*Journal of Glaciology*

**Article**

# Deep multi-scale learning for automatic tracking of internal layers of ice in radar data

Maryam Rahnemoonfar[1], Masoud Yari[1,*], John Paden[2], Lora Koenig[3] and Oluwanisola Ibikunle[2]

[1]Computer Vision and Remote Sensing Laboratory, University of Maryland, Baltimore County, Baltimore, MD 21250; [2]Center for Remote Sensing of Ice Sheets (CReSIS), University of Kansas Lawrence, KS and [3]National Snow and Ice Data Center, Cooperative Institute for Research in Environmental Sciences, University of Colorado, Boulder, CO

*Present address: Computer Vision and Remote Sensing Laboratory (Bina Lab), University of Maryland, Baltimore County, Baltimore, MD 21250.

**Cite this article:** Rahnemoonfar M, Yari M, Paden J, Koenig L, Ibikunle O (2020). Deep multi-scale learning for automatic tracking of internal layers of ice in radar data. *Journal of Glaciology* 1–10. https://doi.org/10.1017/jog.2020.80

Received: 25 August 2020
Revised: 31 August 2020
Accepted: 1 September 2020

**Key words:**
Ice and climate; radio-echo sounding; remote sensing

**Author for correspondence:**
Masoud Yari, E-mail: yari@umbc.edu

## Abstract

In this study, our goal is to track internal ice layers on the Snow Radar data collected by NASA Operation IceBridge. We examine the application of deep learning methods on radar data gathered from polar regions. Artificial intelligence techniques have displayed impressive success in many practical fields. Deep neural networks owe their success to the availability of massive labeled data. However, in many real-world problems, even when a large dataset is available, deep learning methods have shown less success, due to causes such as lack of a large labeled dataset, presence of noise in the data or missing data. In our radar data, the presence of noise is one of the main obstacles in utilizing popular deep learning methods such as transfer learning. Our experiments show that if the neural network is trained to detect contours of objects in electro-optical imagery, it can only track a low percentage of contours in radar data. Fine-tuning and further training do not provide any better results. However, we show that selecting the right model and training it on the radar imagery from the start yields far better results.

## 1. Introduction

Ice loss in Greenland and Antarctica has accelerated in recent decades (IMBIE-Team and others, 2019; Shepherd and others, 2018; Rignot and others, 2019). Melting polar ice sheets and mountain glaciers have considerable influence on sea level rise and ocean currents; potential floods in coastal regions could put millions of people around the world at risk (IPCC, 2014*b*). The Intergovernmental Panel on Climate Change (IPCC) estimates that sea level could increase by 26–98 cm by the end of this century (IPCC, 2014*a*). This large range in predicted sea level rise can be partially attributed to an incomplete understanding of the surface mass balance and ice dynamics in Greenland and Antarctica. Recent large-scale radar surveys of Greenland and Antarctica, for example as a part of NASA Operation IceBridge (Rodriguez-Morales and others, 2018), reveal internal ice layers on a continental scale. These internal ice layers illuminate many aspects of ice sheets dynamics, including their history and their response to climate and subglacial forcing (MacGregor and others, 2016; Cavitte and others, 2016; Koenig and others, 2016; Medley and others, 2014). The Scientific Committee on Antarctic Research has formed an action group, AntArchitecture, to catalog internal layers over the whole of the Antarctica ice sheet (Bingham and others, 2019) and several other efforts have completed internal layer tracking of smaller regions of Antarctica and across much of Greenland (Medley and others, 2014; MacGregor and others, 2015*a*; Koenig and others, 2016). Due to the large size of the radar datasets, it is important to develop fully automatic artificial intelligence (AI) techniques to detect internal layers hidden within the ice sheets.

In this study we focus on the Snow Radar (Rodriguez-Morales and others, 2018) data collected by NASA Operation IceBridge. The Snow Radar operates from 2 to 18 GHz and is able to resolve shallow layers with fine resolution over wide areas of the ice sheet. In many areas, these shallow layers represent annual isochrones allowing the annual snow accumulation to be estimated to better understand the surface mass balance in these regions (Medley and others, 2014; Koenig and others, 2016). In the past two decades a number of layer tracing algorithms have been developed for tracking internal layers. Carrer and Bruzzone (2016) provide a comprehensive review of these internal layer tracing algorithms. There are a few methods presented which are fully automatic and only require parameter tuning with a training set. The local-Viterbi-based algorithm presented by Carrer and Bruzzone (2016) is an example. Two tracking algorithms (de Paul Onana and others, 2014; Mitchell and others, 2013*a*) have been tested on the Snow Radar dataset. de Paul Onana and others (2014)'s algorithm facilitated the tracking of a large amount of data for Koenig and others (2016), but an improved algorithm with more efficient and comprehensive tracking is desired since a significant amount of time is still required for indexing layers and quality control, and many layers that are traceable by a trained operator are not detected by the algorithm. Mitchell and others (2013*a*)'s algorithm has only been used to track a small set of images. None of the algorithms reviewed by Carrer and Bruzzone (2016) make use of deep learning. A few algorithms have been published since Carrer and Bruzzone (2016) published their review including

Rahnemoonfar and others (2016, 2017*b*); Xu and others (2018); Rahnemoonfar and others (2019*b*); and Berger and others (2019). Some of these more recently published algorithms use deep learning (Xu and others, 2018; Kamangir and others, 2018; Rahnemoonfar and others, 2019*b*), but they focus on tracking the ice surface and ice bottom. Internal layer tracking is generally more difficult because there are many layers, the layers may be closely spaced, and the number of layers is not known. For this reason, algorithms designed for the surface and bottom only are not well suited for internal layer tracking without adaptation. Although a comprehensive intercomparison of existing algorithms on a common dataset would be of value, the focus of this present work is to introduce an internal layer tracking algorithm based on deep learning. A short version of this work is presented in Yari and others (2019) and also Yari and others (2020). A simulation result of internal ice layers based on deep learning is presented in Rahnemoonfar and others (2020).

The advancement of AI techniques in recent years has had a great impact on our approaches to data analysis. Deep learning, in particular, has shown great success in many areas of practical interest such as classification (Krizhevsky and others, 2012*a*; Szegedy and others, 2015*a*; Sheppard and Rahnemoonfar, 2017), object recognition (Girshick and others, 2014; Hariharan and others, 2014), counting (Rahnemoonfar and Sheppard, 2017*a*, 2017; Rahnemoonfar and others, 2019*a*) and semantic segmentation (Farabet and others, 2013; Mostajabi and others, 2015; Rahnemoonfar and others, 2018; Rahnemoonfar and Dobbs, 2019). Despite their progress, these algorithms are limited mainly to optical imagery. Non-optical sensors such as radars present a great challenge due to coherent noise in the data. The goal of this work is to test the capability of a deep neural network to track ice-sheet internal layers. We experiment with a deep learning architecture and training strategies that have shown great success in other areas of computer vision.

