ETH zürich

IGP
Institut für Geodäsie und Photogrammetrie

Interdisciplinary Project

Machine Learning-based Lake Ice Monitoring Using UAVs

Han Sun

Autumn Semester 2020

Professorship

Prof. Konrad Schindler (PRS)

Prof. Fabian Walter (VAW)

Supervisors

Manu Tom (PRS)

Nikolai Adi Kalischek (PRS)

Abstract

The duration and extent of lake ice are vital indicators of climate change. It has been recognized as part of the Essential Climate Variables (ECV) by the Global Climate Observing System (GCOS). One way to monitor the dynamics of lake ice is through semantic segmentation of lake images. In this project, we aim to detect the frozen parts of lake Sils (Graubuenden, Switzerland) using RGB images captured with UAVs. By adopting DeepLabV3+, we achieve an Intersection-over-Union (IoU) value of 88% for the segmentation task, proven to be accurate enough to monitor lake ice changing for three days. Besides, we attempt to adopt up-to-date domain adaptation approaches to help transfer learning between different lake ice datasets (Photi-Lakeice webcam dataset and the UAV dataset). Our experiments show the potential application of domain adaptation methods to reduce required manual annotation work and improve the target dataset's prediction results. Based on the experiment results of various approaches, possible limitations of current unsupervised domain adaptation methods are explored and discussed.

Keywords: Domain Adaptation, Semantic Segmentation, Climate Monitoring, Lake Ice, Webcams, UAV

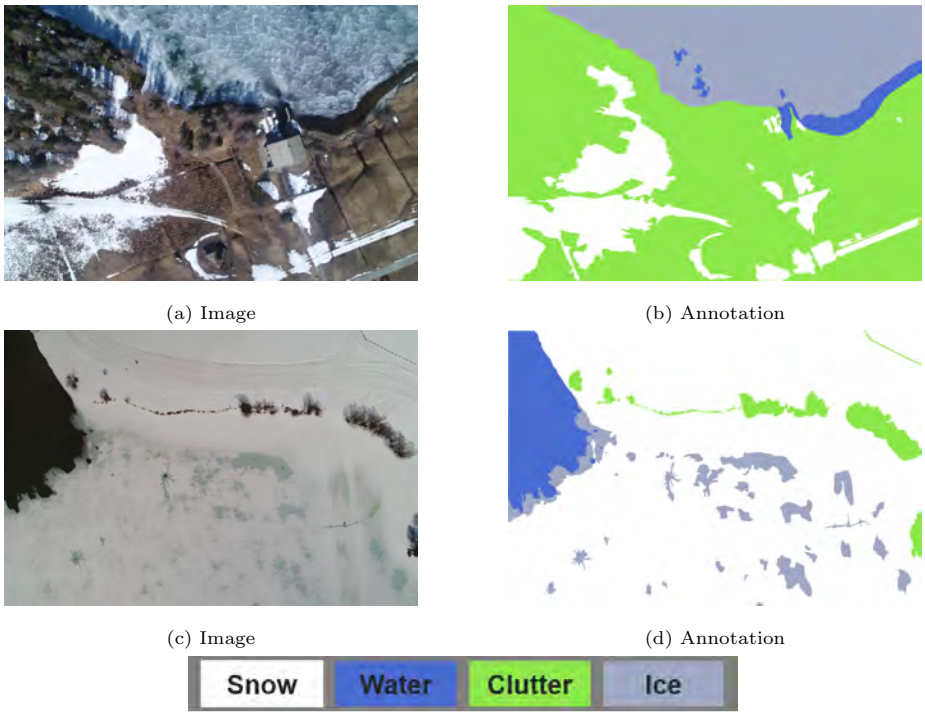
1. Introduction

Lake ice is an essential variable in understanding local and global climate change, and it has recently been recognized as an ECV. The use of machine learning technology to monitor and analyze lake ice provides valuable information for climate research. This work is part of the project "Integrated lake ice monitoring and generation of sustainable, reliable, long time series" initiated and financed by the Federal Office of Meteorology and Climatology (Meteo Swiss) in the framework of GCOS (Global Climate Observing System), Switzerland. Our task is to classify each pixel of images of lake Sils captured by UAVs as water, ice, snow, or clutter (objects on the lake surface and background other than the lake). Example annotated images and the corresponding color code (consistent throughout the report) are shown in Fig. 1. Based on effective segmentation results, we could achieve lake ice monitoring by comparing the percentage of lake ice during a specific period.

We base our approach on the previous work of [1], which shows the effectiveness of deep neural network (DNN) in the lake images' semantic segmentation task. We adopt the DeepLabV3+ model pretrained in this work with another lake ice dataset of webcam images, the Photi-Lakeice dataset, and transfer this model to predict on the new UAV dataset by finetuning with several labeled target images. As those two datasets share the same purpose and categories, considering further reducing the tedious annotation work, we try to incorporate domain adaptation methods. In this project:

1. We adopt DNN for lake ice segmentation task using state-of-art semantic segmentation algorithm DeepLabV3+ [2];
2. We transfer knowledge from the Photi-Lakeice dataset by finetuning its pretrained model with several labeled UAV images;
3. We experiment with three domain adaptation methods: AdaptSegNet [3], Fourier Domain Adaptation (FDA) [4], and Central Moment Discrepancy (CMD) [5] to test their ability in closing a considerably large domain gap in this specific task;

4. As part of this project, to quantitatively evaluate our experiment results, we annotate part of the UAV dataset (77/186 images).



(a) Image (b) Annotation

(c) Image (d) Annotation

(e) Color Code

Figure 1: Examples of UAV Dataset

2. Related Work

2.1. Semantic Segmentation

Semantic segmentation aims to assign each pixel a semantic label in an image. Recently, methods based on convolutional neural network (CNN) have achieved significant progress in semantic segmentation. In 2017, a full convolutional network (FCN) [6] was proposed, enabling CNN to make dense pixel prediction without a fully connected layer. This method promotes the rapid development of CNN in the field of semantic segmentation, and an FCN-based semantic segmentation model trend has emerged. Another problem using CNN

for semantic segmentation is the information loss due to the pooling layers. As pooling layers expand the receptive field and aggregate the context, it results in the loss of location information. In 2015, Google released the first version of the DeepLab model [7], aggregating atrous algorithm, and conditional random (CRF) field in semantic segmentation. Later the authors have incorporated Atrous Special Pyramid Pooling (ASPP) in DeepLabV2 [8]. DeepLabV3 [9] has gone further, using a cascaded deep ASPP module to incorporate multiple contexts [10]. DeepLabV3+ [2] refines segmentation results (especially at the boundary of the target) by incorporating a decoder into the network. Besides, it applies Xception and depthwise separable convolution into ASPP and the decoder.

2.2. Transfer Learning

One main advantage of DNN is their ability to transfer across domains. For one to train a simple task from scratch, millions of labeled data and massive graphics card resources would be needed due to the demand for training such deep networks. Besides, it is not always possible to have clean and ample data resource for training. Transfer learning deals with these problems by utilizing a trained model for a new task at hand. It provides a shortcut to save time and improve performance.

To transfer the knowledge of source data, one could take advantage of the network structure which is proven to be effective in this task, and train on the target dataset using the same structure. Another approach is to directly use the pre-trained model, where the weight of each node in the deeper layers of the well-trained network is migrated to the network to be utilized for the new task. The crucial point of transfer learning is its transferability. Transfer learning only works in deep learning if the model features learned from the first task are general enough to be transferred, or the second task is well-related to the source task. In most cases, the adopted network is fine-tuned on the new dataset to improve its performance on the target task. This still requires supervision in the target domain. To further reduce the manual annotation work, different

approaches have been proposed to diminish the domain shift between source and target domain, helping the network learn more domain-invariant features.

2.3. Domain Adaptation

Over the past few years, machine learning has achieved great success and has benefited real-world applications. However, collecting and annotating datasets for every new task and domain is costly and time-consuming. Sufficient training data may not always be available. Fortunately, the big data era makes a large amount of data available for other domains and tasks [11].

Domain adaptation is a specific instance of transfer learning, which learns and transfers knowledge across domains. It aims to solve the problem of degraded performance due to the domain shift between the source and target domain. It utilizes labeled data in the source domain to perform new tasks on the target domain.

Different approaches have been raised to address this problem. One popular approach that is widely researched is adversarial-based method. It adds a discriminator, which aims to classify whether its input is from the source or the target domain. By doing this, it encourages confusion between the source and target domain [3] [12] [13]. Another approach is discrepancy-based method. It aims to fine-tune the network with unlabeled target data to diminish the domain gap between source and target domains. Various criteria are used to compute the discrepancy across domains. Among those criteria, the statistic criterion aligns the statistical distribution shift between the source and target domains using some mechanisms. Different mechanisms have been proposed, such as maximum mean discrepancy (MMD) [14], Kullback-Leibler (KL) divergence [15], and CMD [5]. Some other methods preprocess the dataset to diminish the difference between the source and target images before training the network [4].

