

Deep Ice Layer Tracking and Thickness Estimation using Fully Convolutional Networks

Debvrat Varshney, Maryam Rahnemoonfar*, Masoud Yari
Computer Vision and Remote Sensing Laboratory
University of Maryland Baltimore County
Baltimore, MD, USA
dvarshney, maryam*, yari@umbc.edu
*Corresponding Author

John Paden
Center for Remote Sensing of Ice Sheets (CREGIS)
University of Kansas Lawrence
Lawrence, KS, USA
paden@ku.edu

Abstract—Global warming is rapidly reducing glaciers and ice sheets across the world. Real time assessment of this reduction is required so as to monitor its global climatic impact. In this paper, we introduce a novel way of estimating the thickness of each internal ice layer using Snow Radar images and Fully Convolutional Networks. The estimated thickness can be used to understand snow accumulation each year. To understand the depth and structure of each internal ice layer, we perform multi-class semantic segmentation on radar images, which hasn't been performed before. As the radar images lack good training labels, we carry out a pre-processing technique to get a clean set of labels. After detecting each ice layer uniquely, we calculate its thickness and compare it with the processed ground truth. This is the first time that each ice layer is detected separately and its thickness calculated through automated techniques. Through this procedure we were able to estimate the ice-layer thicknesses within a Mean Absolute Error of approximately 3.6 pixels. Such a Deep Learning based method can be used with ever-increasing datasets to make accurate assessments for cryospheric studies.

Index Terms—Ice Layer Thickness, Semantic Segmentation, Fully Convolutional Networks, Radargrams

I. INTRODUCTION

Polar ice has been declining rapidly due to global warming. Studies suggest that sea level will increase by almost a meter at the end of this century [1]. To quantify and analyse this change, airborne Snow Radars [2] are used which help in detecting internal ice-sheet layers. These instruments give two-dimensional grayscale images (Figure 1 for example) where the horizontal axis corresponds to the flight direction of the instrument and the vertical axis corresponds to the depth in the sub-surface ice. The bright pixels correspond to signals reflected with a higher power, while the dark pixels correspond to signals reflected with a lower power [3]. By analysing the depth of these ice layers, one can assess the snow accumulation rate [4].

Snow radar imagery presents various challenges. Most of the radar images are noisy, with indistinguishable layers, especially the deeper layers. Moreover, the annotated layers are incomplete. As can be seen in Figure 1 the lower layers of the snow radar image are thinner and have lesser contrast variation compared to the upper layers. This results in training labels being available (either through manual annotations or through traditional automated techniques) only for some parts

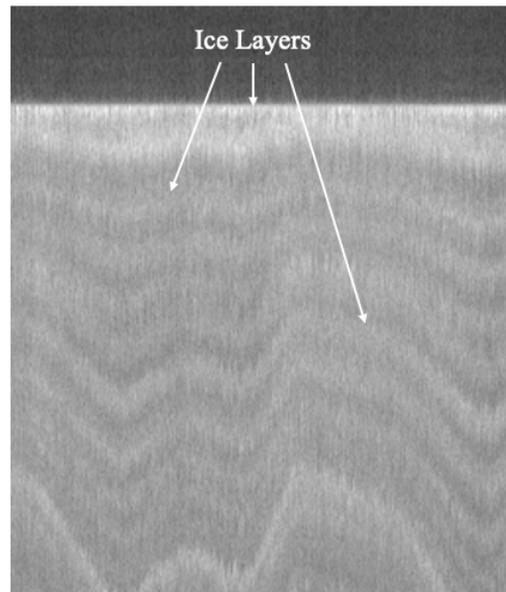


Fig. 1: A sample snow radar image.

of layers, and not for all parts of each layer. This is further explained in Section III.

Detecting each layer separately is a challenge in itself. There have been several automated techniques in the past which detect ice layers from radar data [5]–[8]. But these methods focused on a binary detection of an ice layer i.e. whether an ice layer is present at a pixel or not. The uncertainty in these binary outputs, along with the fact that an ice pixel could belong to just *any* layer makes it very hard to calculate the depth of each layer uniquely even through post-processing.

Recently, Convolutional Neural Networks (CNNs) have shown a lot of promise in understanding complex images and extracting features from them. They have especially been used for computer vision problems such as image classification, object detection, and semantic segmentation [9]–[13]. CNNs contain convolving filters which can segregate the varying shapes and textures in an image similar to how human vision works. CNNs have also been used to detect ice layers

from radar images [7], [14]. But, extracting each internal layer separately, and calculating its depth is still an issue. More recently, Fully Convolutional Networks (FCNs) [15] have been introduced for semantically segmenting an entire image. By supplying enough diverse training labels to these networks, we can get pixel-wise classification of each image. HED [16], a multi-scale FCN for edge detection, was used in [17] to detect internal ice layers in Snow Radar data. However, the authors detected ice layers in a binary format, segmenting the image into ice-layer pixels, and non-ice-layer pixels. We aim to achieve a similar output but by detecting each layer (present at different depths) uniquely.

In this paper, we use some state-of-the-art FCNs to understand each internal ice layer uniquely, and thus semantically segment Snow Radar images. We do so by first discarding the incomplete layer-labels and populating the complete layer-labels within the inter-layer regions. This will help us prepare training data where every pixel across the depth of each layer has a label. By pixel-wise annotating each layer uniquely, we can feed its specific features to an FCN for it to learn. The pixel-wise distribution of the labels in the FCN output can then help us estimate the thickness of each layer.

The rest of the paper is distributed in the following sections: Section II describes past work in ice-layer detection using radargrams, and also covers some state-of-the-art FCNs for Computer Vision in recent years. Section III describes the Snow Radar dataset that we use, and the challenges faced while detecting ice-layers from it. Section IV gives the Methodology, and highlights how we process the available training labels before feeding them to FCNs, the FCN architectures in detail, and the post-processing we carry out in order to obtain layered outputs. Section V explains the hyperparameters that we use with every architecture, and the evaluation metrics we use to assess their outputs. Section VI then quantifies the results, and also highlights qualitative results. We conclude the paper in Section VII.

II. RELATED WORK

Although there have been various automated techniques for binary detection of ice layers from radar images [5], [7], [8], [18], there is no technique to the best of our knowledge which creates a multi-class output for radar images taken over ice-sheets, especially using neural networks. In this section, we briefly describe existing techniques and highlight some state-of-the-art FCNs for semantic segmentation.

