wang, J. Li, D. Shen, and S. Ji, "Deep learning based imaging data completion for improved brain disease diagnosis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 305–312, Springer, 2014.

[97] F. Li, G. Zhang, W. Wang, R. Xu, T. Schnell, J. Wen, F. McKenzie, and J. Li, "Deep models for engagement assessment with scarce label information," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 4, pp. 598–605, 2017.

[98] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[99] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.

[100] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[101] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7167–7176, 2017.

[102] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[103] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[104] K. R. Weiss and T. M. Khoshgoftaar, "An investigation of transfer learning and traditional machine learning algorithms," in *Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on*, pp. 283–290, IEEE, 2016.

[105] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1717–1724, IEEE, 2014.

[106] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[107] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898, 2014.

[108] W. Zhang and D. Wu, "Discriminative joint probability maximum mean discrepancy (djp-mmd) for domain adaptation," *arXiv*, pp. arXiv–1912, 2019.

[109] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*, pp. 443–450, Springer, 2016.

[110] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5715–5725, 2017.

[111] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *arXiv preprint arXiv:2001.05566*, 2020.

[112] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[113] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," pp. 3859–3869, 2017.

[114] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[115] G. Hinton, N. Frosst, and S. Sabour, "Matrix capsules with em routing," 2018.

[116] T. Iesmantas and R. Alzbutas, "Convolutional capsule network for classification of breast cancer histology images," *arXiv preprint arXiv:1804.08376*, 2018.

[117] P. Afshar, A. Mohammadi, and K. N. Plataniotis, "Brain tumor type classification via capsule networks," *arXiv preprint arXiv:1802.10200*, 2018.

[118] E. Xi, S. Bing, and Y. Jin, "Capsule network performance on complex data," *arXiv preprint arXiv:1712.03480*, 2017.

[119] J. Kirk, *Light and Photosynthesis in the Sea*. Cambridge University Press, 2nd ed. ed., 1994.

[120] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[121] D. Perez, K. A. Islam, B. Schaeffer, R. Zimmerman, V. Hill, and J. Li, "Deepcoast: Quantifying seagrass distribution in coastal water through deep capsule networks." in The First Chinese Conference on Pattern Recognition and Computer Vision (PRCV), (2018).

[122] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[123] M. Lyons, S. Phinn, and C. Roelfsema, "Long term land cover and seagrass mapping using landsat sensors from 1972–2010 in the coastal environment of south east queensland, australia," in *34th International Symposium on Remote Sensing of Environment-The GEOSS Era: Towards Operational Environmental Monitoring*, 2011.

[124] A. Knudby, C. Newman, Y. Shaghude, and C. Muhando, "Simple and effective monitoring of historic changes in nearshore environments using the free archive of landsat imagery," *International Journal of Applied Earth Observation and Geoinformation*, vol. 12, pp. S116–S122, 2010.

[125] M. Gullström, B. Lundén, M. Bodin, J. Kangwe, M. C. Öhman, M. S. Mtolera, and M. Björk, "Assessment of changes in the seagrass-dominated submerged vegetation of tropical chwaka bay (zanzibar) using satellite remote sensing," *Estuarine, Coastal and Shelf Science*, vol. 67, no. 3, pp. 399–408, 2006.

[126] A. G. Dekker, V. E. Brando, and J. M. Anstee, "Retrospective seagrass change detection in a shallow coastal tidal australian lake," *Remote Sensing of Environment*, vol. 97, no. 4, pp. 415–433, 2005.

[127] M. M. Coffer, B. A. Schaeffer, R. C. Zimmerman, V. Hill, J. Li, K. A. Islam, and P. J. Whitman, "Performance across worldview-2 and rapideye for reproducible seagrass mapping," *Remote Sensing of Environment*, vol. 250, p. 112036, 2020.

[128] A. Seilacher, "Bathymetry of trace fossils," *Marine geology*, vol. 5, no. 5-6, pp. 413–428, 1967.

[129] V. E. Kostylev, B. J. Todd, G. B. Fader, R. Courtney, G. D. Cameron, and R. A. Pickrill, "Benthic habitat mapping on the scotian shelf based on multibeam bathymetry, surficial geology and sea floor photographs," *Marine Ecology Progress Series*, vol. 219, pp. 121–137, 2001.

[130] M. F. Wilson, B. OâConnell, C. Brown, J. C. Guinan, and A. J. Grehan, "Multiscale terrain analysis of multibeam bathymetry data for habitat mapping on the continental slope," *Marine Geodesy*, vol. 30, no. 1-2, pp. 3–35, 2007.

[131] B. McConnell, C. Chambers, and M. Fedak, "Foraging ecology of southern elephant seals in relation to the bathymetry and productivity of the southern ocean," *Antarctic Science*, vol. 4, no. 4, pp. 393–398, 1992.