## 2. Related works

Several semi-automated and automated methods exist for surface and bottom tracking in radar images (Crandall and others, 2012; Lee and others, 2014, Mitchell and others; Mitchell and others 2013*a*; 2013*b*, Rahnemoonfar and others; Rahnemoonfar and others; Rahnemoonfar and others 2016; 2017*a*; 2017*b*). As mentioned above, tracking internal layers is a significantly different and generally more difficult task because of the large number of layers in close proximity and the number of layers is unknown. Panton (2014) describes a semi-automatic method for tracing internal layers in radio echograms based on the Viterbi algorithm. The algorithm was tested on a relatively short 423 line-km radar depth sounder dataset and showed promising results, but still relied on the use of manually labeled seed points. MacGregor and others (2015*b*) transform the images so that the layers are flattened by using an internal layer slope field. The flattened layers are then tracked with a simple snake algorithm that tracks the peak power column to column. The simple algorithm works well when the layers are contiguous (no gaps in the layer) and flat, but still requires some user input to seed the snake algorithm. Two methods are used to generate the slope fields. One is a phase coherent method that uses a matched filter for specular layer scattering to estimate the layer slopes. Although this algorithm works well for specular (coherent) layers, some layers, including those typically detected by the Snow Radar, are not coherent scatterers and therefore an algorithm relying on phase coherence from column to column is not likely to work well. MacGregor and others (2015*b*) also use a method that works on incoherent scatterers based on Sime and others (2011) which uses filtering, thresholding and feature extraction to estimate layer slopes from the slopes of the features. Using this process of flattening and the simple snake algorithm, the task took a couple years to complete and although many layers were tracked in the imagery that was analyzed, there are still untraced layers due to time limitations. de Paul Onana and others (2014) describe an algorithm that relies on surface flattening, filtering for horizontal features and automated peak finding to seed layers. The method is largely automated, but still required a manual step to index layers. The algorithm is generally not able to track all human-detectable layers and especially struggles with layers with larger slopes relative to the surface. Carrer and Bruzzone (2016) present a locally applied Viterbi algorithm which works well with ice sounder data collected on the Mars north polar ice cap. This method is fully automated after an initial training phase. To the best of our knowledge, there are no internal layer tracking algorithms that utilize deep learning techniques.

Most traditional approaches to edge and contour detection problems fundamentally make use of image spatial derivative operators. Since the derivative operators possess high-pass characteristics of the image, they can effectively enhance edges in an image and make them more pronounced. The downfall of the derivative operator, however, is that it is susceptible to noise. Now in order to reduce the sensitivity of derivative operators to noise, one can employ regularization filters, such as a Gaussian filter. Traditional edge detectors, such as Canny (1986) and Marr and Hildreth (1980), are prime examples of edge detectors that combine regularization with derivative operators.

Traditional edge detection techniques are based on handcrafted feature engineering and cannot be applied to a large-scale dataset. In recent years, there have been several deep learning techniques for contour detection in optical imagery (Shen and others, 2015; Bertasius and others, 2015; Xie and Tu, 2015; Liu and Lew, 2016; Liu and others, 2017). The pioneering deep learning techniques use fully connected layers so they are limited to images with fixed sizes. However, recent deep learning techniques focus more on fully convolutional networks and multi-scale features to extract both local and global information from the image. There are two main types of network structures to generate multi-scale features. The primary option is to resize the original image and pass it to the network and then combine the results. The more intelligent way is to extract multi-scale features through the hidden layers in a deep neural network; Holistically nested Edge Detection (HED) (Xie and Tu, 2015), Relaxed Deep Supervision (RDS) (Liu and Lew, 2016) and Richer Convolutional Features (RCF) (Liu and others, 2017) are among the second approach. The early layers in the deep neural network extract the local and detailed information while the last layers extract the global information. Since our radar data are noisy, we have implemented a multi-scale edge detection technique which can extract both local and global information and at the same time address the noise issue.

## 3. Dataset

We analyze Snow Radar (Rodriguez-Morales and others, 2018) images produced by the Center for Remote Sensing of Ice Sheets for NASA Operation IceBridge. The Snow Radar is a profiling instrument which produces vertical sounding images of snow layers over ice sheets and ice caps. The radar signal is sensitive to shallow annual snow density changes that occur in the top few tens of meters of ice due to the seasonal transitions from summer to winter; this density change results in a snow permittivity change at the interface that scatters the radar signal which is measured by the radar's receiver. For training our network we used the output of semi-supervised layer tracking from de Paul Onana and others (2014). The layers were quality controlled by a human analyst for Koenig and others (2016). This included correcting layers, adding missing layers and deleting layers which were in error or too difficult to interpret with sufficient confidence as annual layers. While the tracking is largely comprehensive, there are some layers that are not tracked. In some cases, layers were removed if they were thought to be caused by something other than the annual layer indicating the transition from summer to fall. In other cases, layers that could be tracked by the analyst may not have been due to perceived value in tracking the layer, difficulty to track the layer in question, and time constraints to complete the work. In total, we use 93700 line-km of images collected over Greenland during the 2012 field season that were tracked by Koenig and others (2016). The images we use have the same preprocessing steps applied as described in Koenig and others (2016) which includes filtering, decimation and surface flattening. The filtering operates on incoherent power detected data, helps to denoise the data and is analagous to multilooking. The 2012 field season includes coverage of most of Greenland and the dataset is well representative of Snow Radar image variability.

In one of our experiments we used synthetic data for training. The synthetic dataset models the layers as the superposition of many point targets. The scattered signal from each point target is represented by a weighted sinc function. The weight for each sinc function is a complex Gaussian random variable. The thickness of each layer represents the separation between interfaces and is generated by a smoothed Gaussian random process. The mean thickness or separation decreases slightly with depth and starts at 75 pixels for the first layer. The standard deviation of the random process is 10% of the mean thickness. The depth of a layer results from the summation of the thickness of the layer and all layers above it. For each column of the image, the scattered signal for each layer is created by summing the contribution of 100 point targets spread slightly in range. The spreading in range is generated from the summation of a Gaussian distribution and an exponential distribution to simulate the spread of the scatterers over several range bins as well as the backscatter fall-off from each layer as a function of cross-track incidence angle. The mean signal power of each layer follows an exponential decay with depth so that deeper layers have weaker signals on average. The images are power detected, filtered and surface flattened similar to the real radar data. The various parameters used to generate the data are chosen manually to create images that are similar to the Snow Radar data, but no effort is made to precisely match the Snow Radar image statistics. The synthetic images represent a first-order attempt and although they share some resemblance to the Snow Radar images, they are easily distinguished from them.

## 4. Convolutional neural networks

Convolutional neural networks (CNN) are a class of deep neural networks that mainly focus on analyzing imagery. CNN comprise various convolutional and pooling (subsampling) layers that can be compared to the visual system of humans. Generally, image data are fed into the network that constitutes an input layer and produces a vector of reasonably distinct features associated to object classes. From input to output layers there are many hidden layers including convolution layers, pooling layers and fully connected layers.

To provide understanding of the activities at each layer in a neural network, some explanation of the various processes is provided below.

**Convolution:** The main building block of a CNN is a convolutional layer. The parameters of this layer include a set of learnable filters (or kernels), which have a small receptive field, but spread through the full depth of the input. Every filter is convolved along the width and height of the input volume and produces a two-dimensional feature vector of that filter. All the feature vectors generated through different filters are stacked together along the depth dimension to form an output volume of the convolution layer (Bustince and others, 2009). The input to the layer is represented as a tensor and is used in the convolution operation along with the filter kernel. In order to control the number of parameters in convolutional layers, the parameters are shared.

**Activation:** Activation functions are used to normalize outputs of a neural network. A common activation is the Rectified Linear Unit, also known as ReLU. This function simply calculates the maximum of zero and the output of the neuron. A variation of this activation function is known as the Leaky ReLU. Here, the function calculates the maximum of the output of the neuron and a small constant value multiplied to the output.