A. Ice Layer Tracking Techniques

Several automatic techniques are available for tracking the ice surface and bottom [5], [7], [8], [18]. While [18] focused on developing a hidden markov model to process planetary radargrams, [5] coupled Steger and Weiner filtering with denoising methods to detect linear features from radar data acquired over icy regions. Further, [7] used a level set approach to evaluate airborne radar imagery whereas [8] used anisotropic diffusion followed by a contour detection model to identify ice and bedrock layers. These methods, although

giving accurate results, resulted in a binary ice-layer detection, i.e. they detected the presence or non-presence of ice for each pixel. Moreover, these methods focused on detecting only the surface and bottom layers of the radargrams.

Tracking the internal ice sheet layers is much more difficult since the layers are compact and too close to each other. Although there were several works in this field such as [4], [18]–[20] which used automated techniques to detect internal layers, none of these methods used deep learning; and hence were not scalable for larger datasets. Several recent efforts [17], [21]–[24] applied multi-scale deep learning techniques to track and identify internal ice layers. Although these are very efficient methods, they perform binary detection of layers, i.e. they detect the presence or absence of ice at a given pixel. As snow gets accumulated over the years, forming a separate ice layer for each year, there lies a potential to detect *which* layer an ice-pixel belongs to. We aim to solve this problem of tracking the compact, closely spaced, internal ice layers and identifying each layer uniquely. We use deep learning for its recent successes and scalability to large datasets. Since we wanted a pixel-wise distribution of each layer, we used FCNs for semantic segmentation which is described in the following subsection.

B. Semantic Segmentation

FCNs have been used extensively for semantic segmentation of images. The immense applicability of these networks and semantic segmentation in particular has resulted in it becoming a fundamental topic in Computer Vision [25].

The concept of semantic segmentation was introduced in [15], where the the terminating fully connected layers from popular networks like VGG and AlexNet were replaced with fully convolutional layers to bring pixel-wise classification. Since then, these networks have further been modified by various strategies such as global or average pooling [26], batch normalization, different activation [27] and loss functions [28], multi-scale architectures [16], [29], [30]etc. A fusion of various training strategies have led to their success. In this section, we briefly describe some of the very successful semantic segmentation networks in recent times, explaining their utility. In the following section, we give the details of these network architectures.

a) *UNet*: This network [31] contains a contracting path and an expansive path, which are almost symmetric to each other, forming a U-shaped architecture. High resolution features from the contracting path are concatenated with the upsampled outputs in the expansive path in order to obtain high localization. Such a network strategy, combined with data augmentation, turned out to be useful in biomedical image segmentation, especially for detecting tissue deformations. The success of UNet in biomedical images [31] led to its application and improvement for other domains such as remote sensing and autonomous driving.

b) *PSPNet*: In [30], the authors observe that in a traditional FCN, most of the errors were due to a lack of global contextual relationship between different receptive fields. Hence,

they introduced a pyramid pooling module which empirically turned out to be an effective global contextual prior. They start with global average pooling as a baseline for global contextual prior, and then fuse it with different sub-region context hierarchically so as to contain information from different scales. They refer to this hierarchical structure as a *pyramid pooling module*. Such a network strategy turns out to be very useful for snow radar dataset, as not only do we need to extract gradient changes in the noise locally, we also need to uniquely identify each layer from a global perspective. More details of the architecture are described in Section IV.

c) *DeepLabv3+*: This architecture [25] uses the concept of spatial pyramid pooling and applies several parallel atrous convolutions at different dilation rates to build an encoder architecture. The output of this is further upsampled (decoded) to extract features at the image scale. Such a combination of pyramid pooling and encoder-decoder architecture helps in learning multi-scale contextual information while also detecting sharp object boundaries through spatial information.

We learnt that spatial pyramid pooling, such as that used in PSPNet, helps in learning a global contextual prior, but it also misses out on detailed object information due to the pooling operation. Atrous convolutions, help in this regard by dilating the receptive field in a controlled manner before it is pooled. The subsampled pooled output can then be decoded to obtain sharper spatial information. This network strategy can help us get highly detailed information about each layer change in the Snow Radar data. This architecture is further explained in Section IV.

III. DATASET

A. Characteristics

We use the Snow Radar data from 2012 year, provided by the Center for Remote Sensing of Ice Sheets (CReSIS) [32] and having a resolution of 4 cm per pixel in the vertical direction. This is publicly available and consists of 2361 training images and 260 test images. We have used the output of a semi-supervised technique [4] as the ground truth. In our ground-truth data, each unique ice-layer is marked as a separate class.

B. Challenges

The snow radar images are quite noisy, and it is very hard to distinguish where each layer begins. Moreover, there are hardly any contrasting features which can help us distinguish between layers. There are also certain anomalies, creating vertical perturbations in the horizontal ice layers. All these issues in the radar images can be seen in Figure 2a.

These issues in the radar images propagate to the their labels, Figure 2b, which is the output of [4] and which we use as ground truth. Most, if not all, the labels for the deeper layers are missing here. Moreover, the labels which are available, are incomplete as they do not cover the corresponding ice layer completely. These significant anomalies and issues in the original radar images, as well as the training labels, make it challenging for them to be directly trained with any FCN.

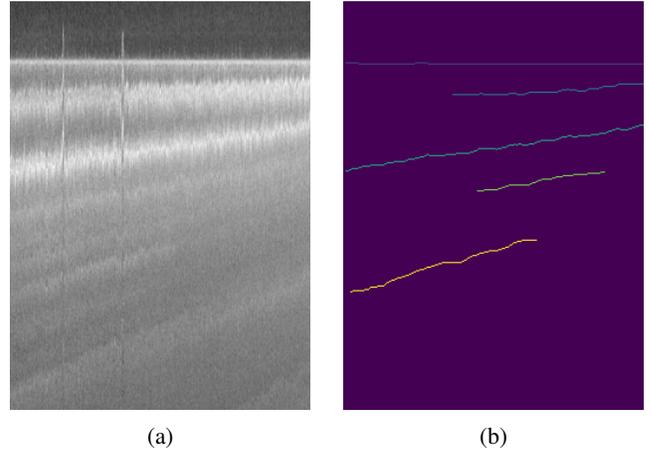


Fig. 2: Noisy radar image (a), having multiple indistinct layers, and its corresponding training labels (b). The lower layers of the radar image are not so easy to be distinguished by human eyes, whereas the available labels do not span across the corresponding layers completely.