[132] J. C. Brock, C. W. Wright, T. D. Clayton, and A. Nayegandhi, "Lidar optical rugosity of coral reefs in biscayne national park, florida," *Coral Reefs*, vol. 23, no. 1, pp. 48–59, 2004.

[133] C. W. Finkl, L. Benedet, and J. L. Andrews, "Interpretation of seabed geomorphology based on spatial analysis of high-density airborne laser bathymetry," *Journal of Coastal Research*, pp. 501–514, 2005.

[134] "nationalgeographic bathymetry." `https://www.nationalgeographic.org/encyclopedia/bathymetry/`. (Date last Accessed July 15, 2019).

[135] R. J. Huizinga, "Bathymetric and velocimetric surveys at highway bridges crossing the missouri river near kansas city, missouri, june 2–4, 2015," tech. rep., US Geological Survey, 2016.

[136] R. J. Huizinga, "Bathymetric and velocimetric surveys at highway bridges crossing the missouri and mississippi rivers on the periphery of missouri, june 2014," tech. rep., US Geological Survey, 2015.

[137] "usgs bathymetry." https://www.usgs.gov/centers/oki-water/science/bathymetric-surveys?qt-science_center_objects=0#qt-science_center_objects. (Date last Accessed July 15, 2019).

[138] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[139] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.

[140] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[141] L. Rokach and O. Maimon, "Decision trees," in *Data mining and knowledge discovery handbook*, pp. 165–192, Springer, 2005.

[142] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[143] L. Mason, J. Baxter, P. L. Bartlett, and M. R. Frean, "Boosting algorithms as gradient descent," in *Advances in neural information processing systems*, pp. 512–518, 2000.

[144] D. R. Lyzenga, "Passive remote sensing techniques for mapping water depth and bottom features," *Applied optics*, vol. 17, no. 3, pp. 379–383, 1978.

[145] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," *arXiv preprint arXiv:1711.03213*, 2017.

APPENDIX A

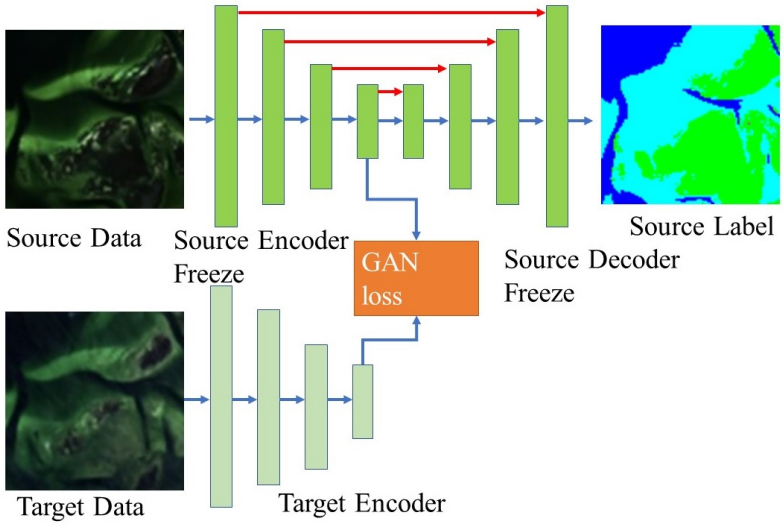
UNSUPERVISED DOMAIN ADAPTATION (UDA) ACROSS LOCATION

A.1 Method

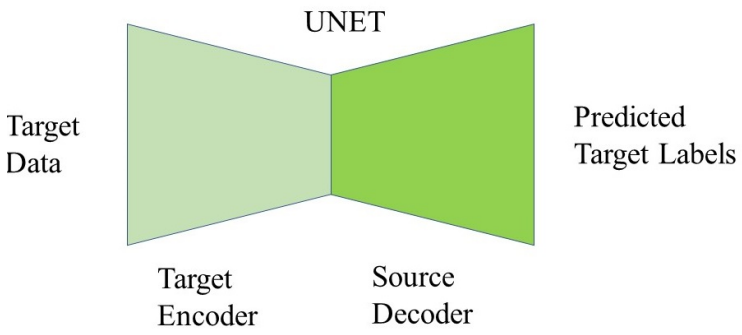
A.1.1 UDA in the Encoding layers

We propose an unsupervised domain adaptation method for seagrass detection where we adapt the encoding layers of the target segmentation model using generative adversarial network (GAN) based loss. We modify the target embedding layer using the GAN loss so that the domain classifier gets confused when determining whether the samples are coming from the source domain or target domain. Given the source domain samples and labels are x_s and y_s, the target domain has only unlabeled sample x_t. First, we train the U-Net segmentation model using the source dataset (x_s and y_s). Then, we use the target domain unlabeled dataset to adapt the seagrass detection model in the target domain as shown in Fig. 34.

We use the GAN loss [98] to adapt the source domain U-Net's encoder layers, E_s to the target domain. This unsupervised domain adaptation step tries to learn a target encoder, E_t based on E_s using the unlabeled samples from target domains shown in Fig. 33a. First, the target encoder E_t is initialized with the source encoder weights, E_s. Then, E_t and



(a) Unsupervised domain adaptation in the encoder layer.