**Pooling:** The pooling layer is a form of non-linear downsampling applied to reduce the size of the feature vectors generated through convolution. The idea is to reduce the number of parameters and the computation required and therefore to control overfitting. Several non-linear functions are available to perform pooling such as max pooling, min pooling and average pooling. Max pooling reports the maximum output within a rectangular neighborhood. Average pooling reports the average output of a rectangular neighborhood.The most common approach to apply pooling is between successive convolution layers.

**Fully connected layer:** After a series of convolutional and pooling layers, finally the abstract-level reasoning is performed using a fully connected layer. In this layer, neurons have full connections to all the activation outputs in the previous layer, similar to classical neural networks. There can be many fully connected layers before the final output layer.

**Regularization:** A common way to prevent over-fitting for a network is to employ the concept of dropout. In this method, neurons are randomly ignored in a layer. This essentially makes the network into an ensemble of many possible subnets. Following this method, there is less chance that one neuron will ever be relied upon too heavily.

**Classification:** For every possible class the network is trained to identify, there will be an output neuron. In order for these outputs to follow a probability distribution, a softmax function is applied after the final output of the network. When softmax is used, all the probabilities will add up to one.

Several deep neural network architectures such as AlexNet (Krizhevsky and others, 2012*b*), VGG-net (Simonyan and Zisserman, 2015) GoogLeNet (Szegedy and others, 2015*b*) and Resnet (He and others, 2016) have made such significant contributions to the field that they have become widely known standards. AlexNet was the pioneering deep neural network architecture that won the ImageNet Large Scale Visual Recognition Challenge (Deng and others, 2009) in 2012. VGGNet won the challenge in 2014 along with GoogLeNet. The VGGNet architecture is depicted in Figure 1. As you can see in this figure, the building blocks of VGGNet includes the convolution, pooling and fully connected layers that were explained previously.

## 5. Methodology

In this research, we employ a multi-scale learning model to trace snow boundary layers for real radar data. Multi-scale deep learning models are characterized by their use of the so-called side-output. Figure 2 provides an example of a multi-scale learning model where several side-outputs at different scales are extracted. Each side-output layer is associated with a classifier and an

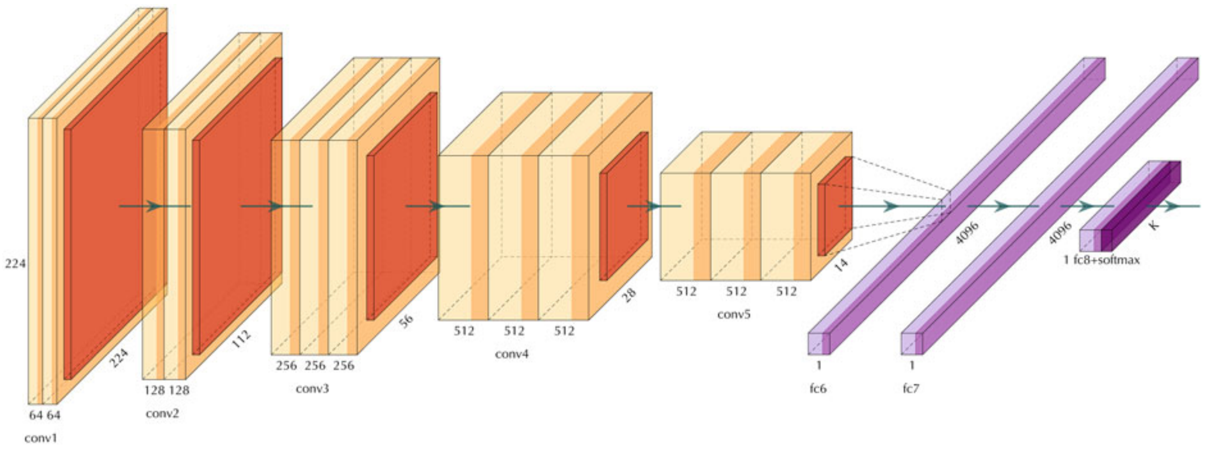

**Fig. 1.** The architecture of VGGNet. The orange layers show convolution layers, the red layers are pooling layers, and violet layers are fully connected layers.

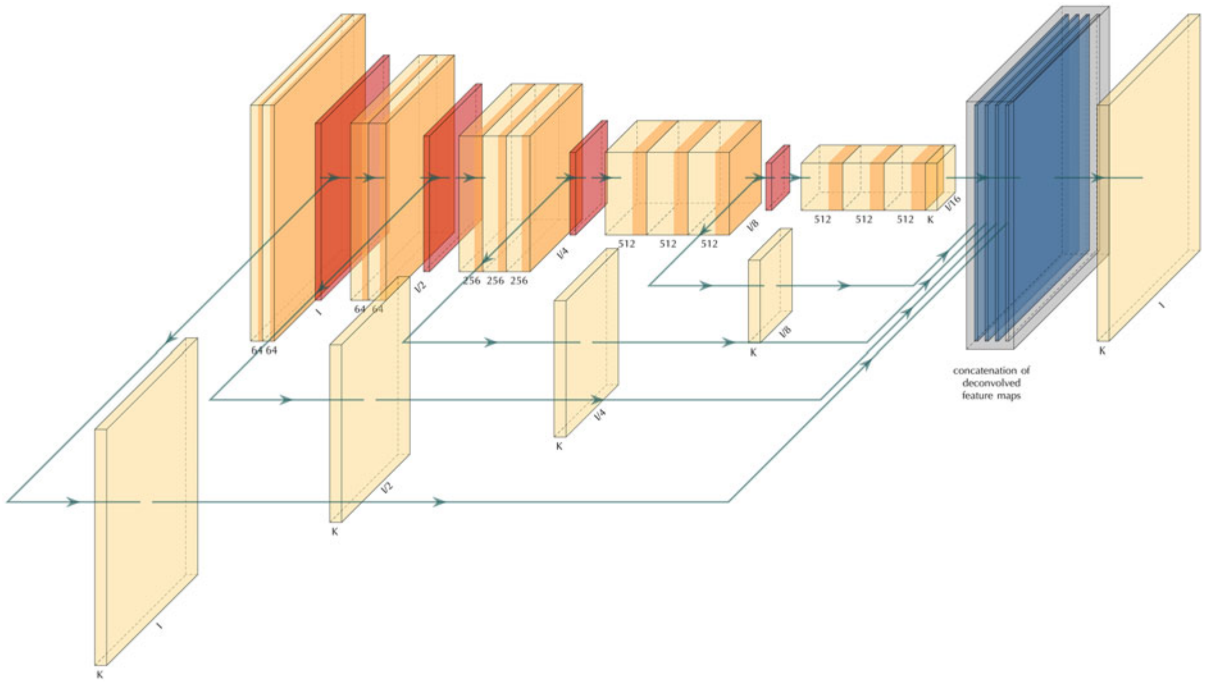

**Fig. 2.** The architecture of a multi-scale convolutional neural network. The orange layers show convolution layers, the red layers are pooling layers, and the blue layer is the fused layer.

objective function. The ensemble of the side-outputs generates a fuse prediction at the final stage and is associated with another objective function. Therefore, the model is equipped with several objective functions and learning components at different scales. In terms of the architecture, we base our construct on an existing architecture to take advantage of pre-trained parameters. This approach is particularly helpful for transfer learning purposes.The multi-scale architecture introduced in (Xie and Tu, 2015) takes the VGGNet (Fig. 1) as its parent architecture. Each of the first five blocks of the VGGNet follows the same architecture. It consists of a consecutive number of convolutional layers and activations, followed by a max-polling layer. The max-pooling layer will rescale the image based on the size of the max-pooling feature map. For a two by two max-pooling size, the size of the image will reduce to half the original size in each dimension.