As FCNs or CNNs are highly data dependent, any anomaly or issue in the original data or labels will directly propagate to the network output, leaving it to be of no practical use.

Thus, in order to get around these significant issues in the data and labels, we introduce some steps to process the training labels. These are described in Section IV-A and help us in extracting only the complete labels for layers, by discarding the incomplete labels. We follow this procedure to crop out consecutive sets of complete training labels, and also crop out the corresponding regions from the original radar image. From the original CReSIS data of 2361 training images, our cropping procedure leaves us with 1157 images. 20% of these training images, i.e. 232, are explicitly used for neural network validation purposes. We use the entire 260 images for testing purposes, the annotations of which we were able to manually complete using the Darwin V7 platform [33].

IV. METHODOLOGY

In this section, we discuss how we process the incomplete or missing training labels, the network architectures of the three FCNs, and the post-processing we carry out to obtain layered outputs. In the next section, we talk about the hyperparameter setup for the FCNs, and the evaluation metrics we use to assess their performance.

A. Processing the Training Labels

As some of the training labels available to us were incomplete (such as the second, fourth and fifth layer in Figure 2b), we removed these completely, that is turn them into background pixels. Then, starting from the topmost layer, we searched for a consecutive set of at least two layers. For every consecutive set found, we calculate the row index of the peak of the top layer, and the row index of the valley of the bottom layer. We then added a margin of five to both these indices setting these as the y - coordinate values of the bounding box

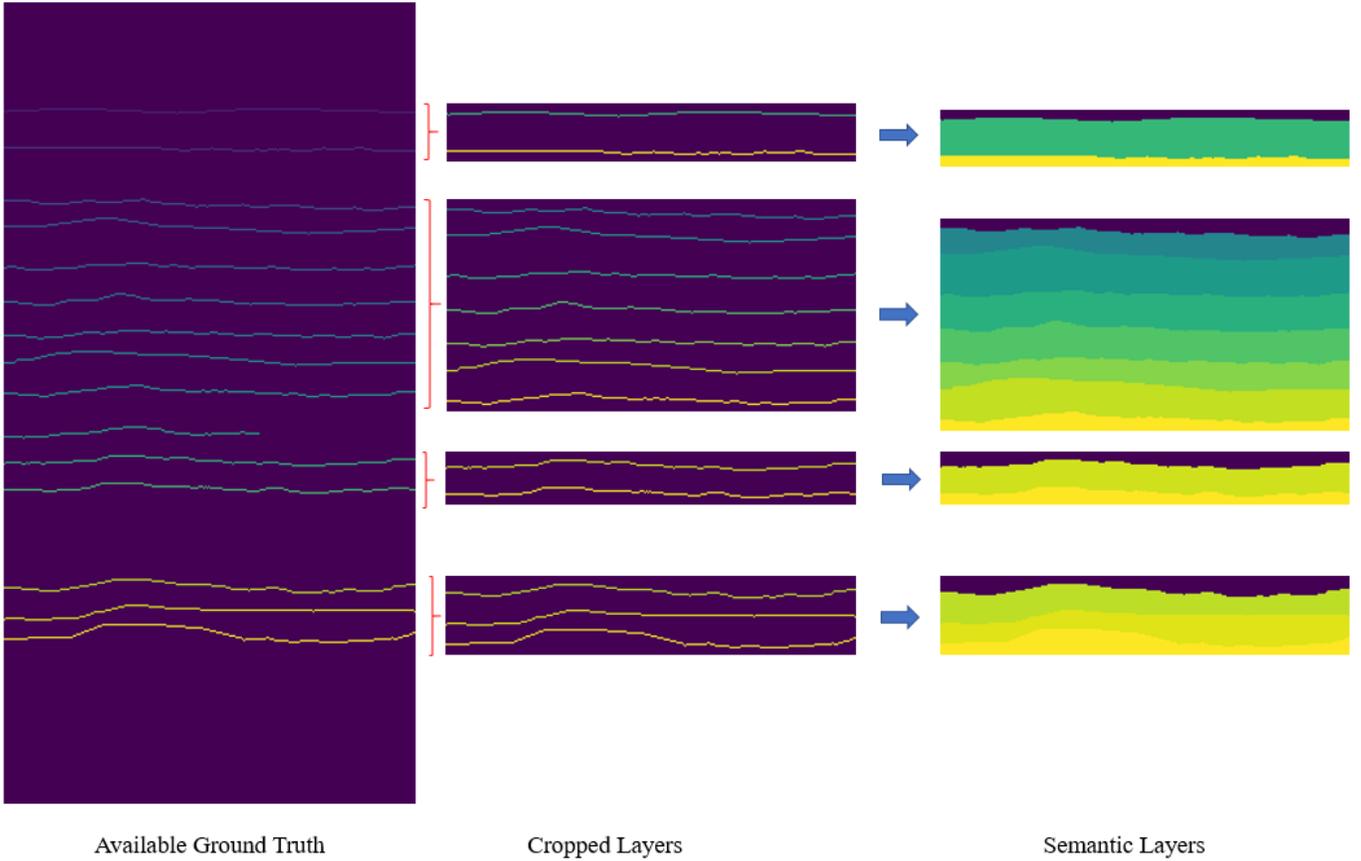


Fig. 3: Processing the Training Labels: First, consecutive sets of completely labeled layers are cropped out. In the second step, pixels in between two layers are filled up with the label of the upper layer. This pixel-wise annotation across the thickness of the layers will help FCN models to learn features of every layer uniquely. Each label color represents a unique layer. Note that the colors are generated in a spectrum, and are not necessarily consistent across the ground truth and the semantic output. The red curly braces represent the row indices for cropping.

for cropping this consecutive set of layers. The bounding box spans across all the columns of the training labels, i.e. its x -coordinate values are the first and the last column index of the ground truth image respectively. These bounding box coordinates are then used to crop out the same region from the corresponding, original radar image.