(b) Predicting the label on the target domain.

Fig. 33: Proposed unsupervised domain adaptation (UDA) architecture in the Encoder layer (a) Unsupervised domain adaptation in the encoder layer, (b) Predicting the label on the target domain.

the discriminator D in Fig. 33a are trained by MinMax optimization with the GAN loss $L_{adv_D}(X^s, X^t, E_s, E_t)$,

$$L_{adv_D}(X^s, X^t, E_s, E_t) = E_{x^s \sim X^s}[logD(E_s(x^s))] -$$

$$E_{x^t \sim X^t}[log(1 - D(E_t(x^t)))] \quad (36)$$

The target embedding function E_t modifies its parameters using the following generator loss,

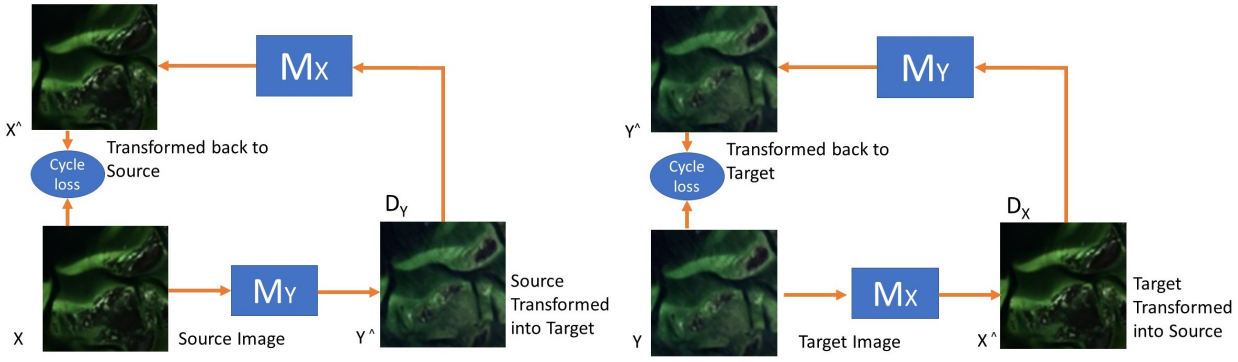
$$Min_{E_t} L_{adv_E}(X^s, X^t, D) = -E_{x^t \sim X^t}[logD(E_t(x^t))] \quad (37)$$

This is the same as the standard GAN loss where E_t modifies its weights to mimic source domain sample embedding. During training, we keep E_s fixed while changing E_t. Once the model converges, the proposed model utilizes the trained target encoder E_t to encode the image and the source decoder F_s to classify the target images as shown in Fig. 33b.

A.1.2 UDA in the soft-max layer with an Image-to-image translation algorithm

A.1.2.1 Cycle-GAN

After comparing the image spectral reflectance signatures for the SJB, KB, and SGS locations in Fig. 10, we find that seagrass's spectral signature varies a lot from location to location. Source location may contain deep tea-sediment or CDOM in the water whereas the target location may contain a different property. As a result, the classifier trained at source location may perform poorly at the target location. In the proposed approach, we use an image-to-image translation algorithm to add target domain properties to source domain images. Then, we further fine-tune the source classifier with the transformed source images.



(a) Source Cycle. (b) Target Cycle.

Fig. 34: Image to Image translation for seagrass detection using Cycle-GAN based approach (a) Source domain cycle (b) Target domain cycle.

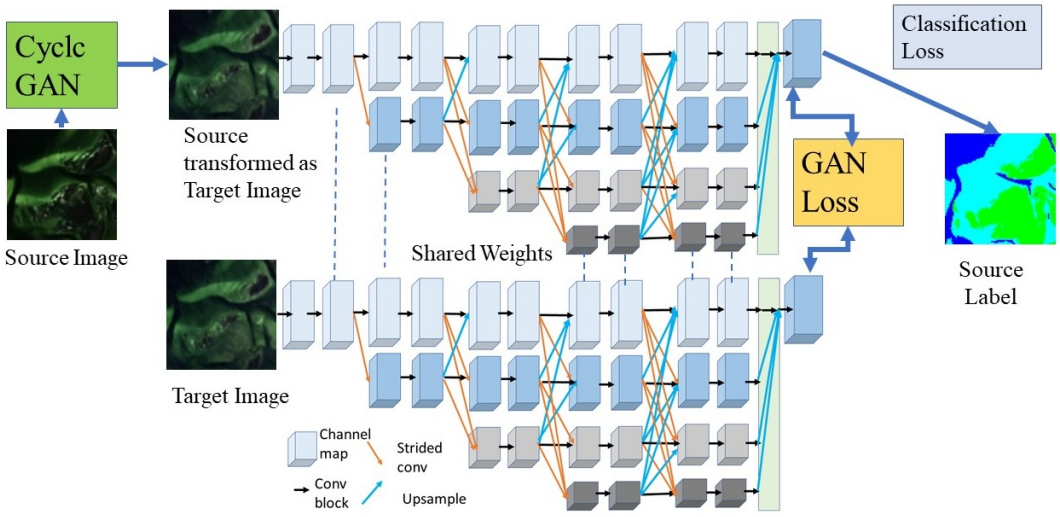


Fig. 35: Unsupervised domain adaptation across location images. Cycle-GAN model is applied to the source images to translate into target domain before applying the UDA.