Following the HED model of Xie and Tu (2015), we extract the side-outputs right before the max-pooling layers and drop the final fully connected layers of the VGGNet. This way, we end up with five side-outputs which we will fuse at the end to produce the final prediction. As mentioned previously, in addition to our final objective function, we have five other objective functions associated with each side-output. Moreover, the fuse weights are part of the learnable parameters; therefore, the model tries to get the best way of fusing the side-outputs as well. Another advantage of this approach is that it facilitates using the various sized input images.

In mathematical terms, we denote the original data in our training dataset by $X = \{X_n{:}n = 1N\}$, where $N$ is the size of the dataset; we also denote the corresponding boundary data by $Y = \{Y_n{:}n = 1N\}$. The HED model pulls out $M$ side-outputs by $Y_n^{(m)}$for $m = 1M$, and a final output of the weighted-fusion layer, denoted by $\widetilde{Y}_n$. The model includes $M$ image-level loss functions at each side-output layer, denoted by $\ell_m$ for side-output $m$, and a loss function at the fusion layer, denoted by $\ell_f$. We denote all parameters of the classifier associated with the $m^{th}$ side-

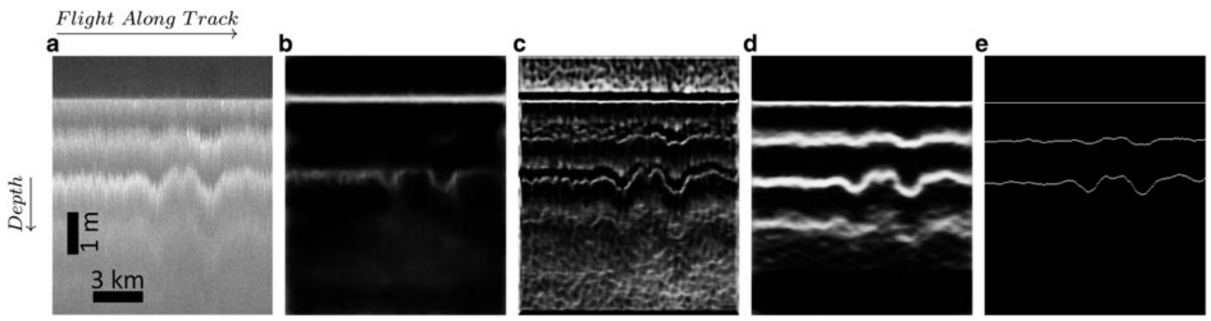

**Fig. 3.** (a) The original image. (b) The test result of our model trained on augmented BSDS500. (c) The test result of our model trained on synthetic data. (d) The result of the model trained on ICE2012. (e) The ground truth.

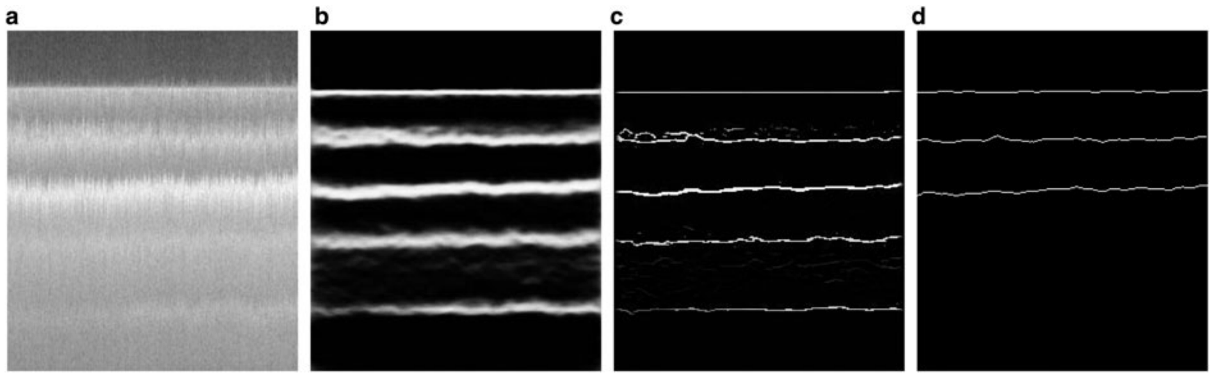

**Fig. 4.** A test result of the ICE experiment: (a) the original image, (b) the prediction result, (c) the non-maximal suppression result, (d) the ground truth.

output by $\theta_m$. Then the loss function $\ell_m$ is defined as a class-balanced cross-entropy function as in Eqn (1):

$$\ell_m = \ell_m(\theta_0, \theta_m) = -\beta \sum_{j\in Y^+} \log\left(Pr(y_j = 1|X;\, \theta_0, \theta_m)\right) - (1-\beta) \sum_{j\in Y_-} \log\left(Pr(y_j = 0|X;\, \theta_0, \theta_m)\right) \tag{1}$$

where $\theta_0$ represents all other standard network layer parameters, $Y_-$and $Y_+$ are the edge and non-edge labels respectively and $\beta = |Y_-|/|Y|$. The loss function for the final fusion layer is defined by

$$\ell_f(\theta, w) = CE(Y, \widetilde{Y}), \tag{2}$$

where $CE$ is a cross-entropy loss function that measures dissimilarities of the fused prediction and the ground truth label; $\theta = (\theta_0, \theta_1, , \theta_m)$, and $w = (w_1, , w_m)$ represents the fusion weights. Putting everything together, the goal is to minimize the following objective function via standard (back-propagation) stochastic gradient descent:

$$(\theta, w)^* = \text{argmin}\left(\ell_f + \sum_{m=1}^{M} \ell_m\right) \tag{3}$$

We use a mini-batch gradient descent that computes the gradient of the cost function with respect to the parameters $\theta$ for the entire training dataset: $\theta = \theta - \eta\nabla_\theta\mathcal{L}(\theta, x^I, y^I)$. Here we used the symbol $\theta$ for all parameters. This minimization approach is based on Nesterov accelerated gradient technique as discussed in Sutskever and others (2013):

$$v_t = \mu v_{t-1} + \eta\nabla_\theta\mathcal{L}(\theta_{t-1} - \mu v_{t-1}), \quad \theta_t = \theta_{t-1} - v_t \tag{4}$$

where $\mu \in [0,\ 1]$ is the momentum and $\eta > 0$ is the learning rate, see Sutskever and others (2013).

Input images are not resized for training or testing. Since we get the side-outputs right before applying the max pooling, the size of the first output matches with the original input size. But after applying the max pooling, the second side-output is half the size of the first side-output; likewise, each subsequent side-output is going to be half the size of the previous side-output. Therefore, each side-output is generated at a different scale.

In the training process, we have used the following parameters $\gamma = 0.1$, learning rate $\eta = 10^{-6}$ and the momentum $\mu = 0.9$. We have also used weight decay rate of $2 \times 10^{-4}$.

## 6. Experimental results

In this section we report our experimental results on our dataset. We refer to our real dataset of ice radar imagery, as ICE2012, which consists of 2360 training images and 260 test images. We refer to our synthetic dataset as SYNT_ICE, which consists of one thousand synthetic images, and use it for the training phase only. For pre-training and transfer learning models, we use the benchmark BSDS500 dataset (Martin and others, 2001). We present the results of three main experiments that we carried out.