Consider Figure 3 as an example. The available ground truth here has the following layer-labels available: 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 18, 19, 20. There are no layers labelled as 1, 4, 15, 16, and 17. Also, layer labelled 12 is incomplete, and doesn't span across the width of the image. To process this ground truth, we first completely remove layer labelled 12, i.e. we turn it to background. We do this by converting all pixels having value 12, to have value 0 (the background class). So now we are left with layers labelled as: 2, 3, 5, 6, 7, 8, 9, 10, 11, 13, 14, 18, 19, 20. Out of these, the consecutive sets available are: $\{2,3\}$, $\{5, 6, 7, 8, 9, 10, 11\}$, $\{13, 14\}$, and $\{18, 19, 20\}$. The first layer in each set is its top layer, and the last layer in each set is its bottom layer. By calculating the row indices of the peaks of the top layers of each set, i.e. layers 2, 5, 13, and 18; and row indices of the valleys of the

bottom layers of each set, i.e. layers 3, 11, 14, and 20; we are able to crop these sets out into separate images, after adding a margin of five pixels to the aforementioned row indices.

The bounding box coordinates are explained in Equation 1. For a peak row index p and a valley row index v , of an image with width w , the bounding box used for cropping has coordinate values (the top-left coordinate, and the bottom-right coordinate) as computed by Equation 1. These same coordinates are used for cropping the corresponding radar image, as shown in Figure 4.

$$(x_1, y_1), (x_2, y_2) = (0, p - 5), (w, v + 5) \quad (1)$$

Further, in order to feed FCNs for semantic segmentation, we need training labels which are annotated for each pixel of the image. To accomplish this in the cropped subsets, we fill all the intermediate background pixels between two layers with the label of the *upper* layer. This leaves us with labelled pixels across most of the image, except for the background pixels above the top-most layer of the image. This layer-filling

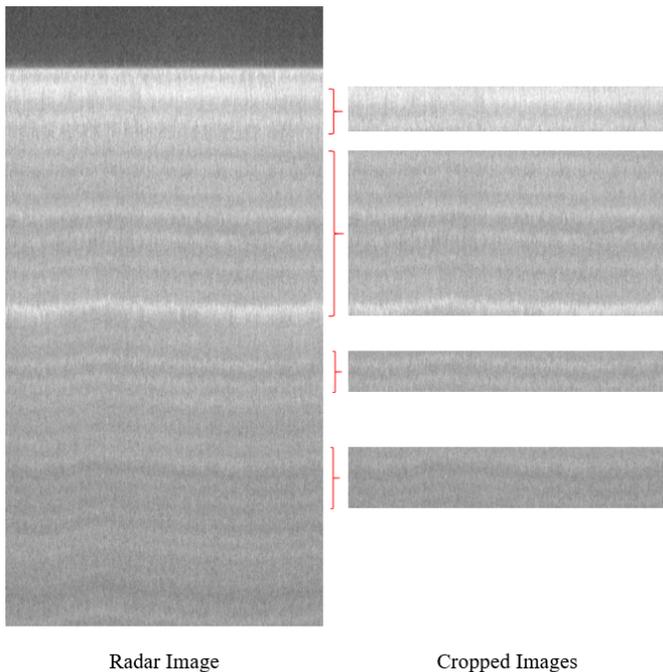


Fig. 4: This is the radar image for the training labels of Figure 3. This image is cropped at the same regions as its corresponding label-image in Figure 3. The red curly braces represent the row indices where it was cropped to generate ‘Cropped Images’.

process is also shown in Figure 3, where we generate the ‘Semantic Layers’ from the ‘Cropped Layers’.

B. Network Architectures

We carried out semantic segmentation of Snow Radar images using three state-of-the-art FCNs: UNet [31], PSPNet [30] and DeepLabv3+ [25]. In this Section, we give some details about their architectures.

1) *UNet*: The architecture of UNet [31] consists of a contracting path and an expansive path. The contracting path (left side) has repeated applications of two 3×3 unpadding convolutions and 2×2 max pooling (of stride 2) operation for downsampling. Each convolution is followed by a rectified linear unit (ReLU) activation and at each downsampling step, the number of feature channels are doubled. The expansive path (right side) then focuses on upsampling the feature maps followed by 2×2 convolutions which half the number of channels, which are then concatenated with the corresponding cropped features maps from the contracting paths. These are then convolved by two 3×3 filters each having a ReLU activation function. Finally, a 1×1 convolution is used to reduce the feature vector to the desired number of classes.

2) *PSPNet*: The pyramid pooling module fuses features from four different pyramid scales. The coarsest level generates a single bin output through a global pooling scheme, whereas other levels generate pooled representations for different sub-regions. These low dimension pooled outputs from

different levels are then upsampled to get feature maps of the same size as the original feature map via bilinear interpolation. These different features are then concatenated to give the final prediction. The pyramid pooling module that we adopt has four bin size of 1×1 , 2×2 , 3×3 and 6×6 respectively.

The baseline CNN that we use in the PSPNet architecture is ResNet-50 [34]. However, contrary to the ResNet architecture, PSPNet incorporates an additional, auxiliary loss after the fourth stage (residual block) of ResNet to deeply supervise [35] the network architecture. The entire network is trained by a weighted loss that balances between this auxiliary loss and the main, terminal loss.

3) *DeepLabv3+*: We build DeepLabv3+ [25] using a ResNet-50 [34] as the baseline. We then apply multiple atrous convolutions with different dilation rates (6, 12 and 18) to extract the spatial information. This is then fused with the pooled output of the feature maps and later convolved with 1×1 filters. These encoder features are then bilinearly upsampled by a factor of 4, to be later concatenated with the low-level features from the network backbone which have the same spatial resolution. Further, in the decoder path, 1×1 convolutions are applied on these features to reduce the number of channels and make the training easier. Feature maps are then concatenated and convolved with a couple of 3×3 filters to refine the features, which are then upsampled by a factor of 4 by using bilinear interpolation.

C. Processing the Network Outputs

The fully convolutional networks described above are expected to give us pixel-wise outputs, like those shown in ‘Semantic Layers’ column of Figure 3. In order to convert them back to individual layers (such as ‘Cropped Layers’ of Figure 3), we iterate over each row of every column of the output to convert all *duplicate* labels to the background pixel. Thus, each column of the output will have only one unique pixel for every label. We do this for all the columns, thus reconstructing layered output similar to the ‘Cropped Layers’ column of Figure 3.

V. EXPERIMENTAL SETUP

This section explains the setup of our experiments and describes the metrics we used to assess the FCNs’ performance to segment the images and calculate the ice-layer thickness. The maximum number of unique layers we had available were 27, and we trained all networks on 28 classes to predict layer pixels as well as background pixels.