The source and target images are unpaired, and we use the Cycle-GAN model to perform the image-to-image translation.

Let X and Y denote source and target domains, and have n and m numbers of training samples, $\{x_i\}_{i=1}^{n}$, where $x_i \epsilon X$ and $\{y_i\}_{j=1}^{m}$, where $y_j \epsilon Y$, respectively. The Cycle-GAN model uses two mapping functions, $M_Y : X \rightarrow Y$ for source to target mapping, and $M_X : Y \rightarrow X$ for target to source mapping. Cycle-GAN model uses two adversarial discriminators D_x and D_y. Discriminator D_x tries to distinguish original images $\{x\}$ from translated images $\{M_X(y)\}$. Similarly, discriminator D_y aims to discriminate original images $\{y\}$ from translated images $\{M_Y(x)\}$.

Adversarial loss: We apply the following adversarial losses to $M_Y : X \rightarrow Y$ and D_Y,

$$L_{GAN}(M_Y, D_Y, X, Y) = E_{y \sim P_{data}(x)}[logD_Y(y)] - E_{x \sim P_{data}(x)}[log(1 - D_Y(M_Y(x))))] \quad (38)$$

where M_Y tries to generate images $M_Y(x)$ that look similar to images from domain Y. Here, $M_Y(x)$ and D_y compete with each other and improve each other's performances. A similar objective function for M_X and D_x is defined as follows:

$$L_{GAN}(M_X, D_x, X, Y) = E_{x \sim P_{data}(y)}[logD_X(x)] - E_{y \sim P_{data}(y)}[log(1 - D_X(M_X(y))))] \quad (39)$$

Cycle-consistency Loss: The adversarial loss can not ensure that the learned mapping function will map source images to target images. To reduce the space of mapping function miss-alignment, we use a cycle-consistency loss between source and target domain images. For each image x from domain X, the cycle-consistency loss should bring the translated image back to original domain as $X \rightarrow M_Y(x) \rightarrow M_X(M_Y(x)) \approx x$. We define the cycle-consistency loss as follows:

$$L_{cyc}(M_Y, M_X) = E_{x \sim P_{data}(x)}[||M_X(M_Y(x)) - x||_1] + E_{y \sim P_{data}(y)}[||M_Y(M_X(y)) - y||_1] \quad (40)$$

The overall objective function will be,

$$L(M_Y, M_X, D_X, D_Y) = L_{GAN}(M_Y, D_Y, X, Y) + L_{GAN}(M_X, D_x, X, Y) + \lambda * L_{cyc}(M_Y, M_X) \tag{41}$$

where λ controls the relative importance of the two objectives. We aim to solve following:

$$M_Y^*, M_X^* = arg\ min_{M_Y, M_X}\ max_{D_X D_Y}\ L(M_Y, M_X, D_X, D_Y) \tag{42}$$

A.1.2.2 Unsupervised Domain Adaptation(UDA)

The Cycle-consistence GAN model may fail to map source domain images to the target domain effectively. To ensure the classifier can learn target domain image properties, we further adapt the source segmentation model to the target domain using a GAN-based UDA method. After mapping the source images x to the target domain $X_{s_{transformed}} = M_Y(x_s)$, we apply a GAN based loss to modify the last layer of source segmentation model G to fit in the target domain.

We finetune the segmentation network G with the transformed images from source domain $X_{s_{transformed}}$ with their original labels y_s using the cross-entropy loss $\zeta_{seg}(x_s)$ defined in Equ. 21. We feed target images to the segmentation model G and get the soft-max output $P_t = G(X_t)$. Similarly, we feed $X_{s_{transformed}}$ to the segmentation model G and get the soft-max output $P_{s_{transformed}} = G(X_{s_{transformed}})$. The proposed approach's overall objective is to make the segmentation model's features before the soft-max layer $P_{s_{transformed}}$ and P_t similar to each other using the GAN loss. We feed the soft-max output of the source images $P_{s_{transformed}}$ into a discriminator network D using a cross-entropy loss. We train the cross-entropy loss for two classes: source domain and target domain. We use the following loss to

train the discriminator $\zeta_d(P)$,

$$\zeta_d(P) = -\sum_{h,w}(1-z)log(D(P_t)^{h,w,0}) + zlog(D(P_{s_{transformed}})^{h,w,1}) \tag{43}$$