3. Data

3.1. Webcam Data

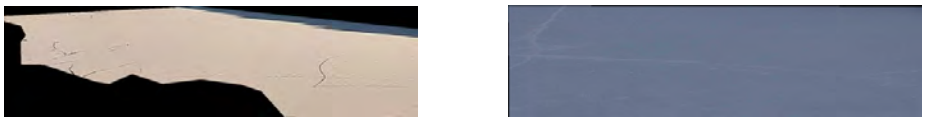
In this project, we use webcam images from the Photi-Lakeice dataset as our source dataset. This dataset is publicly available to the research community. All the webcam images are manually annotated with the LabelMe tool (Wada, 2016) to generate pixel-wise ground truth concerning the work of [1]. Additionally, the dataset is cleaned by discarding excessively noisy images due to bad weather (thick fog, heavy rain, and extreme illumination conditions). The images vary in spatial resolution, magnification, and tilt, depending on camera type (fixed or rotating) and parameters [1]. The annotated Photi-Lakeice dataset could be downloaded on `https://github.com/czarmanu/photi-lakeice-dataset`.

The dataset comprises images from two lakes (St. Moritz, Sihl) and two winters (winter 2016-17 and winter 2017-18) with pixel-wise ground truth for foreground-background segmentation as well as for lake ice segmentation. We only use the data of lake St. Moritz in consideration of image quality. There are two different fixed webcams (Cam0 and Cam1), observing lake St. Moritz at different zoom levels [1]. Details of this dataset are listed in Table 1.

As shown in Fig. 6(a), this dataset has a considerable class imbalance, which needs to be taken care of during the training process. Another problem with this source dataset is its comparably low image quality. This influences the appearance of each class, especially water. As shown in Fig. 3, the water surface contains many shadows cast from the environment, and watercolor changes largely in each image, making it hard to adapt.

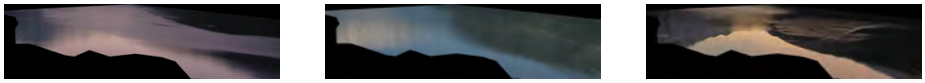
Table 1: Key Figures of the Photi-Lakeice Dataset

Winter	Cam	Image Number	Resolution
Winter 2016-17	Cam0	820	1209×324
	Cam1	1180	1209×324
Winter 2017-18	Cam0	474	1209×324
	Cam1	443	1209×324



(a) Cam0 (b) Cam1

Figure 2: Example Images of Photi-Lakeice Dataset



(a) StMoritz_Cam0_2016_1204_07_24 (b) StMoritz_Cam0_2016_1205_12_25 (c) StMoritz_Cam0_2016_1210_07_49

Figure 3: Comparison of Images of Water Surface of Photi-Lakeice Dataset

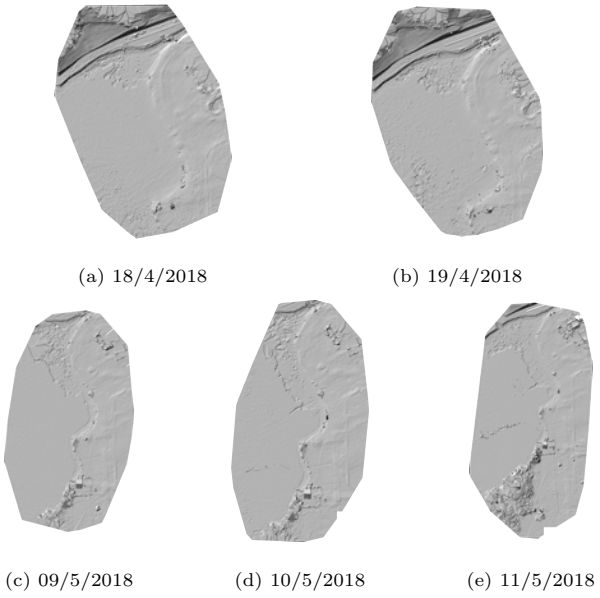
3.2. UAV Data

Our target dataset, the UAV dataset, contains images of lake Sils captured using two UAVs (DJI Phantom 4 Pro, PixHawk). Those images were taken by drone for two days in 2018 (from 18 Apr to 19 Apr) and three days in 2019 (from 09 May to 11 May). Details of this dataset are listed in Table 2. There are in total of 186 images, and 77 of them are manually annotated with the LabelMe tool. From the example images, we could tell that the domain gap between source and target domain is large. The appearance of each class is different. As shown in Fig. 6, the source images do not contain enough information about clutter to adapt. Finally, those images are taken from totally different viewpoints. The target images are taken from the top view and contain abundant background information. While the source images are taken from the side view and only focus on the lake surface.

All images in the target dataset contain geographic location information, and absolute locations could be generated using Pix4D with ground control points (GCPs) recorded in each project. We pick one project for each day with most GCPs as representations. The generated DSM and Mosaic images are shown in Fig. 4 and Fig. 5

Table 2: Key Figures of the UAV Dataset

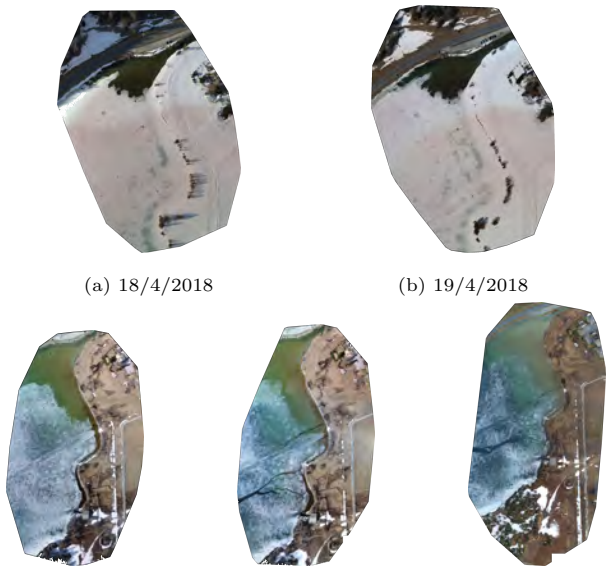
Date	Project	Image Number	Resolution
18/4/2018	filter_ND16_project23	17	5472×3648
	filter_ND04_project24	18	5472×3648
	Nofilter_project25	11	5472×3648
19/4/2018	ND16_new GCPs_project28	17	5472×3648
09/5/2019	Project05	22	5472×3648
10/5/2019	Project06	24	5472×3648
	Project07	20	5472×3648
11/5/2019	Project08	19	5472×3648
	Project09	38	5472×3648



(a) 18/4/2018 (b) 19/4/2018

(c) 09/5/2018 (d) 10/5/2018 (e) 11/5/2018

Figure 4: DSM Images of UAV Dataset



(a) 18/4/2018 (b) 19/4/2018

(c) 09/5/2019 (d) 10/5/2019 (e) 11/5/2019

Figure 5: Mosaic Images of UAV Dataset

(a) Photi-Lakeice Dataset (b) UAV Dataset

Figure 6: Percentage of Each Class of Source and Target Dataset

4. Methodology

4.1. DeepLabV3+

DeepLabV3+ [2] is a state-of-art deep learning model for semantic segmentation. It adopts classical Encoder-Decoder structure, combined with ASPP. Encoder-Decoder Architecture has been proven in literature, such as FPN and U-Net, to be useful in learning location/spatial information, which helps recover boundary information and get shaper segmentation edges. With ASPP, the network

is able to encode multi-scale contextual information. The network structure of DeepLabV3+ is shown in Fig. 7. As shown in the figure, the feature map is extracted by the backbone network in the encoding process, and the atrous convolution is applied correspondingly. DeepLabV3+ augments the ASPP module, which probes convolutional features at multiple scales by applying atrous convolution with four different rates and one average pooling. In the encoder stage, the features after ASPP are up-sampled by a factor of 4 and then concatenated with the corresponding low-level features. After the concatenation, a 3×3 convolution and an up-sampling are applied to get the segmentation result. Modified Xception is adopted as the backbone network in DeepLabV3+ to improve its performance further in semantic segmentation tasks.

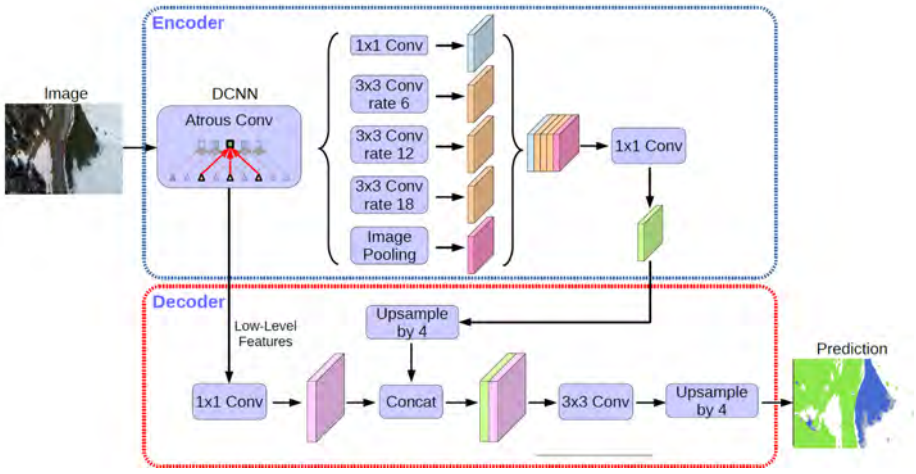


Figure 7: DeepLabV3+ Architecture

4.2. AdaptSegNet

AdaptSegNet [3] is an adversarial-based domain adaptation method for semantic segmentation tasks. This method takes advantage of spatial similarity shared between source and target domains. Even if images from two domains have different appearances, they share a significant amount of local context and spatial information. Based on this intuition, this method deals with the pixel-wise semantic segmentation task in the output space, directly making the predicted

label distribution of source and target domains close to each other. As shown in Fig. 8, the AdapSegNet model consists of two parts: a segmentation network (DeepLab) and a discriminator (based on generative adversarial model (GAN)). First, both source and target images are passed to the segmentation network to obtain predictions. Then in the domain adaptation module, the semantic segmentation loss is computed in the source domain with the annotated ground truth of source images. Besides, a discriminator is incorporated here to distinguish whether the prediction is from the source or the target domain. Then the adversarial loss is computed on the target prediction and backpropagated to the segmentation network. This encourages the segmentation network to generate similar predictions in source and target domains.