In order to understand the usability of our networks especially for internal ice layer tracking, we calculate the metrics on the test images having more than 1, and more than 3 layers. As there were a lot of images with less than 3 layers, this test will give us an idea of how the networks perform for *deeper* layers. Furthermore, as each layer corresponds to the snow accumulated in a particular year, we calculate the performance on the top 10 layers of the test set. This is to study the changes in the past decade which can help us predict any changes in the near future.

A. Hyperparameters

All the networks were trained with ResNet-50 [34] as the baseline network, and an initial learning rate of 0.01. We used a weight decay of 10^{-4} and a momentum of 0.9. We performed two types of experiments for each network, one where the learning rate scheduler was 'Poly' and the other where it was 'OneCycle'. In the Poly learning rate scheduler, the learning rate is linearly reduced from the initial value (0.01) to zero as the training progresses (Figure 5a), while the momentum remains constant at 0.9. In the OnceCycle learning rate scheduler, the learning rate is annealed according to the one-cycle learning rate policy [36] (Figure 5b). This means that, during the initial 30% of the training, the learning rate increases from a tenth of its (0.001) value to its full value (0.01). For the remainder of the training, the learning rate decreases from its full value to a quarter of its value. The momentum is also changed in a similar but opposite way. It decreases from 0.9 to 0.8, and then later increases to 0.9 again during second half (remaining 70%) of the training. All networks were trained with a cross entropy loss having a mini-batch size of 8 images for 200 epochs.

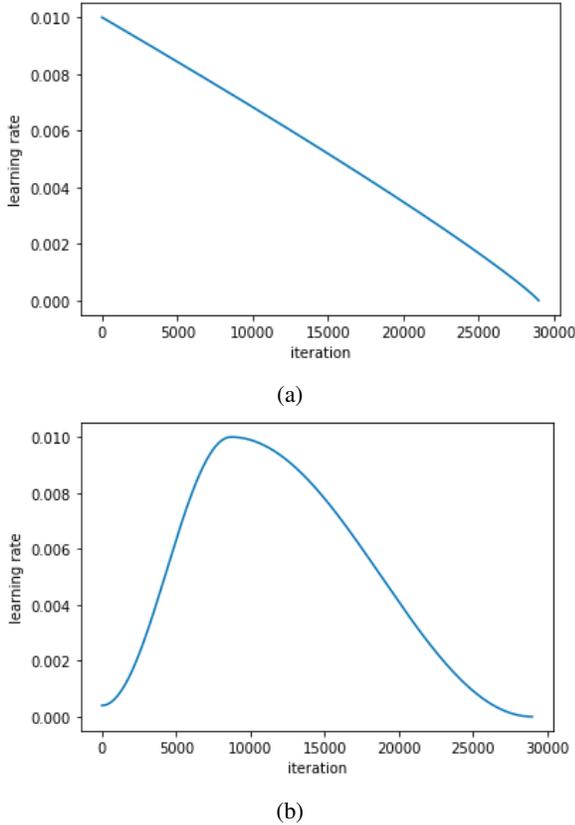


Fig. 5: (a) Poly Learning Rate Scheduler and (b) One Cycle Learning Rate Scheduler.

B. Evaluation Metrics

We assess the performance of these networks with overall accuracy and mean IoU (intersection over union) per label.

For k labels, these metrics are given as

$$Accuracy = \frac{\sum_{i=1}^k \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{k} \quad (2)$$

$$mean IoU = \frac{\sum_{i=1}^k \frac{Predicted Output_i \cap Ground Truth_i}{Predicted Output_i \cup Ground Truth_i}}{k} \quad (3)$$

where TP, TN, FP, FN are True Positives, True Negatives, False Positives, and False Negatives, respectively. This is done on pixel-wise (semantic) outputs and ground truth in the format of 'Semantic Layers' of Figure 3.

We also calculate the *mean* thickness of each layer in every predicted image, and compare it with the corresponding ground truth semantic layers. For calculating this mean thickness, we first count the total number of pixels for each unique class, and divide it by the number of columns (width) of the image. We then calculate the Mean Absolute Error (MAE) between the predicted output and the ground truth across all the layers of a given image. This is given by

$$MAE = \frac{\sum_{i=1}^k |p_i - t_i|}{k} \quad (4)$$

where p_i is the predicted mean thickness and t_i is the true mean thickness of the i^{th} layer.

VI. RESULTS AND DISCUSSION

Network-LRS	Train	Val	Test
UNet-Poly	0.755	0.681	0.714
UNet-OneCycle	0.856	0.754	0.792
PSPNet-Poly	0.948	0.875	0.899
PSPNet-OneCycle	0.938	0.844	0.867
DeepLabv3+-Poly	0.957	0.907	0.887
DeepLabv3+-OneCycle	0.935	0.876	0.886

TABLE I: Accuracy of various networks on the Training, Validation and Test set computed over all the 27 layers. LRS denotes the Learning Rate Strategy - i.e. Poly or OneCycle. The highest values obtained over the Validation and Test sets are highlighted in bold.

Network-LRS	Train	Val	Test
UNet-Poly	0.387	0.288	0.343
UNet-OneCycle	0.549	0.378	0.438
PSPNet-Poly	0.737	0.576	0.65
PSPNet-OneCycle	0.728	0.538	0.589
DeepLabv3+-Poly	0.734	0.609	0.59
DeepLabv3+-OneCycle	0.676	0.552	0.595

TABLE II: Mean Intersection over Union (IoU) of various networks on the Training, Validation and Test sets computed over all the 27 layers. LRS denotes the Learning Rate Strategy - i.e. Poly or OneCycle. The highest values obtained over the Validation and Test sets are highlighted in bold.