where $z = 1$ if the samples are from the source domain, and $z = 0$ if the samples are from the target domain. To make the distribution of P_t closer to that of $P_{s_{transformed}}$, we use the following adversarial loss to train the classifier D,

$$\zeta_{adv}(x_t) = -\sum_{h,w} log(D(P_t)^{h,w,1}) \tag{44}$$

Using this loss, the segmentation network G improves its performance to fool the discriminator by maximizing the probability of target predictions considered as being the source prediction. The overall loss for the domain adaptation contains both segmentation loss and GAN loss as,

$$\zeta(x_{s_{transformed}}, y_s, x_t) = \zeta_{seg}(x_{s_{transformed}}, y_s) + \lambda_{adv}\zeta_{adv}(x_t) \tag{45}$$

where λ_{adv} is the weight to balance between the segmentation loss $\zeta_{seg}(x_{s_{transformed}}, y_s)$ and the GAN loss $\zeta_{adv}(x_t)$.

A.2 Experimental Setup

A.2.1 Worldview-2 Multi-spectral Images

To develop an unsupervised domain adaptation (UDA) approach across location images, we use the worldview-2 multi-spectral satellite images from three locations of Florida: Saint Joseph Bay (SJB), Keaton Beach (KB), and Saint George Sound (SGS). These images

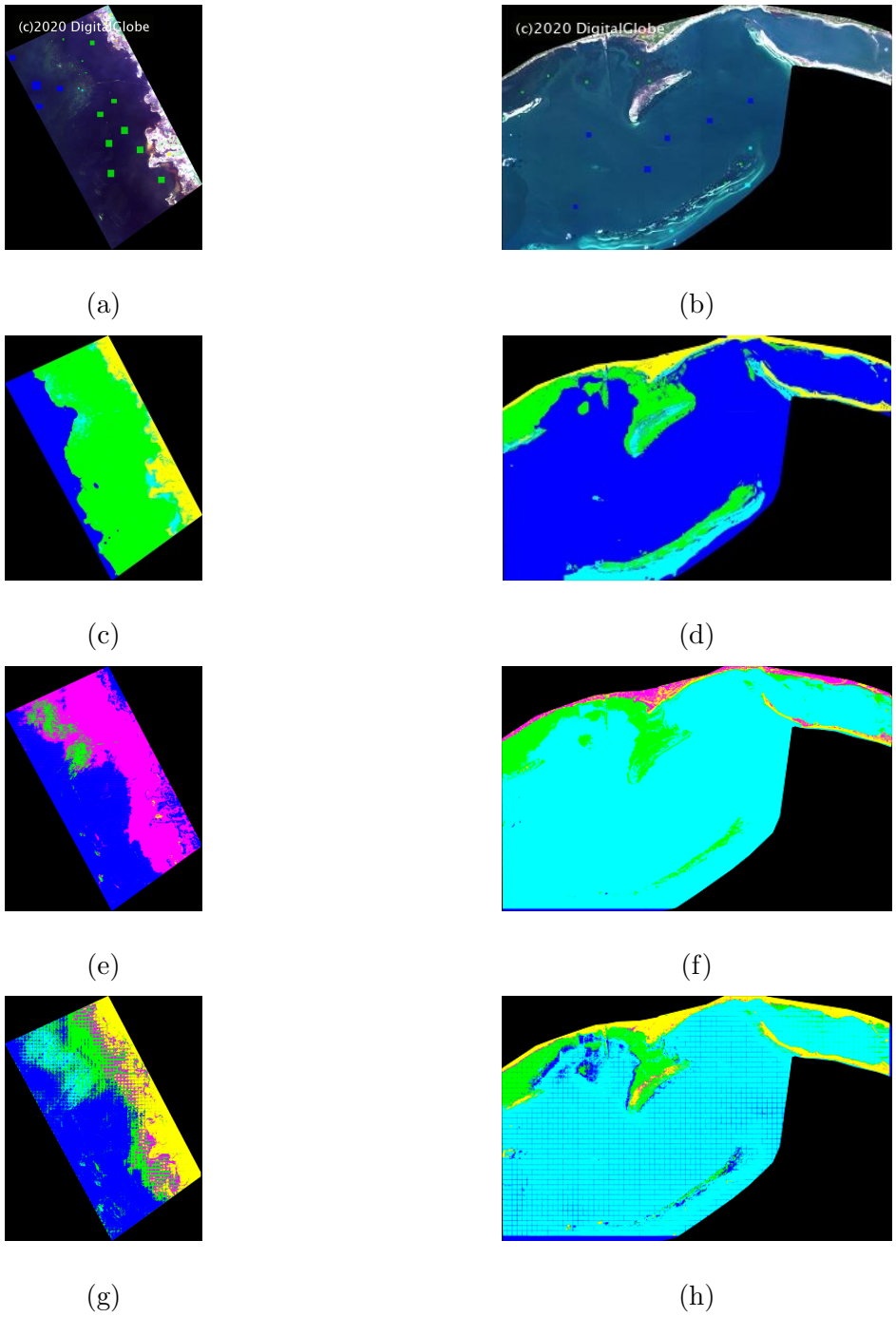


Fig. 36: UDA results using U-Net and GAN based domain adaptation methods in encoder layers: (a) and (b) MS imagery, (c) and (d) physics based ground truth (e) and (f) source model (g) and (h) UDA results after matching all encoder layers.