To further enhance performance, the adversarial model is reconstructed to be multi-level to perform domain adaptation at different feature levels effectively. However, it is only extended to two levels (output space and the previous layer) in experiments due to computational limitations.

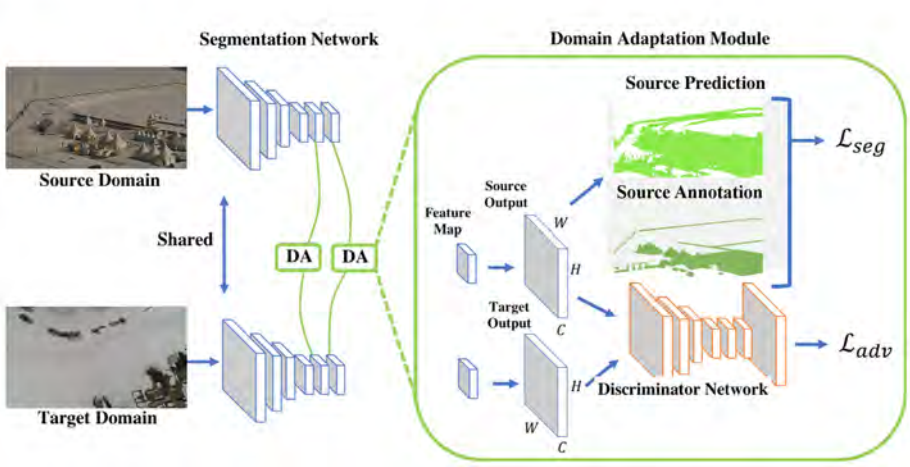


Figure 8: AdaptSegNet Architecture

4.3. FDA

FDA [4] does a straightforward Fast Fourier Transformation (FFT) to adapt the style between source and target domains. As shown in Fig. 9, before

training on the source dataset, the FFT of each input image (source and target) is computed. Then the low-level frequencies of the source image, which play a less important role in high-level image semantics, are replaced by those of the target image. Then the source image is reconstituted via the inverse FFT (iFFT). This method helps the network to generalize across different spectrum characteristics to predict the unchanged categories. Transformed source images together with original annotations are used for training. FDA requires selecting one free parameter, the size of the spectral neighborhood to be swapped (green square in Fig. 9). Increasing β will swap the source image more to the target side, thus decrease the domain gap. Nevertheless, this could also introduce artifacts, which impairs information in the images.

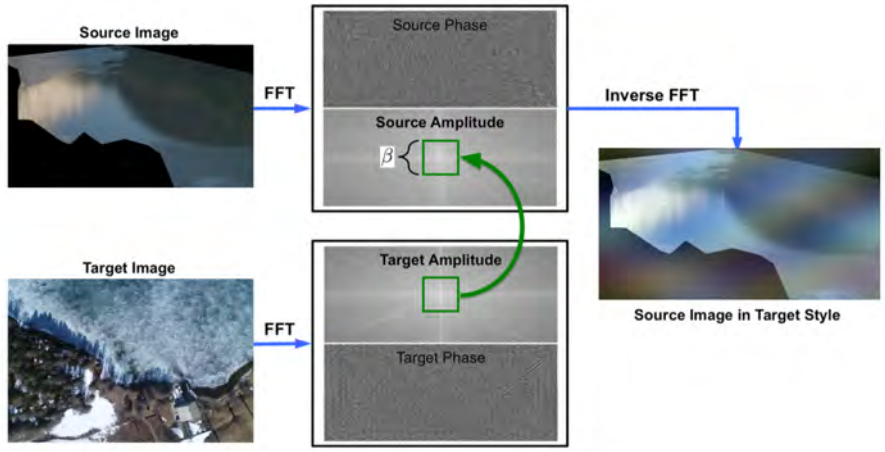


Figure 9: Illustration of FDA Transformation

4.4. CMD

CMD [5] tries to learn domain-invariant representations by mapping the difference of latent feature representations between source and target domains. The basic idea is shown in Fig. 10. $A_H(X_S, \theta)$ represents the source hidden activations, and $A_H(X_T, \theta)$ represents the target hidden activations. In addition to common empirical expectations (denoted by $\mathbf{E}(l(\theta, X_S, Y_S))$), a regularizer (denoted by $d(A_H(\theta, X_S), A_H(\theta, X_T))$) which maps the difference of activations

on each hidden layer is added to the loss function. λ is a weighting parameter representing the ratio between those two parts of the loss function. The goal is to minimize the total loss. By adding this regularizer, the similarity between the latent network representations of source and target domains is enforced.

In the CMD method, the domain regularizer is defined as CMD regularizer. First, the CMD metric is computed as:

$$CMD(p,q) = \frac{1}{|b-a|}\|\mathbb{E}(X) - \mathbb{E}(Y)\|_2 + \sum_{k=2}^{\infty} \frac{1}{|b-a|^k} \|c_k(X) - c_k(Y)\|_2 \quad (1)$$

Where X and Y are bounded random vectors independent and identically distributed from two probability distributions p and q on the compact interval $[a,b]^N$. $\mathbb{E}(X)$ is the expectation of X, and:

$$c_k(X) = \left(\mathbb{E}\left(\prod_{i=1}^{N} (X_i - \mathbb{E}(X_i))^{r_i} \right) \right)_{r_1+\ldots+r_N=k, \quad r_1,\ldots,r_n \geq 0} \quad (2)$$

is the central moment vector of order k. The final central moment discrepancy regularizer is defined as an empirical estimate of the CMD metric. Only the central moments that correspond to the marginal distributions are computed. The number of central moments is limited by a new parameter K, and the expectation is sampled by the empirical expectation [5].

In our experiment, we implement this loss function under the DeepLabV3+ model. The backbone network is chosen to be ResNet101. According to the architecture of ResNet, we set $K = 3$, which means we compute the CMD loss after the last convolutional layer of each block (except for the last layer of prediction). The activation function of each computed hidden layer is Sigmoid. The total loss is a weighted sum of cross-entropy loss (for semantic segmentation task) and CMD loss. CMD loss ratio, which determines the weight between those two losses, is tuned together with DeepLabV3+ parameters.

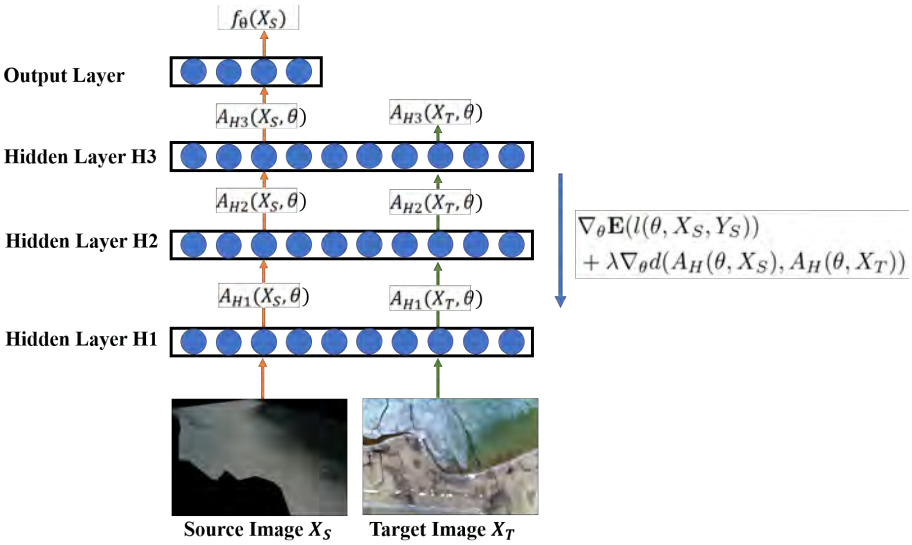


Figure 10: Architecture Sketch of a Basic Network with Backpropagation of an Additional Loss Function Mapping the Distance between Source and Target Hidden Activation Layers

5. Result

5.1. Pre-training on Photi-Lakeice Dataset

To pre-train the DeepLabV3+ model on the source dataset, we refer to the parameter settings in the work of [1] and tune around a bit to find the best pre-trained model with both decent accuracy and transferability.