We calculated the accuracy and mean IoU per (layer) class based on Equations 2 and 3 respectively. We tabulate these

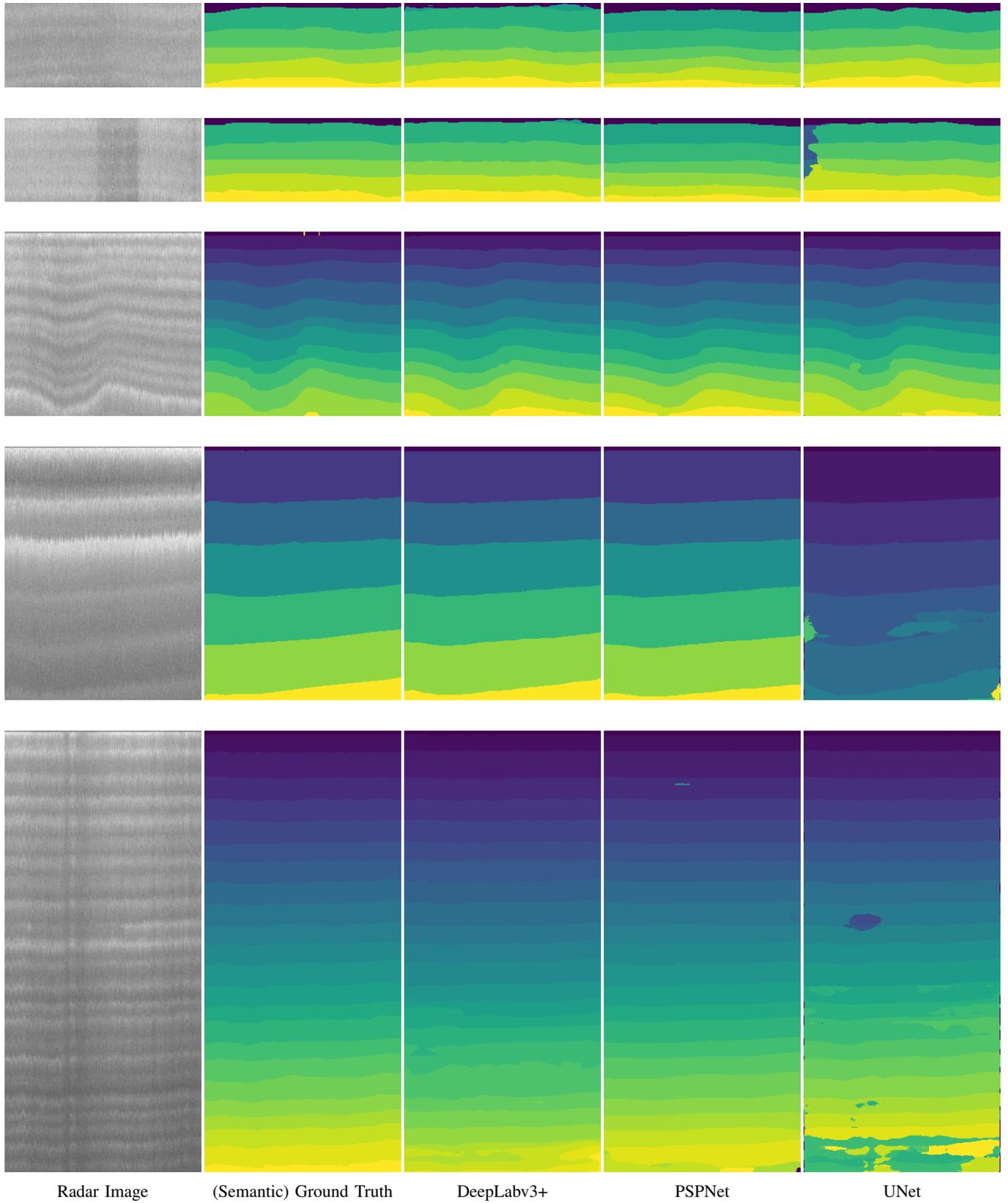


Fig. 6: Comparing the outputs of DeepLabv3+, PSPNet and UNet with respect to the test images and available ground truth.