have eight spectral bands: coastal blue, blue, green, yellow, red, red edge, NIR-1, and NIR-2 with a spatial resolution of $2m$. We use selected regions by domain experts for five classes from these images. The classes seagrass, sand, deep-sea, intertidal, and land are shown using green, cyan, blue, magenta, and yellow colors in Fig 9, respectively. We pre-processed these images with an atmospheric correction method described in Section 3.3.1. We train a source seagrass detection model at one of the locations. We then adapt the model to the other locations using the proposed UDA approaches.

A.2.2 U-Net multiple Layers

To classify the target domain images, we use a UDA method (Section A.1.1) without using any labeled data from the target domain. In the last step, we trained a U-Net model for the source domain using the labeled data from the active learning algorithm. We first initialize the target model using the source model weights. Then, we modify the target encoder layers using a GAN-based loss so that it can use the source decoder model to classify the target imagery. We jointly optimize all four encoding layers using a GAN-based loss and unlabelled target domain data. The discriminator classifies whether the samples are from the source domain or target domain. The target model's encoding layers jointly modify its parameters to fool the domain discriminator. When the source feature distribution and target feature distribution align, we use the source decoder layers to classify the target domain images. We use the SJB location trained U-Net model as the source model and the proposed UDA to detect seagrass in the target KB and SGS locations as shown in Fig. 36g 36h, respectively.

A.3 Across Location Results

The source and target domains have larger domain divergence across locations as compared to that in the time-series images. We compare two UDA methods described in Section A.1.1 and Section A.1.2. First, we use a U-Net based domain adaptation approach to handle the domain-shift where we perform adaptation in each encoding layer of the U-Net model. In another approach, we use a HR-Net based UDA approach where we use a cycle-GAN to add the target domain features in the source classification layer as well as the GAN-based encoding layer's feature adaptation. Both approaches achieve better results than the base-line algorithm.

A.3.1 Active Learning and Source Segmentation model

In the active learning procedure of the WorldView-2 image, we first train the DCNN classification model using the domain expert operator's defined regions. We use a $5 * 5 * 8$ input patch-size to extract patches for seagrass, deepsea, sand, land and intertide classes. WorldView-2 MS images achieve better performance using a $5 * 5 * 8$ patch size because it has $2m$ spatial resolution as compared to Landsat-8 image's $30m$ spatial resolution. The DCNN model has six layers: two convolutional layers with a ReLU activation function, two dropout layers, one flatten layer and a softmax layer. The 1^{st} and 2^{nd} convolutional layers have the 16 convolutional kernels of a size of 2-by-2 and 64 convolutional kernels of a size of 4-by-4 respectively. After each convolutional layer, we utilize the dropout technique with a dropout probability of 0.1 to reduce over-fitting. Then, we use a flatten layer to convert it into 64 hidden vector features. At the last layer, we use a fully connected layer to perform classification for the five classes. We train the model for 500 epochs for training. Once

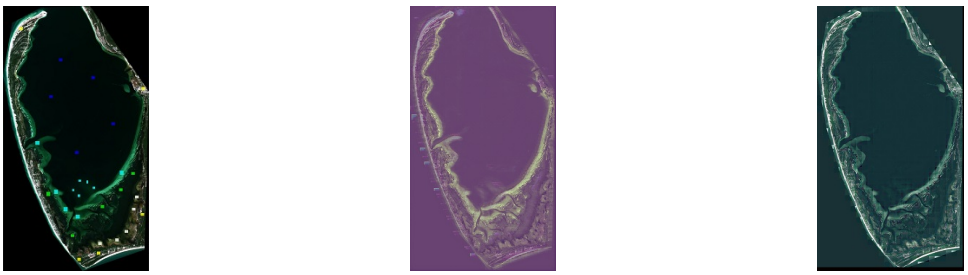
trained, we use the model to classify all the available samples from the same location. Then we evaluate the performance of the DCNN model by the domain expert operator. If the performance of the DCNN shows high agreement with the expert operator evaluation, we use the DCNN predicted map as the ground truth for the segmentation model.

We train both U-Net and HR-Net segmentation models in the source domain using the pixel-wise label mapping from the active learning approach described in Section 5.2.1. For the U-Net model training, we use an input image size of $256 * 256$, batch size of 8, and samples per class of 1000 to train the model. We use four encoding layers to encode the image into a lower dimension vector. Then, we use four decoding layers to decode it back to the original image. Once U-Net is trained on the source domain, we directly apply the trained source U-Net model to the target location without adaptation, and results are shown in Fig. 36e, 36f. We also use an input image size of $256 * 256$, batch size of 8, and 1000 samples per class to train the HR-Net model at SJB as the source model. We directly apply the source model to locations KB and SGS, and results are shown in Fig. 38e, 38f, respectively.