The network is implemented in TensorFlow, adopting the open-source code on `https://github.com/tensorflow/models/tree/master/research/deeplab/`. All models are trained on 75% of sourceimages and evaluated on the remaining 25% after training. The training starts from the initial model pre-trained on the PASCAL dataset (provided in the code repository). According to the previous work in [1], the crop size of training is fixed to 321×321. Per-class losses are balanced by re-weighting the cross-entropy loss with the inverse (relative) frequencies in the training set. Bath size is set to 8, and atrous rates are set to [6, 12, 18] in all experiments. The learning strategy is simply stochastic gradient descent, and the learning rate is reduced according to the poly schedule. During

the training process, weights are unfrozen on each layer. The evaluation of the networks after training is always run at full image resolution without any cropping. To further test out the best transferable parameter setting in our case, we experiment on different combinations of values of learning rate (lr) and training steps. Results are evaluated by IoU. Evaluation is shown in Table 3. As they generate decent mean Intersection-over-Union (mIoU), all those combinations listed in the table are compared by direct prediction in section 5.2.

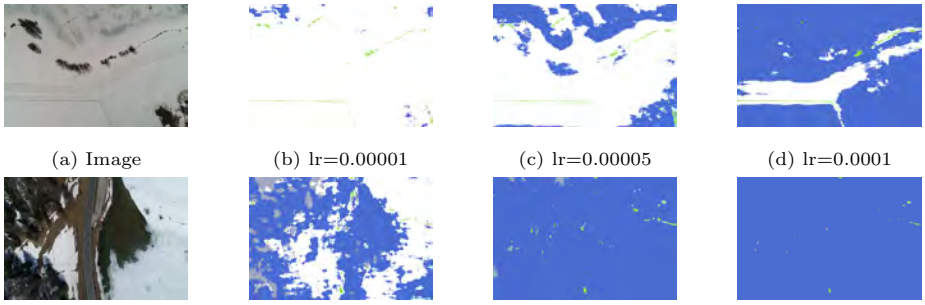
Table 3: Evaluated IoU Values of DeepLabV3+ Model Trained on Photi-Lakeice Webcam Dataset

Parameters		IoU				
lr	training steps	snow	water	clutter	ice	mIoU
0.00001	50,000	0.91	0.98	0.62	0.93	0.86
0.00001	100,000	0.92	0.99	0.67	0.92	0.88
0.00001	200,000	0.94	0.99	0.70	0.93	0.89
0.00005	100,000	0.94	0.99	0.72	0.94	0.90
0.00005	150,000	0.94	0.99	0.74	0.95	0.91
0.0001	100,000	0.95	0.99	0.75	0.95	0.91

5.2. Direct Prediction

As all the pre-trained results are quite good, we first try to directly using these models to predict on the UAV dataset and check the visualization prediction results. Comparisons of prediction results using different learning rates are shown in Fig. 11. As shown in the figure, with a larger learning rate of 0.0001 and 0.00005, the transferability of the model is comparably weak, although the mIoU tested on the source domain is comparably higher in Table 3. Those two pre-trained models tend to predict pixels as the water category, although the weight has already been adjusted. The prediction result with a learning rate of 0.00001 does not give reasonable predictions as well, but visually looks a bit nicer, at least giving reasonable patterns although without the correct pixel labels. So we decide to use this pre-trained model for our later experiments.

The final full parameter settings are listed in Table 4.



(a) Image (b) lr=0.00001 (c) lr=0.00005 (d) lr=0.0001

(e) Image (f) lr=0.00001 (g) lr=0.00005 (h) lr=0.0001

Figure 11: Comparison of Direct Predictions on UAV Dataset Using Models Pre-trained on Photi-Lakeice Dataset with Different Learning Rates ($training\_steps = 100,000$)

Table 4: Final Parameter Settings of Pre-training DeepLabV3+ on Photi-Lakeice Webcam Dataset

Parameter	Value
model_variant	"xception_65"
label_weights	[1.0, 1.1, 8.7, 4.2]
atrous_rates	[6, 12 18]
output_stride	16
decoder_output_stride	4
num_classes	4
train_crop_size	[321, 321]
train_batch_size	8
training_number_of_steps	100,000
fine_tune_batch_norm	false
base_learning_rat	0.00001
learning_policy	"poly"

5.3. Supervised Method

As shown in section 5.2, the direct prediction does not work, which is expected due to the large domain gap discussed in the data section. In this

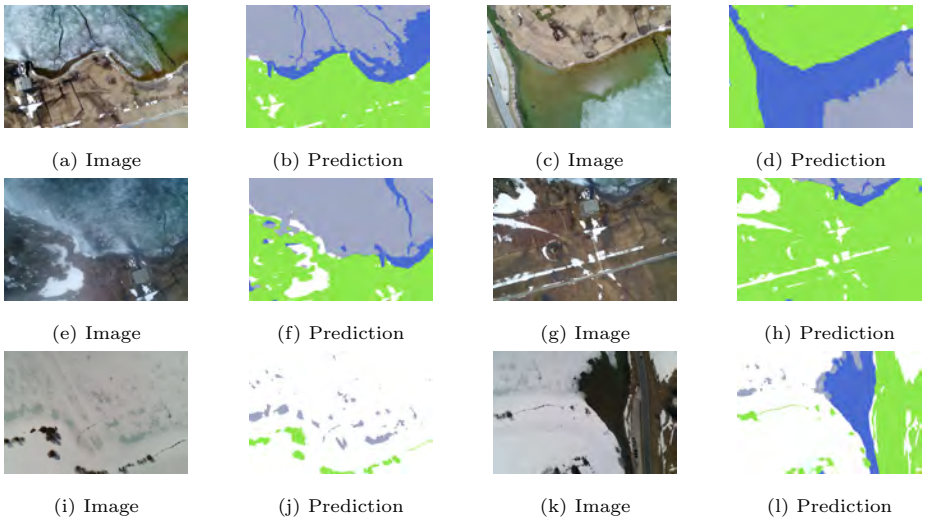
section, we fine-tune the DeeplabV3+ model based on the pre-trained model of Photi-Lakeice webcam dataset, as discussed in section 5.2. 77 images are annotated manually as pixel-wise ground truth. In this experiment, we use 10 of them for training and the rest 67 for evaluation. After several rounds of experiments, the best training parameters for fine-tuning are set as following: $training\_number\_of\_steps = 10,000, base\_learning\_rate = 0.00001$. Instead of stochastic gradient descent, the learning strategy is set as Adam [16] to help with faster convergence. All the other parameters are set the same as discussed in section 5.2. Fig. 12 shows the visualized prediction results. Table 5 and Table 6 shows the quantitative analysis result. As shown in the results, the fine-tuned model performs well with 0.88 mIoU and 0.95 pixel-accuracy.

Table 5: IoU Values Evaluated on 67 UAV Images after Fine-tuning DeepLabV3+ on 10 Labeled UAV Training Images

Snow	Water	Clutter	Ice	mIoU
0.90	0.81	0.91	0.90	0.88

Table 6: Confusion Matrix Evaluated on 67 UAV Images after Fine-tuning DeepLabV3+ on 10 Labeled UAV Training Images

Class	Snow	Water	Clutter	Ice
Snow	444929601	801341	12849142	4341937
Water	4203197	104806177	1432355	4233559
Clutter	16046070	9355192	460408907	3765463
Ice	9966746	4764092	287251	255253322



(a) Image (b) Prediction (c) Image (d) Prediction

(e) Image (f) Prediction (g) Image (h) Prediction

(i) Image (j) Prediction (k) Image (l) Prediction

Figure 12: Prediction Results of Supervised Method after Fine-tuning DeepLabV3+ on 10 Labelled UAV Training Images

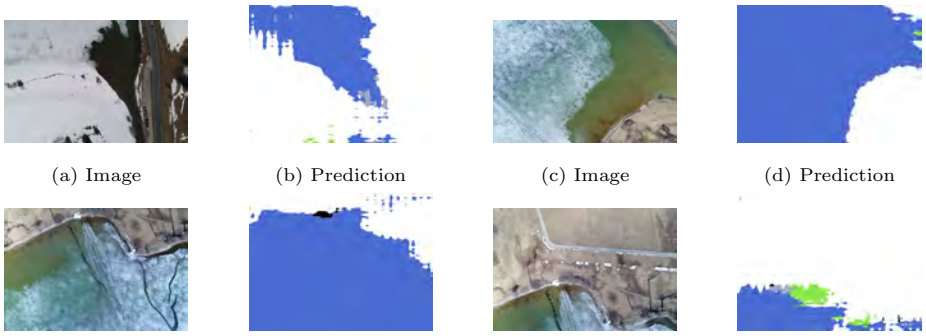
5.4. Unsupervised Method

DeepLabv3+ still needs manual work to label a certain number of UAV images and fine-tuning on the new domain to generate meaningful prediction results. Although only 10 high-resolution UAV images with annotations are adequate for fine-tuning, the labeling work is tedious and time-consuming. Therefore, we would like to try to incorporate domain adaption in order to generate better direct prediction results without supervision in the target dataset.