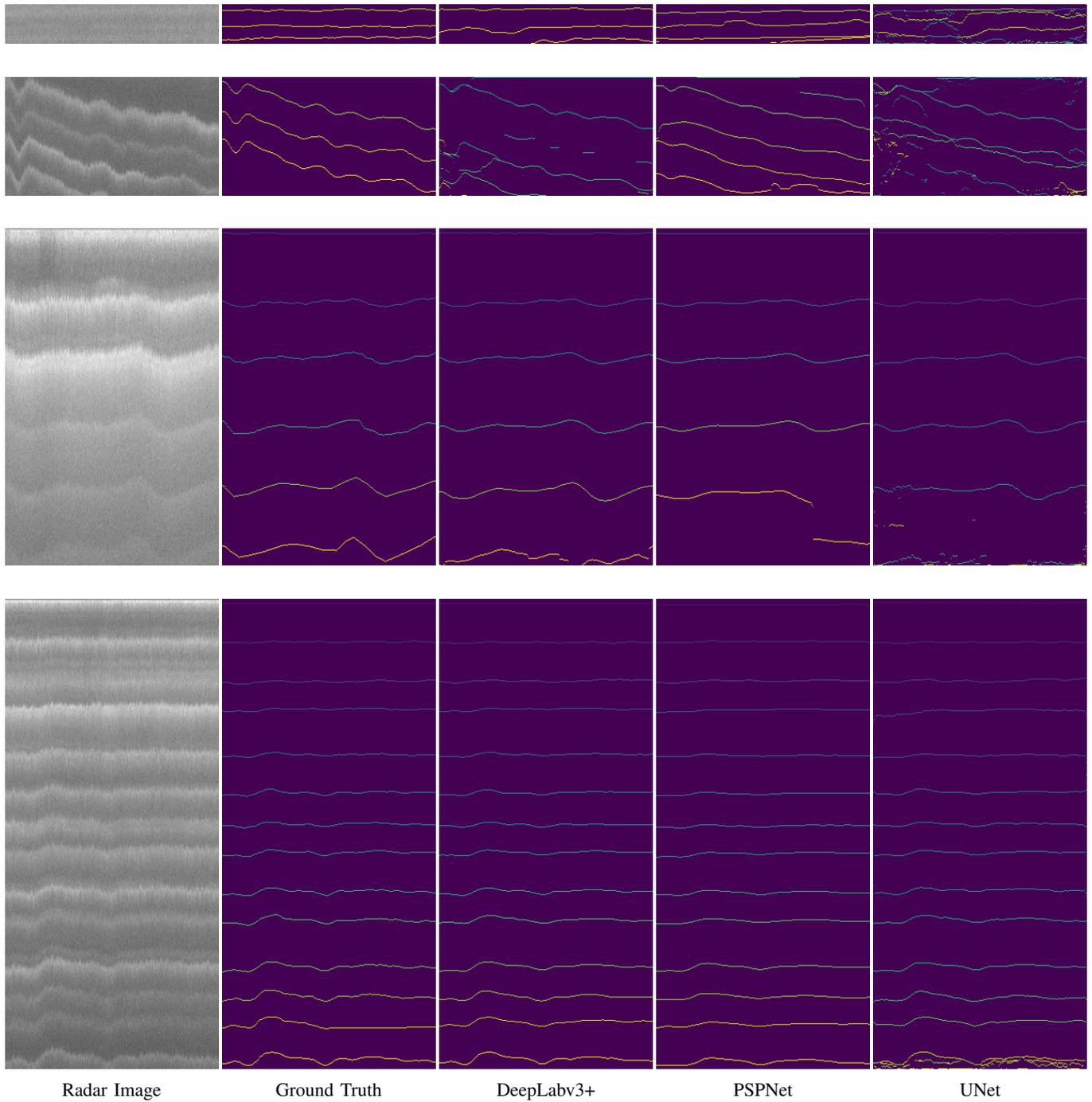


Fig. 7: Comparing the outputs of DeepLabv3+, PSPNet and UNet with respect to the test images and available ground truth.

for all six experiments (three networks, and two learning rate schedulers per network) in Tables I and II. In these tables we further highlight the highest accuracy and mean IoU obtained over Validation and Test sets obtained over all the 27 layers. We also calculated the performance metrics on the test images having more than 1 and more than 3 layers, with the metrics calculated over the top 10 layers; which is more relevant for climate studies. These are shown in Tables III and IV.

Network-LRS	> 1 layers	> 3 layers
UNet-Poly	0.633	0.778
UNet-OneCycle	0.626	0.82
PSPNet-Poly	0.797	0.947
PSPNet-OneCycle	0.681	0.888
DeepLabv3+-Poly	0.733	0.915
DeepLabv3+-OneCycle	0.831	0.943

TABLE III: The accuracy calculated over the top 10 layers of the test set. The results here are for images having more than 1, and more than 3 internal layers, which is more useful for our analysis.

Network-LRS	> 1 layers	> 3 layers
UNet-Poly	0.266	0.302
UNet-OneCycle	0.292	0.351
PSPNet-Poly	0.387	0.419
PSPNet-OneCycle	0.366	0.413
DeepLabv3+-Poly	0.392	0.424
DeepLabv3+-OneCycle	0.425	0.435

TABLE IV: The mean IoU calculated over the top 10 layers of the test set. The results here are for images having more than 1, and more than 3 internal layers, which is more useful for our analysis.

From tables I and II we see that, the Poly learning rate gives higher performance with the multi-scale networks of PSPNet and DeepLabv3+. For UNet, the OneCycle learning rate works better. The gradual increase in the learning rate does not help with the multiple pooling strategies that PSPNet and DeepLabv3+ incorporate. For DeepLabv3+, both the learning rate schedulers give similar accuracy and mean IoU over the test set. Overall, DeepLabv3+ gave a higher mean IoU over the 27-layered Validation and Test sets, while also giving the highest accuracy over the Validation set. UNet performed the worst, both qualitatively (Figures 6 and 7) and quantitatively (Tables I and II). For most of the images in Figure 6, UNet creates botchy patches, not being able to predict a layer completely across its width. This further leads to a lot of broken lines when we convert the semantic outputs to layered outputs (Figure 7).

We believe that the poor performance of UNet is due to its primitive architecture, as compared to PSPNet and DeepLabv3+. DeepLabv3+ captures not only a global contextual prior, but it is also able to retain intricate spatial information. Due to these reasons, it is able to decipher the highly ambiguous ice-layers while detecting them from a broader perspective.

We also calculated the thickness of each layer in the networks' predicted output and compared it with the semantic

Network-LRS	Train	Val	Test
UNet-Poly	7.95	10.14	8.75
UNet-OneCycle	5.22	7.66	6.17
PSPNet-Poly	2.80	4.79	3.63
PSPNet-OneCycle	4.03	7.34	5.62
DeepLabv3+-Poly	2.36	3.66	3.75
DeepLabv3+-OneCycle	3.08	4.53	3.59

TABLE V: The Mean Absolute Error (MAE) in pixels of all the layers calculated over the Training, Validation and Test sets. Values highlighted in bold are the least thickness values obtained over the Validation and Test sets.

ground truth layers (such as those present in Figure 3). We report the Mean Absolute Error (MAE, Equation 4) of all the layers across all the images of each dataset (i.e. Training, Validation and Test sets) in Table V. From this table, we see that the semantic segmentation networks have predicted well, resulting in an average MAE across all datasets and all networks to be 5.28 pixels. Further, DeepLabv3+ gave the best outputs with its MAE falling between 3 to 4 pixels for the Validation and Test sets; closely followed by PSPNet which gave an MAE between 3.5 and 7 pixels. UNet's outputs gave the worst thickness estimates, with an MAE of upto 10 pixels. This is majorly due to the botchy semantic output that it generates over the deeper layers.

Although training till 200 epochs could not improve UNet's output, tuning other hyperparameters apart from learning rate should definitely improve its results. Further, as UNet is a 'lighter' model in terms of number of weights, complexity, and mathematical operations involved, experimenting with it would be useful.

VII. CONCLUSION

Global warming is rapidly melting glaciers and ice sheets across the world. This calls for automated accurate methods which can process the large amount of data that is available from Earth observation. In this paper, we use Snow Radar data to track internal ice sheets and estimate the thickness of each layer. More specifically, we mitigate some of the challenges of the data set and its ground truth by a set of pre-processing techniques. We also use state-of-the-art fully convolutional networks to understand the pixel-wise distribution and extent of each ice-layer. By using this methodology, we are able to estimate the thickness of these layers within a Mean Absolute Error of 3 to 4 pixels.

Even a slight change in the polar ice-sheets can be devastating for the world. Our work can thus be expanded by incorporating datasets from multiple years, and creating data-driven, real-time monitoring solutions which can go beyond label ambiguities.

ACKNOWLEDGMENT

This work is supported by NSF BIGDATA awards (IIS-1838230, IIS-1838024), IBM, and Amazon.