A.3.2 UDA in the last layer with Cycle-GAN

A.3.2.1 Cycle-GAN

The target image may not have the same properties as the source image. As a result, the trained source classifier fails to classify target images. We proposed to use an unpaired image-to-image translation algorithm to add the target domain properties to the source images and then update the seagrass detection model. The proposed model maps the



(a) SJB (b) SJB transformed as KB (c) SJB transformed as SGS

Fig. 37: Cycle-GAN transforms source domain image to target domain. (a) source image at SJB (b) SJB image transformed to KB (c) SJB image transformed to SGS.

source image in the target domain using the image-to-image translation algorithm. First, we initialize the target model with the source HR-Net model. We then fine-tune the model with transformed images from the source and with their original labels. The transformed images by cycle-GAN are shown in Fig. 37.

We perform two sets of image-to-image translation experiments from the source to the target domain: SJB $\leftrightarrow$ KB and SJB $\leftrightarrow$ SGS. After training the cycle-GAN model for these two pairs, we use the trained generator model to transform the SJB image $\rightarrow$ KB domain in Fig. 37b and SJB image $\rightarrow$ SGS domain in Fig. 37c, respectively. We also use the Cycada model [145] and Cycle-GAN model with VGG loss to repeat the same experiments. In the Cycle-GAN model with VGG loss, we replace the L_1 loss with VGG-based loss. After comparison, we found that the Cycle-GAN model performs than the other two approaches.

A.3.2.2 GAN-based feature adaptation

If we just fine-tune the target model with transformed images from the source domain,