5.4.1. AdaptSegNet

The method is implemented in PyTorch, adopting the open-source code on `https://github.com/wasidennis/AdaptSegNet`. As the semantic segmentation network of AdaptSegNet is also DeepLab, we refer to the best parameter settings in section 5.1 & 5.2 and tune around these parameters together with parameters for the discriminator (refer to the recommended setting in the original paper [3]). Unfortunately, after several rounds of experiments, we do not see any meaningful prediction results that could show some possible directions for improvement. One set of experiment results is shown in Fig. 13. After the

failure, we tried another strategy, which first excludes the discriminator from the whole structure, and only trains the semantic segmentation network until it becomes stable. At this point, we add the discriminator to start learning from the unlabeled target images. However, this does not bring much difference to the experiment result.



(a) Image (b) Prediction (c) Image (d) Prediction

(e) Image (f) Prediction (g) Image (h) Prediction

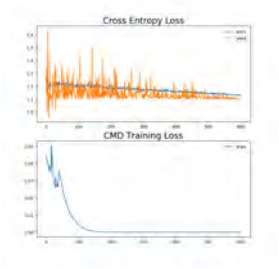
Figure 13: Prediction Results of AdaptSegNet Trained on Full Labeled Source Dataset and Unlabeled Target Dataset

5.4.2. CMD

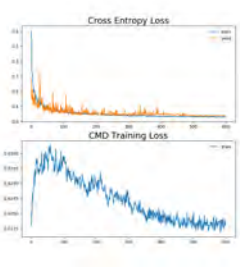
In the CMD experiment, we tune around two crucial parameters: the learning rate and the CMD loss ratio. Fig. 14 shows the comparison of the training loss curves of different parameter settings. Comparing those images, we could tell that the CMD training loss does not converge with too small CMD loss ratio, which decides how much the target dataset contributes to the training process. A larger learning rate also increases the weight of CMD loss in each epoch. However, as the learning rate is set for the whole network, including backpropagation of cross-entropy loss, a too large learning rate also leads to divergence, as shown in Fig. 14(a). Fig. 15 shows the training process of one set of experiment with comparably promising parameter settings.

Fig. 16 shows the corresponding prediction result. In some patterns, CMD does help the network to learn, compared to direct prediction without incorporating CMD Loss. But still, it fails to close the large gap between the source and

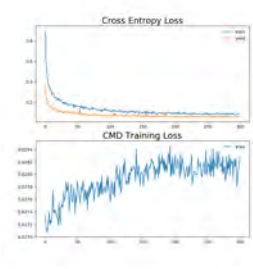
target domain. As shown in Fig. 15, although the CMD loss ratio has been set to 1,000, the cross-entropy loss still converges to a very small value. This indicates that the network still learns most of its knowledge from the labeled source dataset, as it is much easier comparing to mapping such a large domain gap.



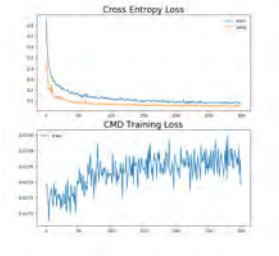
(a) Learning_Rate=0.01, CMD_Loss_Ratio=100.0



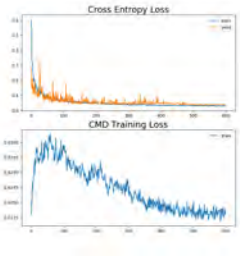
(b) Learning_Rate=0.0001, CMD_Loss_Ratio=100.0



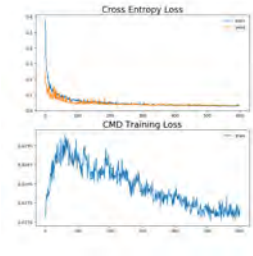
(c) Learning_Rate=0.00001, CMD_Loss_Ratio=100.0



(d) Learning_Rate=0.0001, CMD_Loss_Ratio=10.0

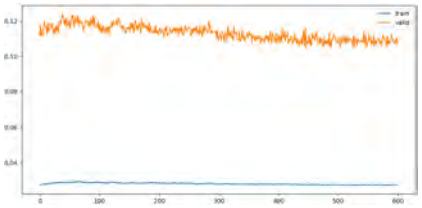


(e) Learning_Rate=0.0001, CMD_Loss_Ratio=100.0

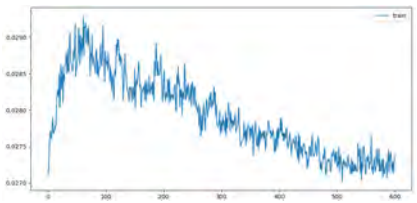


(f) Learning_Rate=0.0001, CMD_Loss_Ratio=1000.0

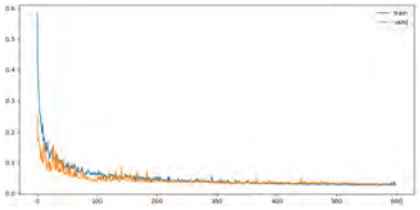
Figure 14: Training Curves of CMD with Different Parameter Settings



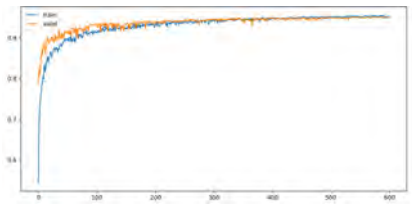
(a) CMD Loss



(b) CMD Training Loss



(c) Cross-Entropy Loss



(d) mIoU of Source Dataset

Figure 15: Training Process of CMD, Parameter Setting: Learning_Rate= 0.0001, CMD_Loss_Ratio = 1000.0

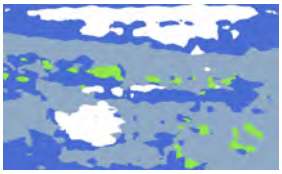
(a) Image

(b) Direct Prediction

(c) CMD Prediction

(d) Image

(e) Direct Prediction



(f) CMD Prediction

(g) Image

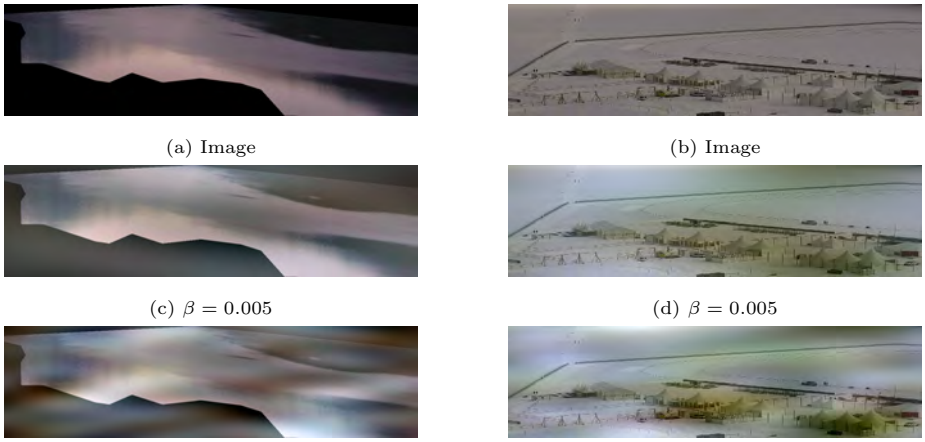
(h) Direct Prediction

(i) CMD Prediction

Figure 16: Prediction Results of CMD Compared with Direct Prediction Results (discussed in section 5.2), Parameter Setting: Learning_Rate= 0.0001, CMD_Loss_Ratio = 1000.0

5.4.3. FDA

For FDA, the crucial parameter is β, which determines how much the source image will be swapped to the target side. A large β value could cause artifacts in transformed images while a small value does not transfer enough information from the target side. As shown in Fig. 17, we could see artifacts in transformed images, even with small β values. This might be due to the large variance in the richness of content between the source and target dataset, as the former only focuses on the lake surface. We transform source images with a wide range of β values and train DeepLabV3+ on those transformed source images. After that, we use these pre-trained models to predict on the target dataset directly and compare the results with the prediction in section 5.2. One set of experiment results with $\beta = 0.008$ is shown in Fig. 18. As shown in this figure, FDA does not help generate meaningful prediction results.