(1) BSDS: We trained our model on augmented BSDS500.
(2) SYNT: We trained the model on the synthetic ice dataset (SYNT_ICE).
(3) ICE: We trained the model on ICE2012.

Figure 3 shows the qualitative results for the three experiments on a sample image. Figure 3a shows the input image to our network. In the first experiment, we trained our network on the BSDS500 benchmark dataset and its augmentation. The parameters of the CNN blocks of the model were transferred over

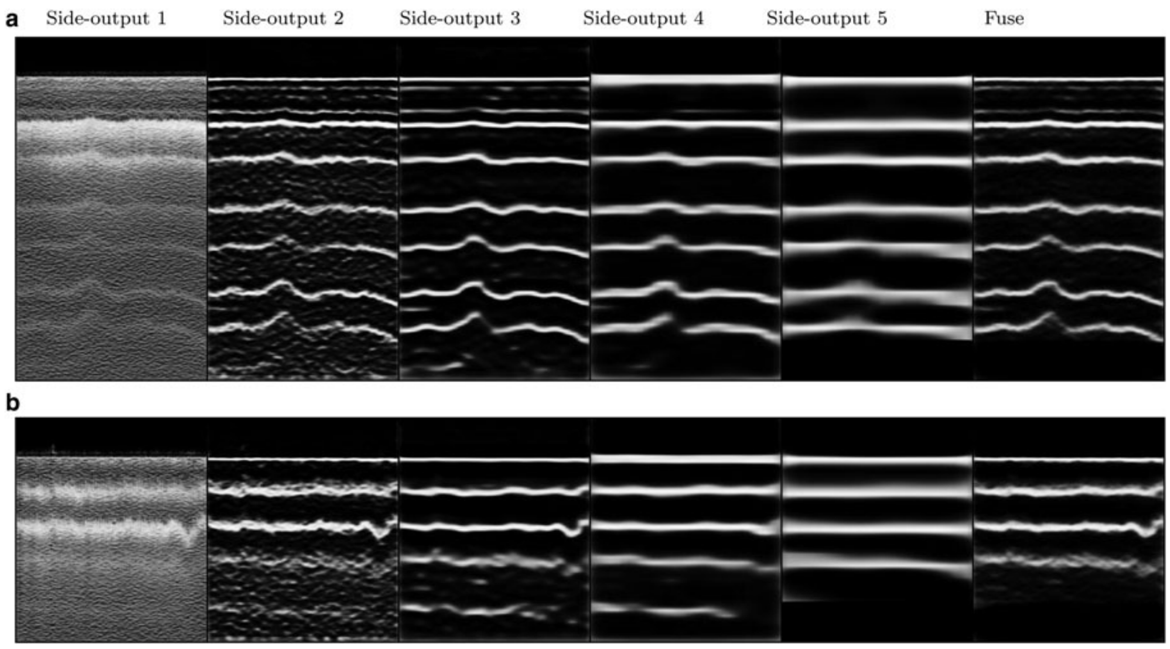

**Fig. 5.** From left to right: the first image is the first side-output which is the same size as the original image; the second image is the second side-output which is half the size of the first side-output; likewise the third side-output is half the size of the second side-output and so on. The utmost right image is the fusion of the five side-outputs.

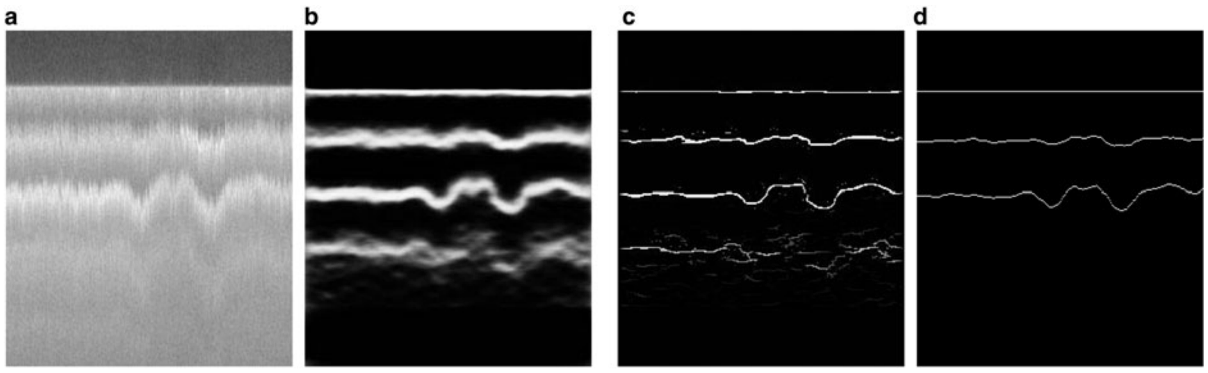

**Fig. 6.** A test result of the ICE experiment: (a) original image with sharp fluctuations in the layer boundaries, (b) the prediction result, (c) the non-maximal suppression result, (d) the ground-truth.

from the VGGNet trained on the ImageNet dataset. We used the same hyper parameters as used in Xie and Tu (2015) and trained it for 10 epochs. In the testing phase, we applied the trained model on our test dataset. Figure 3b shows the result of this experiment on one of the test images. As one can see in this experiment, we can only detect the top layer (surface) and hardly any internal layers are detected.We did some further experiments with transfer learning techniques, but all of them failed to converge; therefore, we have not included their results in this paper. For instance, we transferred VGG16 parameters and continued training the model on our own dataset, ICE2012. In another experiment, we transferred VGGNet parameters, continued training on BSDS500 with augmentation, and continued training it further on ICE2012. Both experiments failed to converge. In fact both algorithms diverged in very early stages.

The second experiment produced a better result. We trained the model on a synthetic dataset of 1000 images (SYNT_ICE). Figure 3c shows the result of this experiment on one of the test images. The synthetic data generation process is explained in the Dataset section. As mentioned before, the simulator has not been tuned to match the actual Snow Radar data properties and statistics, but these preliminary results suggest that if we can generate synthetic data with a noise and signal generator better matched to the Snow Radar data, we may be able to achieve good results.The third experiment is conducted on our real dataset, ICE2012, with a random initialization. We notice a considerable improvement in our results. Figure 3d shows the result of this experiment on one of the test images. Figure 3e shows the annotated ground truth contours side by side of the results of our three experiments.

Figure 4 shows another sample of our results. In this figure, in addition to the predicted results by our network (Fig. 4b), we have also shown the post-processing results after non-maximal suppression (Fig. 4c). Non-maximal suppression (NMS) (Dollár and Zitnick, 2014) results in thinned edge maps. As we can see in Figure 4, the original image has five different layers, but only the three top layers are annotated in the ground truth image. With our training strategy, our model has been able to detect all five layers.