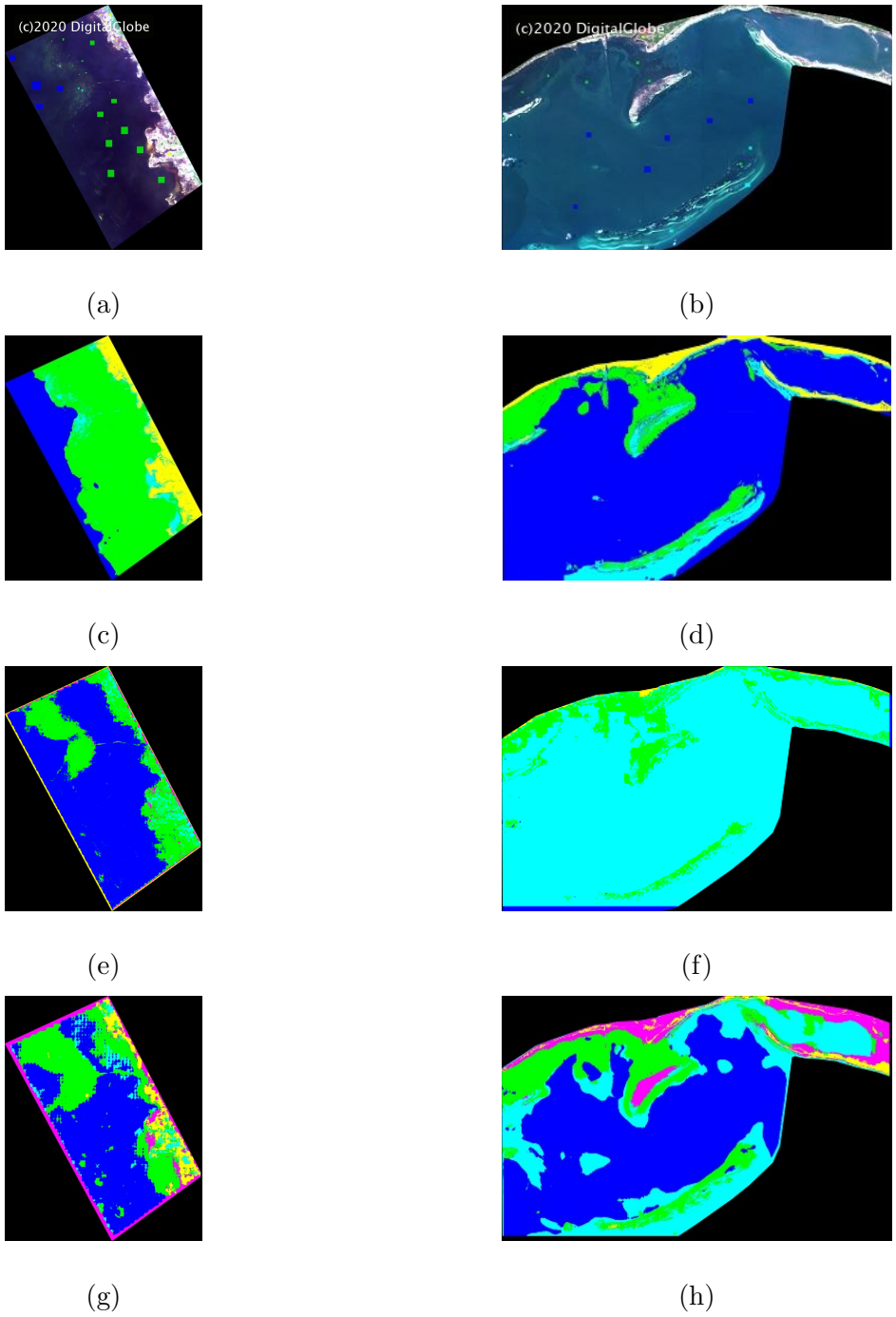


Fig. 38: UDA results with a cycle-GAN model with HRNet backbone. (a) and (b) MS imagery, (c) and (d) Physics based ground truth (e) and (f) without adaptation (g) and (h) UDA results.

it may not give us good performance. We also use a GAN-based loss to adapt the target model's features in the layer before the soft-max layer to match the source domain features. We first initialize the target domain model with the well trained source model weights. We then modify the target model using a joint optimization of the two losses: segmentation loss and GAN-based feature adaptation loss. In the GAN-based feature adaptation model, the discriminator classifies whether the input samples are from the source domain or target domain. The target model modifies its weights so that it can fool the discriminator. We use the joint optimization mechanism to ensure that the target model learns the target domain features and performs the seagrass detection in the target domain simultaneously. We use the proposed model to perform domain adaptation at KB and SGS, and results are shown in Fig. 38g 38h respectively.

A.4 Discussion

We align multiple encoder layers of the U-Net model between the source and the target domains to detect seagrasses in the target domain as described in Section A.1.1. The UDA model predicts classification maps for target domains at KB and SGS locations, and results are shown in Fig. 36g and Fig. 36h. The baseline results are shown in Fig. 36e and Fig. 36f for the KB and SGS locations, respectively. The baseline model fails to detect seagrass correctly in both locations and predicts intertidal as seagrass at KB location. We improve the seagrass detection performance using the encoder layers-based UDA approach, and results are shown in Fig. 36g, and Fig. 36h, but it still fails to correctly identify the seagrass classes at KB location. The model under-predicted the seagrass class area at the KB location by a large margin due to a large domain shift. Besides modifying all the

encoder layers jointly, we also tried other variations of the proposed approach. We find that modifying all encoder layers is a better approach.

We use the cycle-GAN model to transform the source SJB image to the target domain as shown in Fig 37. The Cycle-GAN based translated images are SJB $\rightarrow$ KB in Fig. 37b and SJB $\rightarrow$ SGS in Fig. 37c. Instead of Cycle-GAN, we also apply cycada and cycle-GAN with VGG loss to translate SJB $\rightarrow$ KB and SJB $\rightarrow$ SGS. However, these algorithms performed worse than the Cycle-GAN model in unpaired multispectral image translation.

We directly apply the trained HR-Net model at source SJB location to target locations at KB and SGS, and label it as baseline model. The baseline model's performance in scenario SJB $\rightarrow$ KB is shown in Fig. 38e and scenario of SJB $\rightarrow$ SGS is shown in Fig. 38f. The baseline model fails to detect seagrass at the target locations due to high domain divergence between the source and target domains. It is also impossible for the model to learn the difference without any adaptation. We perform UDA at the target locations using segmentation loss and GAN based feature adaptation loss in the layer before softmax layer described in Section A.1.2. We train the target model by fine-tuning with the transformed source images and align the target features with the source features using the GAN based loss. The proposed UDA based classification maps for SJB $\rightarrow$ KB are shown in Fig. 38g and for SJB $\rightarrow$ SGS are shown in Fig. 38h. With joint optimization, the proposed approach detected seagrass superbly as compared with the baseline model at the location of SGS. This approach performed better than the baseline model at KB; however, the detected seagrass areas is less than that detected by the reference physics based approach.

If we compare the UDA performances presented in Section A.1.1 and Section A.1.2,

the performances shown in Section A.1.2 are better. If we compare the UDA based classification maps for SJB → SGS in Fig. 36h and Fig. 38h, the UDA of Section A.1.2 performed much better as compared with the UDA of Section A.1.1. The UDA performances in SJB → KB as shown in Fig. 36g and Fig. 38g are similar where both approaches performed poorly due to high domain-shift, which is the planned task in our future work.

VITA

Kazi Aminul Islam

Department of Electrical and Computer Engineering

Old Dominion University

Norfolk, VA 23529

EDUCATION

Doctor of Philosophy in Electrical and Computer Engineering August 2021

Old Dominion University, Norfolk Virginia USA

Masters of Engineering Science in Electrical Engineering May 2016

Lamar University, Beaumont Texas USA

Bachelor of Science in Electrical and Electronic Engineering April 2010

Khulna University of Engineering and Technology, Khulna Bangladesh