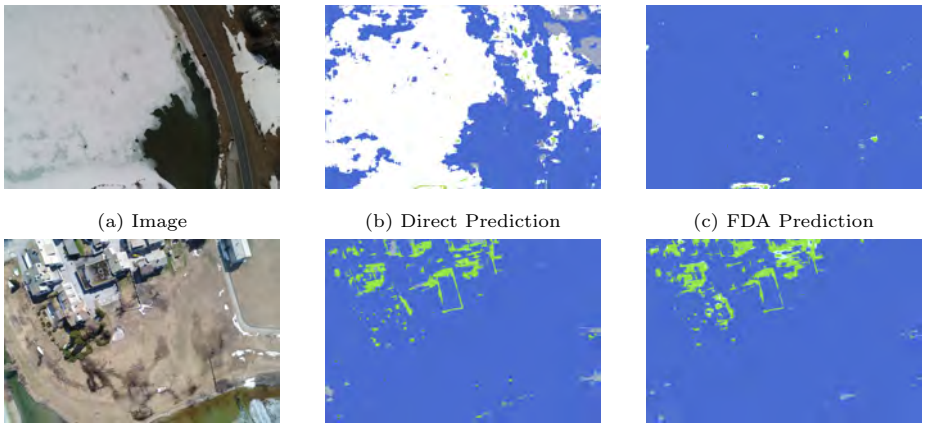


(a) Image (b) Image

(c) $\beta = 0.005$ (d) $\beta = 0.005$

(e) $\beta = 0.015$ (f) $\beta = 0.015$

Figure 17: Source Images Transformed to the Target Side by FDA with Different β values



(a) Image (b) Direct Prediction (c) FDA Prediction

(d) Image (e) Direct Prediction (f) FDA Prediction

Figure 18: Comparison of Prediction Results of FDA ($\beta = 0.008$) with Direct Prediction (discussed in 5.2)

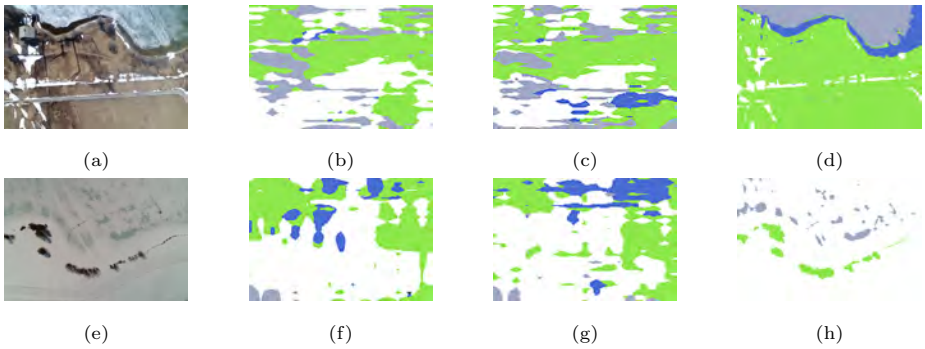
5.5. Semi-supervised Method

As all those domain adaptation methods do not generate satisfying results fully unsupervised, we decide to try those methods again in a semi-supervised way by incorporating fewer training images than the supervised method. Although we achieve decent segmentation results with only ten labeled target images, we would like to see if those domain adaptation methods could help reduce the training images that we need.

5.5.1. AdaptSegNet

Our first try in this experiment is to add 4 labeled UAV images into the source dataset. All the experiments show no clear difference from the unsupervised method. Therefore, we also try the second strategy. This time, we not only pre-train the semantic segmentation network part on the labeled source images but also fine-tune it on 4 labeled target images before adding the discriminator. By doing this, we would like to see whether the discriminator helps improve the trained network by checking prediction results during the training process. One set of experiment results is shown in Fig. 19. As shown in the figure, after training for considerably sufficient steps, the backpropagated loss from the

discriminator impairs the prediction on the target dataset and does not help improving when we continue training.



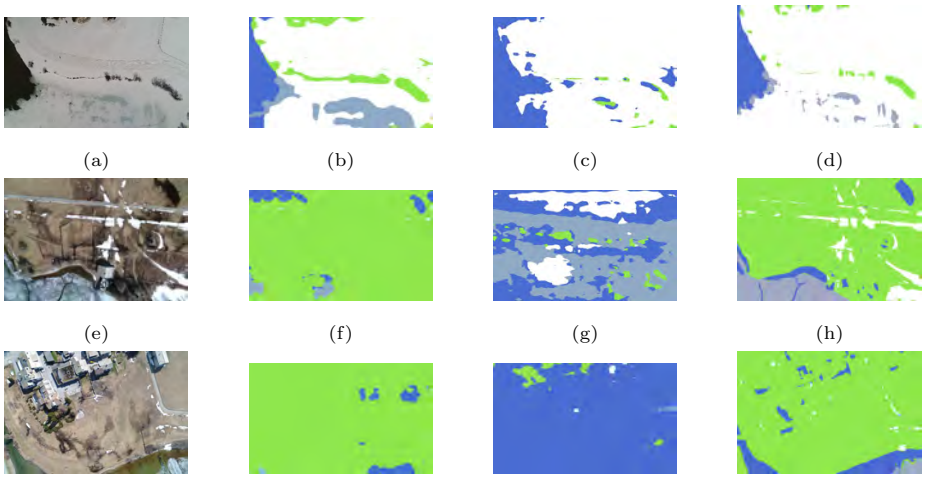
(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 19: Comparison of Prediction Results of AdaptSegNet Incorporating 4 Labeled Target Images, with Prediction Results of Fine-tuning DeepLabV3+ with 4 Labeled Target Images (a)(e): Original Images; (b)(f): AdaptSegNet Prediction Incorporating 4 Labeled Target Images (training for 15,000 steps); (c)(g): AdaptSegNet Prediction Incorporating 4 Labeled Target Images (training for 30,000 steps); (d)(h): Prediction of Fine-tuning DeepLabV3+ with 4 Labeled UAV Images

5.5.2. CMD

In this experiment, we add 4 labeled UAV images into the source dataset. Training parameters are the same as the experiment shown in section 5.4.2. The prediction results are shown in Fig. 20. Although the prediction results are improved compared to unsupervised CMD, it does not overcome the result of fine-tuning DeepLabV3+ with the same labeled target images. Checking the training process in Fig. 21, we could tell that trying to learn more information of target images impairs the prediction on the source dataset a lot.



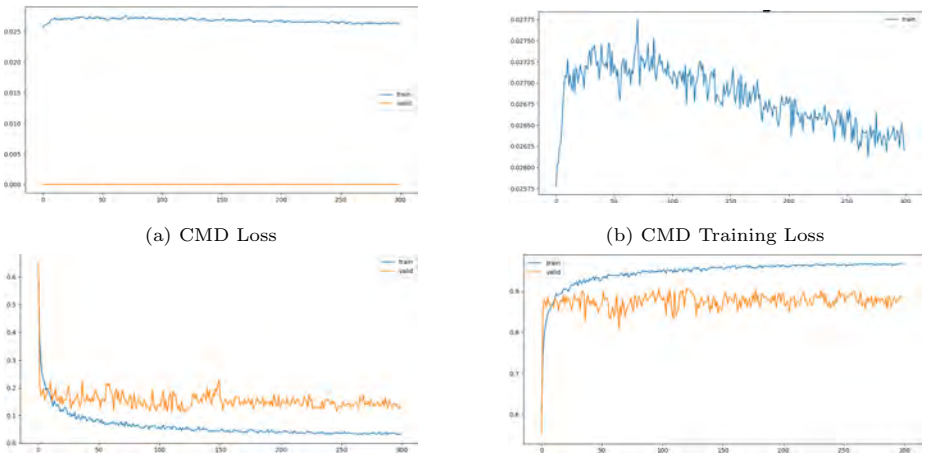
(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 20: Comparison of Prediction Results of CMD Incorporating 4 Labeled Target Images and Unsupervised CMD, with Prediction Results of Fine-tuning DeepLabV3+ with 4 Labeled Target Images

(a)(e)(i): Original Images; (b)(f)(j): CMD Semi-supervised Prediction (with 4 labeled UAV images); (c)(g)(k): CMD Unsupervised Prediction; (d)(h)(l): Prediction of Fine-tuning DeepLabV3+ with 4 Labeled UAV Images



(a) CMD Loss

(b) CMD Training Loss

(c) Cross-Entropy Loss

(d) mIoU of Source Dataset

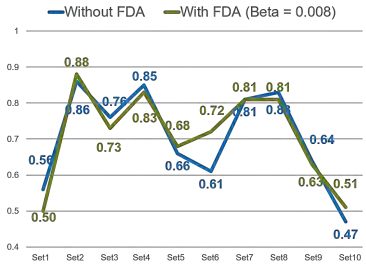
Figure 21: Training Process of Semi-supervised CMD, Parameter Setting: Learning_Rate= 0.0001, CMD_Loss_Ratio = 1000.0

5.5.3. FDA

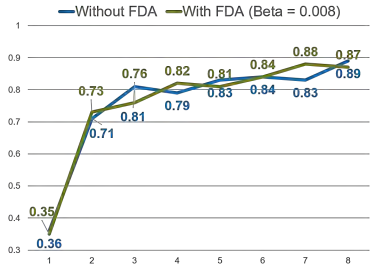
In all the experiments in this section, with the FDA method, we first pre-train the DeepLabV3+ model on source images transformed by FDA. Then we fine-tune the pre-trained model with some labeled target images. In comparison, we also run corresponding experiments without FDA, which means that we pre-train the model on the original source images and then train on the same target images.

First, we manually picked two sets of 4 different labeled terget images and experiment on those two sets with different β values for FDA. According to the experiment results, we fix the β value to be 0.008, which generates the best prediction results on the target dataset on average.