As it was pointed out in the previous sections, HED model produces side-outputs in different scales. Figure 5 shows all side outputs of the model and the final fusion for two images. It is

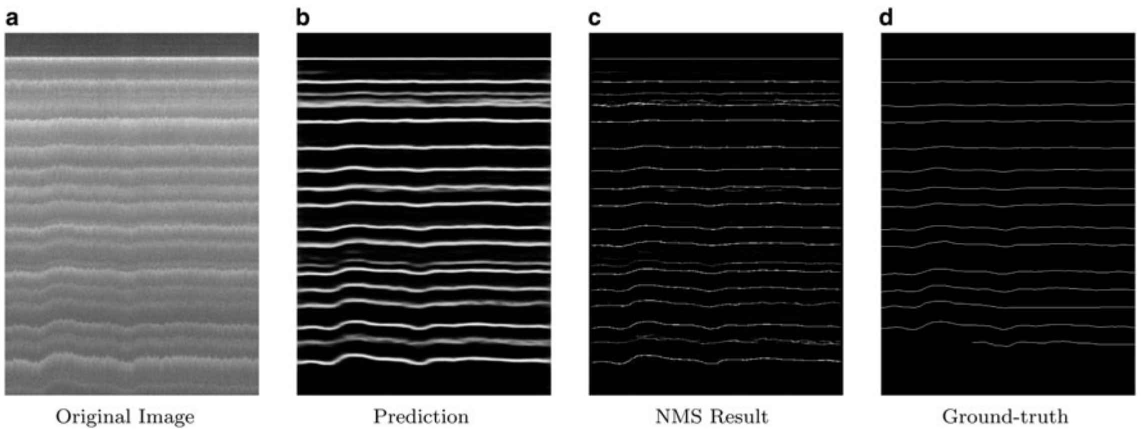

**Fig. 7.** Another sample where the image contains a high number of layer boundaries. The model is trained and tested on ICE2012; (a) is the original image; (b) is the prediction of the deep neural network; (c) is the post-processing results after the non-maximal suppression (NMS) and finally (d) is the ground-truth result.

**Table 1.** Evaluation results for the test dataset. The ICE column illustrates the result of our experiment where we trained and tested the model on the real dataset of ice images. ODS is the Optimal Dataset Scale, OIS is the Optimal Image Scale and AP is the Average Precision. This experiment provided the best results shown in bold fonts. The SYNT column: trained on the synthetic images dataset and tested on the real test data. The BSDS column: trained the model on the BSDS500 dataset and tested on the real ice images

| | BSDS | | | SYNT | | | ICE | | |
|---|---|---|---|---|---|---|---|---|---|
| | ODS | OIS | AP | ODS | OIS | AP | ODS | OIS | AP |
| Side 1 | 0.130 | 0.111 | 0.073 | 0.175 | 0.173 | 0.123 | 0.320 | 0.465 | 0.261 |
| Side 2 | 0.162 | 0.186 | 0.082 | 0.415 | 0.503 | 0.234 | 0.763 | 0.779 | 0.760 |
| Side 3 | 0.199 | 0.202 | 0.075 | 0.614 | 0.628 | 0.491 | 0.796 | 0.824 | 0.786 |
| Side 4 | 0.170 | 0.196 | 0.055 | 0.385 | 0.382 | 0.206 | 0.732 | 0.769 | 0.645 |
| Side 5 | 0.276 | 0.295 | 0.138 | 0.448 | 0.496 | 0.364 | 0.512 | 0.572 | 0.399 |
| Fuse | 0.139 | 0.164 | 0.040 | 0.292 | 0.379 | 0.217 | **0.815** | **0.854** | **0.815** |

apparent that the first side-outputs contain more details of the image while the later side outputs project the general structure of the image. The noise is more apparent in the first side-output while the last side-output is more enhanced.

Figure 6 provides a sample result of an image with more fluctuations in the layer boundaries. Compared to the ground truth labels, our method can detect all layers including layers that are difficult to track manually.

Figure 7 shows another example of how well the model works in the case of images that contain many internal layers. In this figure, the original image contains many layers. Towards the bottom of the image, the ground-truth labeled data miss tracking a couple of layers but our model (Figs b and c) is able to detect them. However, our model does fail to predict the very last bottom layer.

## 6.1. Quantitative results

To evaluate our results quantitatively, we have used three different metrics; The Optimal Dataset Scale (ODS) or best F-measure on the dataset for a fixed scale, the Optimal Image Scale (OIS) or aggregate F-measure on the dataset for the best scale in each image, and the Average Precision (AP) on the full recall range (equivalently, the area under the precision-recall curve) (Arbelaez and others, 2011). Table 1 shows the quantitative results for all three experiments with the three aforementioned metrics (ODS, OIS, AP) for all side-outputs as well as the fuse layer. The first column in Table 1 presents the results for transfer learning in which the model is trained on the BSDS500 dataset, and tested on the real test data. The second column shows the result of the model trained on the synthetic ice data (SYNT_ICE), and tested on the real test data. The final column shows the result of training the model on real data and evaluated on the real test data. As we can see in Table 1, the third experiment (ICE) shows the most accurate results compared to the other two experiments (BSDS and SYNT). Also the fusion of all side-outputs (Fuse) presents the most accurate result compared to each individual side-output.

**Table 2.** Comparison between a traditional edge detection method such as Canny, with the deep learning method trained in different ways

| Methods | ODS | OIS | AP |
|---|---|---|---|
| Canny | 0.229 | 0.240 | 0.072 |
| BSDS | 0.139 | 0.164 | 0.040 |
| SYNT | 0.292 | 0.379 | 0.217 |
| ICE | **0.815** | **0.854** | **0.815** |

Table 2 shows the result of the three experiments in comparison to a traditional edge detection technique (Canny) using the three different metrics. We can see in this table that the deep learning results, especially the ICE experiments, show more accurate results compared to the traditional edge detection technique (Canny). We also used the precision-recall curve where the green line shows the value for F-measure. Figure 8 shows the precision-recall curve for all side-outputs for our ICE experiment. We can see in this figure that the Fuse result depicts a closer curve to the labeled data (green point).

Figure 9 shows the precision-recall curve for all three experiments in addition to the traditional edge detection technique. Here also the ICE experiment presents a closer curve to the labeled data.

## 7. Conclusion and discussion

In this work, we have studied a multi-scale deep learning model and various approaches to implement it for detecting ice layers in radar imagery. It is important to note that most of the well-known deep learning approaches work very well on optical images, but can not produce acceptable results for non-optical sensors especially in the presence of noise. The fact that deep learning models are not robust with respect to noise is discussed in various works (Heaven, 2019). In our experiments we have shown that transfer learning approaches do not work well for radar images, while training from scratch yields far better results. However, the latter requires annotated data provided by the domain experts. One way to avoid this would be to generate synthetic data. Although the synthetic data used for training in this