REFERENCES

- [1] IPCC, "Ipcc ar5 wg1 summary for policymakers," 2014.
- [2] S. Gogineni, J. B. Yan, D. Gomez, F. Rodriguez-Morales, J. Paden, and C. Leuschen, "Ultra-wideband radars for remote sensing of snow and ice," in *IEEE MTT-S International Microwave and RF Conference*, 2013, pp. 1–4.
- [3] E. Arnold, C. Leuschen, F. Rodriguez-Morales, J. Li, J. Paden, R. Hale, and S. Keshmiri, "Cresis airborne radars and platforms for ice and snow sounding," *Annals of Glaciology*, p. 1–10, 2019.
- [4] L. S. Koenig, A. Ivanoff, P. M. Alexander, J. A. MacGregor, X. Fettweis, B. Panzer, J. D. Paden, R. R. Forster, I. Das, J. R. McConnell, M. Tedesco, C. Leuschen, and P. Gogineni, "Annual greenland accumulation rates (2009–2012) from airborne snow radar," *The Cryosphere*, vol. 10, no. 4, pp. 1739–1752, 2016. [Online]. Available: <https://www.the-cryosphere.net/10/1739/2016/>
- [5] A. Ferro and L. Bruzzone, "Automatic extraction and analysis of ice layering in radar sounder data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 3, pp. 1622–1634, 2013.
- [6] A. Ilisei and L. Bruzzone, "A system for the automatic classification of ice sheet subsurface targets in radar sounder data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 6, pp. 3260–3277, 2015.
- [7] M. Rahnemoonfar, G. C. Fox, M. Yari, and J. Paden, "Automatic ice surface and bottom boundaries estimation in radar imagery based on level-set approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5115–5122, 2017.
- [8] M. Rahnemoonfar, A. A. Habashi, J. Paden, and G. C. Fox, "Automatic ice thickness estimation in radar imagery based on charged particles concept," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017, pp. 3743–3746.
- [9] M. Rahnemoonfar, R. Murphy, M. V. Miquel, D. Dobbs, and A. Adams, "Flooded area detection from uav images based on densely connected recurrent neural networks," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 1788–1791.
- [10] C. Sheppard and M. Rahnemoonfar, "Real-time scene understanding for uav imagery based on deep convolutional neural networks," in *Geoscience and Remote Sensing Symposium (IGARSS), 2017 IEEE International*. IEEE, 2017, pp. 2243–2246.
- [11] M. Rahnemoonfar and C. Sheppard, "Deep count: fruit counting based on deep simulated learning," *Sensors*, vol. 17, no. 4, p. 905, 2017.
- [12] M. Rahnemoonfar, D. Dobbs, M. Yari, and M. J. Starek, "Discountnet: Discriminating and counting network for real-time counting and localization of sparse objects in high-resolution uav imagery," *Remote Sensing*, vol. 11, no. 9, p. 1128, 2019.
- [13] M. Rahnemoonfar and D. Dobbs, "Semantic segmentation of underwater sonar imagery with deep learning," in *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2019, pp. 9455–9458.
- [14] M. Rahnemoonfar, J. Johnson, and J. Paden, "Ai radar sensor: Creating radar depth sounder images based on generative adversarial network," *Sensors*, vol. 19, no. 24, p. 5479, Dec 2019. [Online]. Available: <http://dx.doi.org/10.3390/s19245479>
- [15] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 04, pp. 640–651, apr 2017.
- [16] S. Xie and Z. Tu, "Holistically-nested edge detection," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1395–1403.
- [17] M. Yari, M. Rahnemoonfar, J. Paden, I. Oluwanisola, L. Koenig, and L. Montgomery, "Smart tracking of internal layers of ice in radar data via multi-scale learning," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 5462–5468.
- [18] L. Carrer and L. Bruzzone, "Automatic enhancement and detection of layering in radar sounder data based on a local scale hidden markov model and the viterbi algorithm," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 962–977, 2017.
- [19] J. A. MacGregor, M. A. Fahnestock, G. A. Catania, J. D. Paden, S. Prasad Gogineni, S. K. Young, S. C. Rybarski, A. N. Mabrey, B. M. Wagman, and M. Morlighem, "Radiostratigraphy and age structure of the greenland ice sheet," *Journal of Geophysical Research: Earth Surface*, vol. 120, no. 2, pp. 212–241, 2015. [Online]. Available: <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2014JF003215>
- [20] V. d. P. Onana, L. S. Koenig, J. Ruth, M. Studinger, and J. P. Harbeck, "A semiautomated multilayer picking algorithm for ice-sheet radar echograms applied to ground-based near-surface data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 1, pp. 51–69, 2015.
- [21] M. Rahnemoonfar, M. Yari, J. Paden, L. Koenig, and I. Oluwanisola, "Deep multi-scale learning for automatic tracking of internal layers of ice in radar data," *Journal of Glaciology*, pp. 1–10, 2020.
- [22] M. Yari, M. Rahnemoonfar, J. Paden, L. Koenig, and I. Oluwanisola, "Multi-scale and temporal transfer learning for automatic tracking of internal ice layers," in *IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, p. in press.
- [23] M. Rahnemoonfar, M. Yari, and J. Paden, "Radar sensor simulation with generative adversarial network," in *IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, p. in press.
- [24] I. Oluwanisola, J. Paden, M. Rahnemoonfar, D. Crandall, and M. Yari, "Snow radar layer tracking using iterative neural network approach," in *IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, p. in press.
- [25] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 833–851.
- [26] M. Lin, Q. Chen, and S. Yan, "Network in network," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- [27] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with relu activation function and linear spline-type methods," *Neural Networks*, vol. 110, pp. 232 – 242, 2019.
- [28] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 2018, pp. 8778–8788.
- [29] Y. Liu, M. Cheng, X. Hu, J. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1939–1946, 2019.
- [30] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [31] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.
- [32] CREsis, "Snow radar data," Lawrence, Kansas, USA, 2020. [Online]. Available: <https://data.cresis.ku.edu/data/snow/>
- [33] "V7 darwin," <https://www.v7labs.com/darwin>, accessed: 2020-11-11.
- [34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [35] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-Supervised Nets," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Lebanon and S. V. N. Vishwanathan, Eds., vol. 38. San Diego, California, USA: PMLR, 09–12 May 2015, pp. 562–570. [Online]. Available: <http://proceedings.mlr.press/v38/lee15a.html>
- [36] L. N. Smith and N. Topin, "Super-convergence: very fast training of neural networks using large learning rates," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, T. Pham, Ed., vol. 11006, International Society for Optics and Photonics. SPIE, 2019, pp. 369 – 386. [Online]. Available: <https://doi.org/10.1117/12.2520589>