During the experiment, we notice that whether FDA helps improve the prediction results not only relates to the β value but also links with the choice of training target images, as prediction results vary between these two sets. So, we run some experiments, randomly picking four labeled target images for fine-tuning, and compare those results. The result is shown in Fig. 22(a). As shown in the figure, FDA only helps in 4 cases out of 10. Visualized prediction results of set1 and set6 are shown in Fig. 23 and Fig. 24. As shown in these two sets of predictions, FDA shows its potential to improve prediction results. However, it is hard to determine it due to the limitation of our target dataset. As shown in Fig. 22(b), only 4 labeled images help generate satisfying mIoU evaluated on target images. As the target dataset lacks variety, a proper choice of images plays a much more significant role. A larger dataset with more variance in images is needed for further experiments.

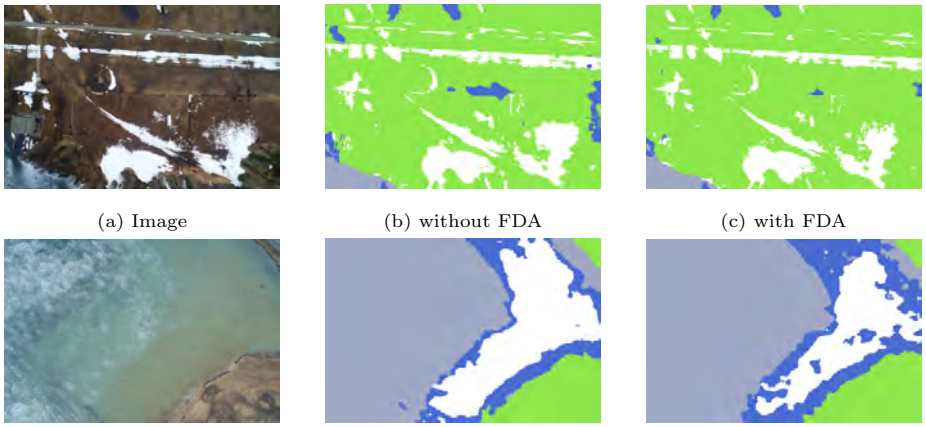


(a) Comparison of mIoU Evaluated on 67 Labeled Target Images (model trained on source images (original/FDA_transformed) and different sets of randomly picked 4 labeled target images)



(b) Comparison of mIoU Evaluated on 67 Labeled Target Images (model trained on source images (original/FDA_transformed) and a random number of labeled target images)

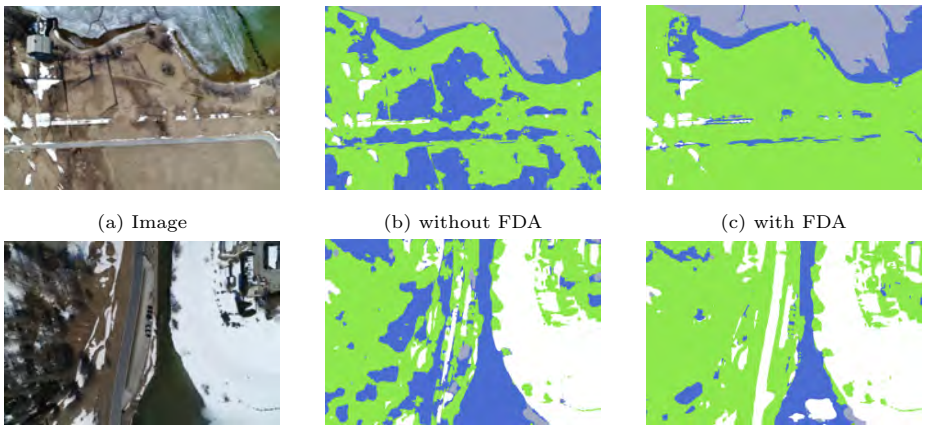
Figure 22: mIoU of FDA Method Evaluated on 67 Labeled Target Images, $\beta = 0.08$



(a) Image (b) without FDA (c) with FDA

(d) Image (e) without FDA (f) with FDA

Figure 23: Comparison of Prediction Results without/with FDA transformation (model trained on source images (original/FDA_transformed) and 4 labeled target images (set 1 in Fig. 22 (a)), $\beta = 0.08$



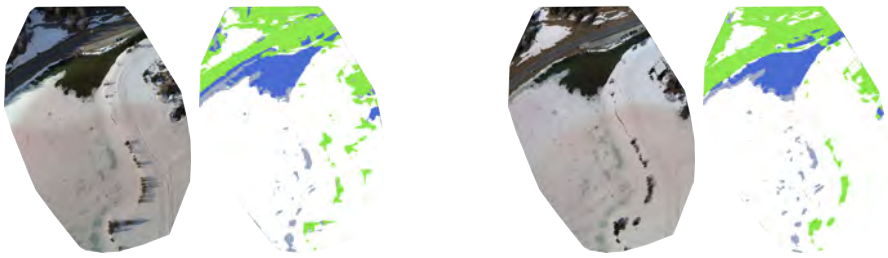
(a) Image (b) without FDA (c) with FDA

(d) Image (e) without FDA (f) with FDA

Figure 24: Comparison of Prediction Results without/with FDA transformation (model trained on source images (original/FDA_transformed) and 4 labeled target images (set 6 in Fig. 22 (a)), $\beta = 0.08$

5.6. Geo-referenced Result

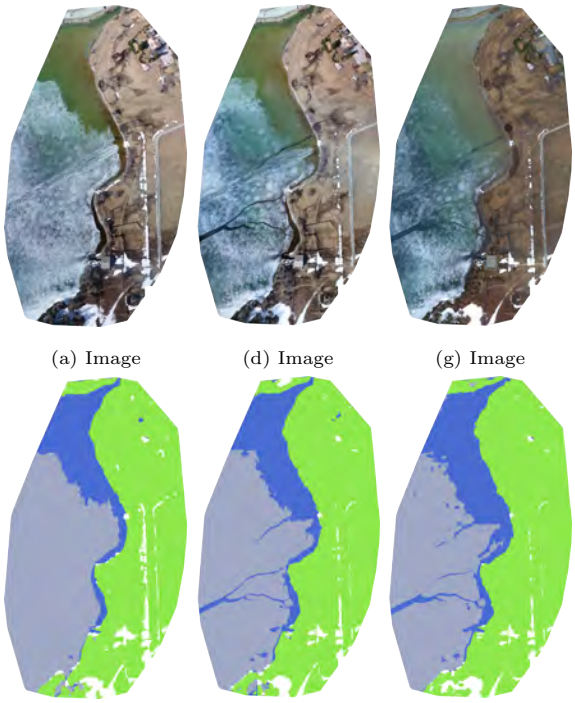
To effectively monitor lake ice of Sils, we do our analysis based on the segmentation results of the supervised method, which fine-tunes DeepLabV3+ model with 10 labeled UAV images (as discussed in section 5.3). The segmentation results of geo-referenced mosaic images of the whole monitored area is shown in Fig. 25 and Fig. 26. Each year's images are clipped according to their overlapped region, so they are good forcomparing the percentage of each class. Changes in the class percentage in 2019 are shown in Fig. 27. We could see from the figure a decrease in the percentage of ice from 0.38 to 0.36 and then to 0.33, indicating ice melting of the lake. We could also see the percentage of snow decreasing while the percentage of water and clutter increasing. This corresponds with the change of ice and proves that our segmentation results are valid.



(a) Image (b) Prediction (d) Image (e) Prediction

(c) 18/4/2018 (f) 19/4/2018

Figure 25: Predicted Segmentation Results of Geo-referenced Images of 2018



(a) Image (d) Image (g) Image

(b) Prediction (e) Prediction (h) Prediction

(c) 09/5/2019 (f) 10/5/2019 (i) 11/5/2019

Figure 26: Predicted Segmentation Results of Geo-referenced Images of 2019

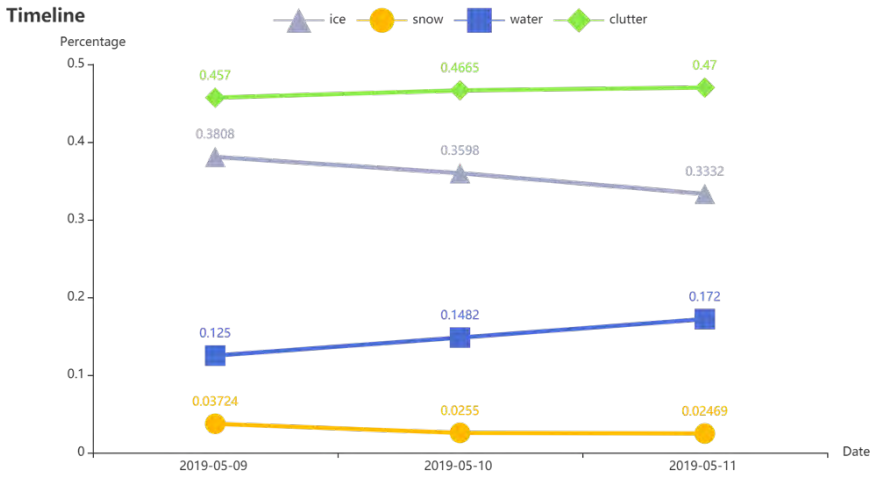


Figure 27: Changing Trend of Percentage of Each Class Analyzed with Segmentation Results of 2019 Mosaic Images

6. Conclusion

In this project, we succeed in the monitoring task through fine-tuning DeepLabV3+ using 10 labeled UAV images. The result is accurate enough to demonstrate a reasonable change in the percentage of each class to help with monitoring lake ice.