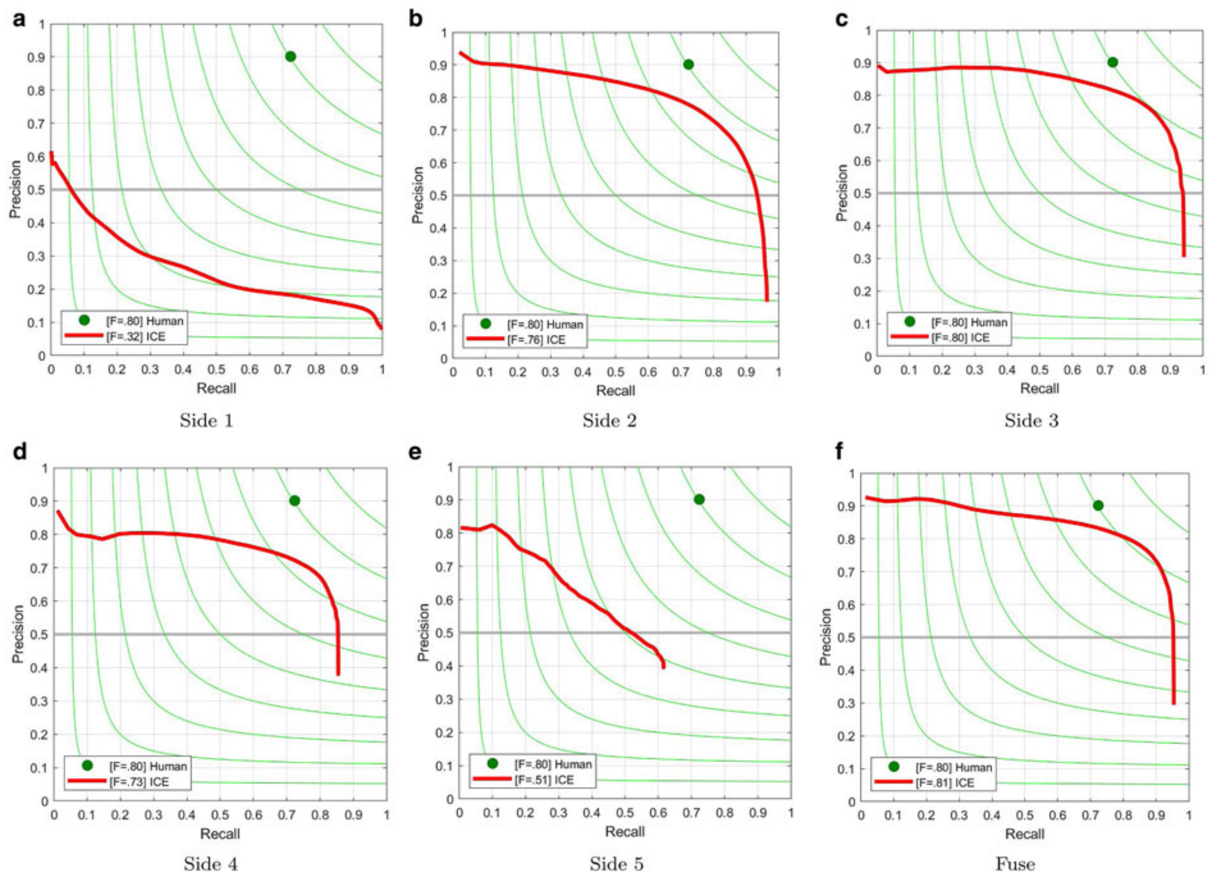

**Fig. 8.** Evaluation of the ICE experiment: precision-recall curve for each side-output and their fusion. (a) Side 1, (b) Side 2, (c) Side 3, (d) Side 4, (e) Side 5, (f) Fuse.

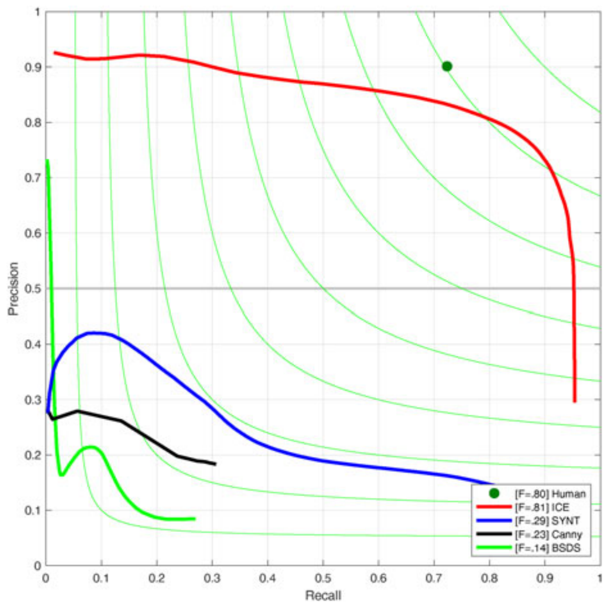

**Fig. 9.** Evaluation comparison: precision-recall curves of various methods applied on our test set: the result of Canny edge detection method, the result of the deep learning model trained on ICE2012 dataset (ICE), BSDS500 dataset (BSDS) and Synthetic data (SYNT).

work only loosely match the actual snow radar, the results indicate that synthetic data could be successfully used for training. Future work should explore training with synthetic data which matches the noise and signal statistics of the actual snow radar data. In the future, we plan to combine AI and physical models to expand the simulated dataset and therefore better train our network. We also plan to develop advanced noise removal technique based on deep learning. Calculating the actual thickness of ice layers from the neural network is also another direction of our research.

**Acknowledgments.** This work is supported by NSF BIGDATA awards (IIS-1838230, IIS-1838024), IBM and Amazon. Lynn Montgomery assisted in the use of the ground truth data. We also would like to thank Annals of Glaciology Associate Chief Editor Dr. Dustin Schroeder for handling our paper.

## References

**Arbelaez P, Maire M, Fowlkes C and Malik J** (2011) Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33**(5), 898–916.

**Berger V**, and 6 others (2019) Automated ice-bottom tracking of 2d and 3d ice radar imagery using viterbi and trw-s. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **12**(9), 3272–3285.

**Bertasius G, Shi J and Torresani L** (2015) Deepedge: a multi-scale bifurcated deep network for top-down contour detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4380–4389.

**Bingham RG, Eisen O, Karlsson N, MacGregor JA, Ross N and Young DA** (2019) Antarchitecture: an international project to use antarctic englacial layering to interrogate stability of the antarctic ice sheets.

**Bustince H, Barrenechea E, Pagola M and Fernández J** (2009) Interval-valued fuzzy sets constructed from matrices: application to edge detection. *Fuzzy Sets and Systems* **160**(13), 1819–1840, ISSN 0165-0114.

**Canny J** (1986) A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**, 679–698.

**Carrer L and Bruzzone L** (2016) 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* **55**(2), 962–977.

**Cavitte MG**, and 7 others (2016) Deep radiostratigraphy of the east antarctic plateau: connecting the dome c and vostok ice core sites. *Journal of Glaciology* **62**(232), 323–334.

**Crandall DJ, Fox GC and Paden JD** (2012) Layer-finding in radar echograms using probabilistic graphical models. *IEEE 2012 21st International Conference on Pattern Recognition (ICPR)*, 1530–1533.

**Deng J**and 5 others (2009) Imagenet: a large-scale hierarchical image database. *2009 IEEE conference on computer vision and pattern recognition*, 248–255.

**de Paul Onana V, Koenig LS, Ruth J, Studinger M and Harbeck JP** (2014) A semiautomated multilayer picking algorithm for ice-sheet radar echograms applied to ground-based near-surface data. *IEEE Transactions on Geoscience and Remote Sensing* **53**(1), 51–69.

**Dollár P and Zitnick CL** (2014) Fast edge detection using structured forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **37**(8), 1558–1570.

**Farabet C, Couprie C, Najman L and LeCun Y** (2013) Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**(8), 1915–1929.

**Girshick R, Donahue J, Darrell T and Malik J** (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 580–587.

**Hariharan B, Arbeláez P, Girshick R and Malik J** (2014) *Simultaneous Detection and Segmentation*. Cham: Springer International Publishing, 297–312.

**He K, Zhang X, Ren S and Sun J** (2016) Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

**Heaven D** (2019) Why deep-learning ais are so easy to fool. *Nature* **574**(7777), 163.

**IMBIE-Team** and others (2019) Mass balance of the greenland ice sheet from 1992 to 2018. *Nature* **579**(7798), 233–239.