According to the experiment results discussed in the report, we are unable to close the domain gap with those up-to-date domain adaptation methods in an unsupervised way. Therefore, this might indicate some limitations of current domain adaptation methods. Although these methods help to improve semantic segmentation prediction results in their own experiments, they might not work in large domain gap scenarios, especially viewpoint changes. Most domain adaptation papers test their method with GTA5 [17] or SYNTHIA [18] as the source domain and Cityscapes [19] as the target domain. Those datasets share very similar viewpoint and content, as illustrated in Fig. 28. Additionaly, they need a comparably larger amount of data for training. GTA5 contains 24,966 annotated images, and SYTHIA contains 9,400. 2,975 images are available in Cityscapes for adaptation. Notably, the semi-supervised method

with FDA looks promising, as it helps improve the prediction result in certain cases. However, no concret conclusion regarding those unsupervised and semi-supervised methods could be drawn due to the limitation of our own dataset. The UAV dataset only contains 186 images over 5 days, and those drone images have an overlap of approximately 70%. A collection of part of the UAV images is dipicted in Fig. 29. We do not have much variety in the UAV dataset, which largely limits the adaptation. This also makes the choice of appropriate images a dominant element as opposed to which method we choose. Further experimentation with a more representative UAV dataset is required to validate all those hypotheses.

Besides strengthening the target dataset, possible future work could aim at exploring a better domain adaptation methodology incorporating viewpoint changes, which is shown to be the primary challenge from our experiments. The domain gap between our source and target dataset is too large to contain enough local structure similarities. Specifically, considering viewpoint changes separately might help identify a better solution.

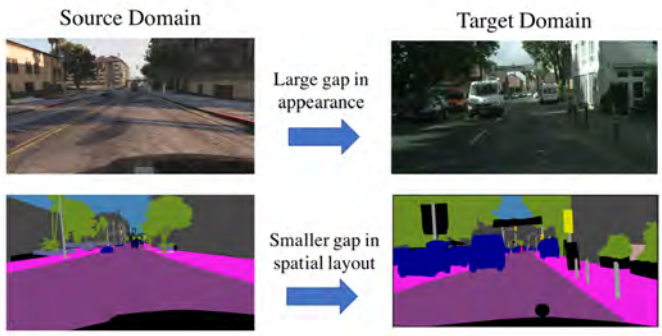


Figure 28: Sample Images of Dataset Experimented in AdaptSegNet[3]

Figure 29: Part of UAV Dataset

References

[1] R. Prabha, M. Tom, M. Rothermel, E. Baltsavias, L. Leal-Taixé, Lake ice monitoring with webcams and crowd-sourced images, ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences V-2-2020 (2020) 549–556. doi:10.5194/isprs-annals-V-2-2020-549-2020.

[2] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: ECCV, 2018.

[3] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, M. Chandraker, Learning to adapt structured output space for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[4] Y. Yang, S. Soatto, Fda: Fourier domain adaptation for semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[5] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, S. Saminger-Platz, Central moment discrepancy (cmd) for domain-invariant representation learning (2019). arXiv:1702.08811.

[6] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (4) (2017) 640–651. doi:10.1109/TPAMI.2016.2572683.

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Semantic image segmentation with deep convolutional nets and fully connected crfs (2016). arXiv:1412.7062.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs (2017). arXiv:1606.00915.

[9] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam, Rethinking atrous convolution for semantic image segmentation (2017). arXiv:1706.05587.

[10] F. Sultana, A. Sufian, P. Dutta, Evolution of image segmentation using deep convolutional neural network: A survey, Knowledge-Based Systems 201-202 (2020) 106062. doi:https://doi.org/10.1016/j.knosys.2020.106062.
URL http://www.sciencedirect.com/science/article/pii/S0950705120303464

[11] M. Wang, W. Deng, Deep visual domain adaptation: A survey (2018). arXiv:1802.03601.

[12] H. Wang, T. Shen, W. Zhang, L. Duan, T. Mei, Classes matter: A fine-grained adversarial approach to cross-domain semantic segmentation, in: The European Conference on Computer Vision (ECCV), 2020.

[13] S. Lee, D. Kim, N. Kim, S.-G. Jeong, Drop to adapt: Learning discriminative features for unsupervised domain adaptation, in: The IEEE International Conference on Computer Vision (ICCV), 2019.

[14] M. Long, H. Zhu, J. Wang, M. I. Jordan, Deep transfer learning with joint adaptation networks, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, Vol. 70 of Proceedings of Machine Learning Research, PMLR, International Convention Centre, Sydney, Australia, 2017, pp. 2208–2217.
URL http://proceedings.mlr.press/v70/long17a.html

[15] Y. Zhu, X. Hu, Y. Zhang, P. Li, Semi-supervised representation learning: Transfer learning with manifold regularized auto-encoders, in: 2018 IEEE International Conference on Big Knowledge (ICBK), 2018, pp. 83–90. doi:10.1109/ICBK.2018.00019.

[16] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization (2017). arXiv:1412.6980.

[17] S. R. Richter, V. Vineet, S. Roth, V. Koltun, Playing for data: Ground truth from computer games, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), European Conference on Computer Vision (ECCV), Vol. 9906 of LNCS, Springer International Publishing, 2016, pp. 102–118.

[18] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, A. M. Lopez, The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3234–3243. doi:10.1109/CVPR.2016.352.

[19] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

7. Appendix: Experiments of Thermal Images

This appendix shows some initial experiment results of thermal images. We still adopt the pre-trained DeepLabV3+ model of Photi-Lakeice dataset and fine-tune with labeled thermal images. We manually pick 2,335 thermal images with meaningful content and annotate 15 images for training. Prediction results are shown in Fig. 30

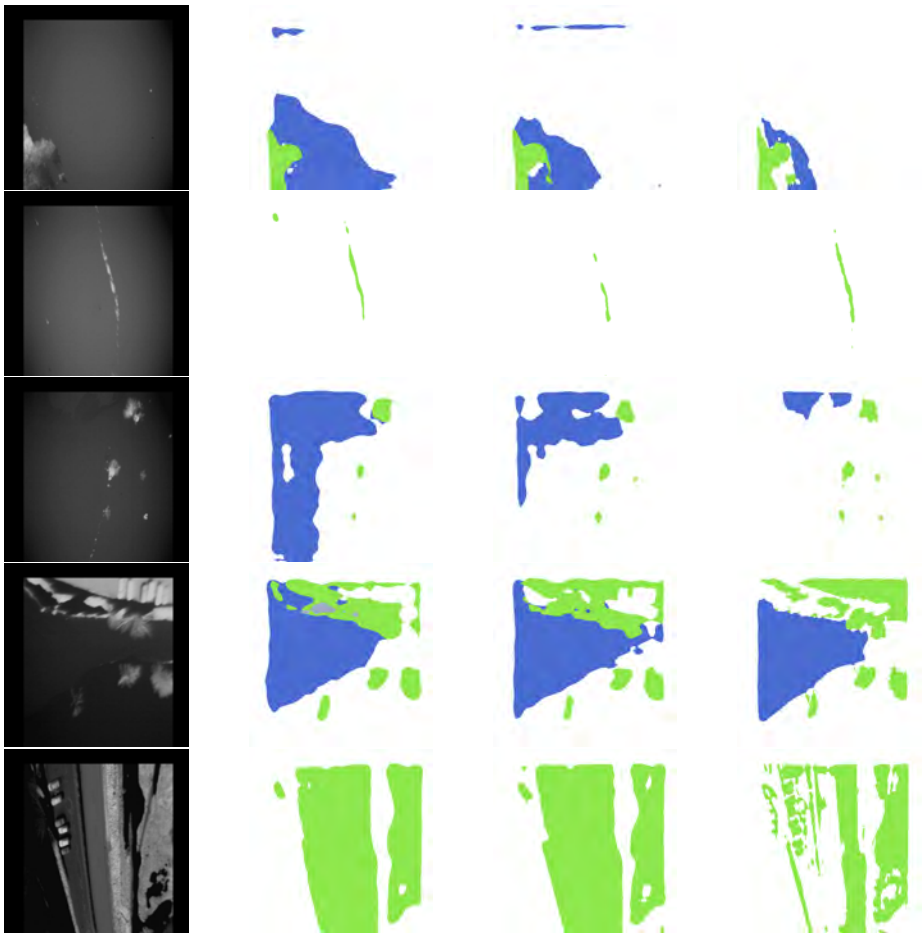


Figure 30: Comparison of Prediction Results of Thermal Images Trained on DeepLabV3+ with Different Parameter Settings, Using 15 Labeled Thermal Images

First Column: Thermal Images;

Second Column: Predictions with Parameter Setting of $base\_learning\_rate = 0.00001, learning\_policy = "poly", training\_number\_of\_steps = 10,000$;

Third Column: Predictions with Parameter Setting of $base\_learning\_rate = 0.00001, learning\_policy = "poly", training\_number\_of\_steps = 100,000$;

Fourth Column: Predictions with Parameter Setting of $optimizer =' adam', adam\_learning\_rate = 0.00001, training\_number\_of\_steps = 10,000$;