**IPCC** (2014*a*) *IPCC AR5 WG1 Summary for Policymakers*. Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press.

**IPCC** (2014*b*) *IPCC AR5 WG2 Summary for Policymakers*. Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press.

**Kamangir H, Rahnemoonfar M, Dobbs D, Paden J and Fox G** (2018) Deep hybrid wavelet network for ice boundary detection in radar imagery. *IEEE IGARSS 2018–2018 IEEE International Geoscience and Remote Sensing Symposium*, 3449–3452.

**Koenig LS and 9 others** 2016) Annual greenland accumulation rates (2009–2012) from airborne snow radar. *The Cryosphere* **10**(4), 1739–1752.

**Krizhevsky A, Sutskever I and Hinton GE** (2012*a*) Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems, vol.* **25**, 1097–1105, Curran Associates, Inc.

**Krizhevsky A, Sutskever I and Hinton GE** (2012*b*) Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

**Lee S, Mitchell J, Crandall DJ and Fox GC** (2014) Estimating bedrock and surface layer boundaries and confidence intervals in ice sheet radar imagery using mcmc. *2014 IEEE International Conference on Image Processing (ICIP)*, 111–115.

**Liu Y, Cheng MM, Hu X, Wang K and Bai X** (2017) Richer convolutional features for edge detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3000–3009.

**Liu Y and Lew MS** (2016) Learning relaxed deep supervision for better edge detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 231–240.

**MacGregor JA**, and 9 others (2015*a*) Radiostratigraphy and age structure of the greenland ice sheet. *Journal of Geophysical Research: Earth Surface* **120**(2), 212–241.

**MacGregor JA**, and 9 others (2015*b*) Radiostratigraphy and age structure of the greenland ice sheet. *Journal of Geophysical Research: Earth Surface* **120**(2), 212–241.

**MacGregor JA**, and 9 others (2016) A synthesis of the basal thermal state of the greenland ice sheet. *Journal of Geophysical Research: Earth Surface* **121** (7), 1328–1350.

**Marr D and Hildreth E** (1980) Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences* **207**(1167), 187–217.

**Martin D, Fowlkes C, Tal D and Malik J** (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. *Proc. 8th Int'l Conf. Computer Vision*, vol. **2**, 416–423.

**Medley B**, and 9 others (2014) Constraining the recent mass balance of pine island and thwaites glaciers, west antarctica, with airborne observations of snow accumulation. *The Cryosphere* **8**(4), 1375–1392.

**Mitchell JE, Crandall DJ, Fox G and Paden J** (2013*a*) A semi-automatic approach for estimating near surface internal layers from snow radar imagery. *IGARSS*, 4110–4113.

**Mitchell JE, Crandall DJ, Fox GC, Rahnemoonfar M and Paden JD** (2013*b*) A semi-automatic approach for estimating bedrock and surface layers from multichannel coherent radar depth sounder imagery. *SPIE Remote Sensing*, 88921–88926, International Society for Optics and Photonics.

**Mostajabi M, Yadollahpour P and Shakhnarovich G** (2015) Feedforward semantic segmentation with zoom-out features. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3376–3385.

**Panton C** (2014) Automated mapping of local layer slope and tracing of internal layers in radio echograms. *Annals of Glaciology* **55**(67), 71–77.

**Rahnemoonfar M, Abbassi A, Paden J and Fox GC** (2017*a*) Automatic ice thickness estimation in radar imagery based on charged particle concept. *IEEE International Geoscience and Remote Sensing Symposium*, 3743–3746.

**Rahnemoonfar M and Dobbs D** (2019) Semantic segmentation of underwater sonar imagery with deep learning. *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 9455–9458.

**Rahnemoonfar M, Dobbs D, Yari M and Starek MJ** (2019*a*) Discountnet: Discriminating and counting network for real-time counting and localization of sparse objects in high-resolution UAV imagery. *Remote Sensing* **11**(9), 1128.

**Rahnemoonfar M, Fox GC, Yari M and Paden J** (2017*b*) Automatic ice surface and bottom boundaries estimation in radar imagery based on level-set approach. *IEEE Transactions on Geoscience and Remote Sensing*, 55(9), 5115–5122.

**Rahnemoonfar M, Johnson J and Paden J** (2019*b*) Ai radar sensor: Creating radar depth sounder images based on generative adversarial network. *Sensors* **19**(24), 5479.

**Rahnemoonfar M, Johnson J and Paden J** (2020) Radar sensor simulation with generative adversarial network. *IGARSS 2020–2020 IEEE International Geoscience and Remote Sensing Symposium (to appear).*

**Rahnemoonfar M, Robin M, Miguel MV, Dobbs D and Adams A** (2018) Flooded area detection from UAV images based on densely connected recurrent neural networks. *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 3743–3746.

**Rahnemoonfar M and Sheppard C** (2017*a*) Deep count: fruit counting based on deep simulated learning. *Sensors* **17**(4), 905.

**Rahnemoonfar M and Sheppard C** (2017*b*) Real-time yield estimation based on deep learning. *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, vol. **10218**, 1021809, International Society for Optics and Photonics.

**Rahnemoonfar M, Yari M and Fox GC** (2016) Automatic polar ice thickness estimation from sar imagery. *SPIE Defense+ Security*, 982902–982902, International Society for Optics and Photonics.

**Rignot E**, and 5 others (2019) Four decades of antarctic ice sheet mass balance from 1979–2017. *Proceedings of the National Academy of Sciences* **116**(4), 1095–1103.

**Rodriguez-Morales F**, and 8 others (2018) Radar systems for ice and snow measurements onboard manned and unmanned aircraft. *IEEE Latin America Transactions* **16**(9), 2473–2480.

**Shen W, Wang X, Wang Y, Bai X and Zhang Z** (2015) Deepcontour: a deep convolutional feature learned by positive-sharing loss for contour detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3982–3991.

**Shepherd A**, and 9 others (2018) Mass balance of the antarctic ice sheet from 1992 to 2017. *Nature* **558**, 219–222.

**Sheppard C and Rahnemoonfar M** (2017) Real-time scene understanding for UAV imagery based on deep convolutional neural networks. *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2243–2246.

**Sime LC, Hindmarsh RC and Corr H** (2011) Automated processing to derive dip angles of englacial radar reflectors in ice sheets. *Journal of Glaciology* **57** (202), 260–266.

**Simonyan K and Zisserman A** (2015) Very deep convolutional networks for large-scale image recognition. *ICLR.*

**Sutskever I, Martens J, Dahl G and Hinton G** (2013) On the importance of initialization and momentum in deep learning. *International Conference on Machine Learning*, 1139–1147.

**Szegedy C**, and 8 others (2015*a*) Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9.

**Szegedy C**, and 8 others (2015*b*) Going deeper with convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

**Xie S and Tu Z** (2015) Holistically-nested edge detection. *Proceedings of the IEEE International Conference on Computer Vision*, 1395–1403.

**Xu M, Fan C, Paden JD, Fox GC and Crandall DJ** (2018) Multi-task spatio-temporal neural networks for structured surface reconstruction. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1273–1282.

**Yari M**, and 5 others (2019) Smart tracking of internal layers of ice in radar data via multi-scale learning. *2019 IEEE International Conference on Big Data (Big Data)*, 5462–5468.

**Yari M, Rahnemoonfar M and Paden J** (2020) Snow radar layer tracking using iterative neural network approach. *IGARSS 2020–2020 IEEE International Geoscience and Remote Sensing Symposium ( to appear).*