Towards Generalizable Deep Learning Models for Radar Echogram Layer Tracking.

By

Oluwanisola Ibikunle

Submitted to the graduate degree program in Department of Electrical Engineering and Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Co-Chair: Dr. John Paden

Co-Chair: Dr. Shannon Blunt

Dr. Carl Leuschen

Dr. James Stiles

Dr. Christopher Depcik

Date Defended: August 27, 2024

The dissertation committee for Oluwanisola Ibikunle certifies that this is the approved version of the following dissertation:

Towards Generalizable Deep Learning Models for Radar Echogram Layer Tracking.

_____

Co-Chair: Dr. John Paden

_____

Co-Chair: Dr. Shannon Blunt

Date Accepted: August 27, 2024

Abstract

The accelerated melting of ice sheets in Greenland and Antarctica, driven by climate warming, is significantly contributing to global sea level rise. To better understand this phenomenon, airborne radars have been deployed to create echogram images that map snow accumulation patterns in these regions. Utilizing advanced radar systems developed by the Center for Remote Sensing and Integrated Systems (CReSIS), around 1.5 petabytes of climate data have been collected. However, extracting ice-related information, such as accumulation rates, remains limited due to the largely manual and time-consuming process of tracking internal layers in radar echograms. This highlights the need for automated solutions.

Machine learning and deep learning algorithms are well-suited for this task, given their near-human performance on optical images. The overlap between classical radar signal processing and machine learning techniques suggests that combining concepts from both fields could lead to optimized solutions.

In this work, we developed custom deep learning algorithms for automatic layer tracking (both supervised and self-supervised) to address the challenge of limited annotated data and achieve accurate tracking of radiostratigraphic layers in echograms. We introduce an iterative multi-class classification algorithm, termed "Row Block," which sequentially tracks internal layers from the top to the bottom of an echogram based on the surface location. This approach was used in an active learning framework to expand the labeled dataset. We also developed deep learning segmentation algorithms by framing the echogram layer tracking problem as a binary segmentation task, followed by post-processing to generate vector-layer annotations using a connected-component 1-D layer-contour extractor.

Additionally, we aimed to provide the deep learning and scientific communities with a large, fully annotated dataset. This was achieved by synchronizing radar data with outputs from a

regional climate model, creating what are currently the two largest machine-learning-ready Snow Radar datasets available, with 10,000 and 50,000 echograms, respectively

Acknowledgments

I am deeply indebted to my advisor, Dr. John Paden, for his unwavering guidance, mentorship, and unwavering support throughout my doctoral journey. His invaluable insights and encouragement have been instrumental in my academic growth and success. I am also grateful to Dr. Shannon Blunt, my official advisor, for his tutelage over the years. His adaptive signal processing class contributed greatly to my choice of research and dissertation. I also would like to thank Dr Carl Leuschen, the director of the Center for Remote Sensing and Integrated Systems (CReSIS), for his invaluable support and mentorship.

I would like to express my sincere gratitude to Dr. Christopher Depcik, the external committee member, for his insightful feedback and constructive criticism. His expertise and guidance have significantly improved the quality of my research.

I am eternally grateful to my parents and siblings for their unconditional love, support, and encouragement. Their belief in me has been a constant source of motivation. Finally, I would like to express my deepest gratitude to my loving wife for her unwavering support, patience, and understanding. Her constant presence and encouragement have made this journey possible.

Table of Contents

List of Figures

List of Tables

Chapter 1: INTRODUCTION

1-1 Background

The last decades have witnessed an increase in contemporary global warming which epitomizes the steep impact of anthropogenic activities on world climate. The far-reaching consequences of global warming has prompted many international bodies such as the World Health Organization and United Nations to unanimously identify it as the single biggest health threat facing humanity [1], [2], [3]. The Intergovernmental Panel on Climate Change (IPCC) forecasts a global temperature rise of 1.5°C to 4.8°C by 2100 relative to pre-industrial levels [4], [5]. The consequent accelerated polar ice melt and increased discharge into the ocean over the years poses a significant threat of global sea-level rise, which has been estimated to increase by an average of 14-18 inches along the Gulf Coast of the United States by 2050 [6]. Addressing this issue is imperative to mitigate the severe impacts, including extreme weather events and widespread coastal flooding, which could detrimentally affect millions of individuals worldwide.

As a result, polar ice sheets in Greenland and Antarctica have been the focus of a plethora of research since the beginning of the 21st century. In recent years, surface mass balance (SMB) processes have been identified as the primary driver of increased Greenland Ice Sheet (GrIS) mass loss. However, the intricate relationship between accumulation and surface melt introduces uncertainties in accurately determining annual accumulation rates [7], [8], [9]. Since accumulation varies across the ice sheets, a precise method of estimating the annual accumulation rate is crucial to capture the catchment-wide spatiotemporal patterns required by scientific climate models to accurately predict future sea-level rise [9], [10].
Traditionally, snow accumulation rates and other ice phenomenology are derived from ice cores which are obtained by drilling ice columns and shallow pits across the polar ice sheets to provide

detailed measurements at one location. However, their inherent sparsity and high operational costs make it challenging to capture catchment-wide accumulation rates. Attempts to interpolate in-situ measurements only introduce further uncertainties due to the high variability in local accumulation rates [11], [12]. To address this limitation, airborne radars have been developed to track the isochronous snow layers at a much larger spatial scale and relatively lower cost. These radar systems offer superior subsurface mapping capabilities compared to other remote sensing methods such as space-borne radars.

The Center for Remote Sensing and Integrated Systems (CReSIS) at the University of Kansas has developed a suite of radar systems for non-invasive monitoring of the changes in the thickness and structures of these polar ice sheets over time [13], [14], [15], [16], [17], [18], [19], [20]. Over a span of 3 decades, science missions using these radars have been conducted over Antarctica and the Arctic including icy regions in Alaska, Greenland, and Canada. As a result, a large repository of snow and ice data has been collected using a wide range of radar systems. Figure 1-1 shows the map of Greenland and the coverage of CReSIS missions over the ice sheet. This extensive repository of radar data houses a wealth of historical and contemporary climate information, providing a rich source of high-resolution spatial and temporal data essential for enhancing the accuracy of scientific weather models.

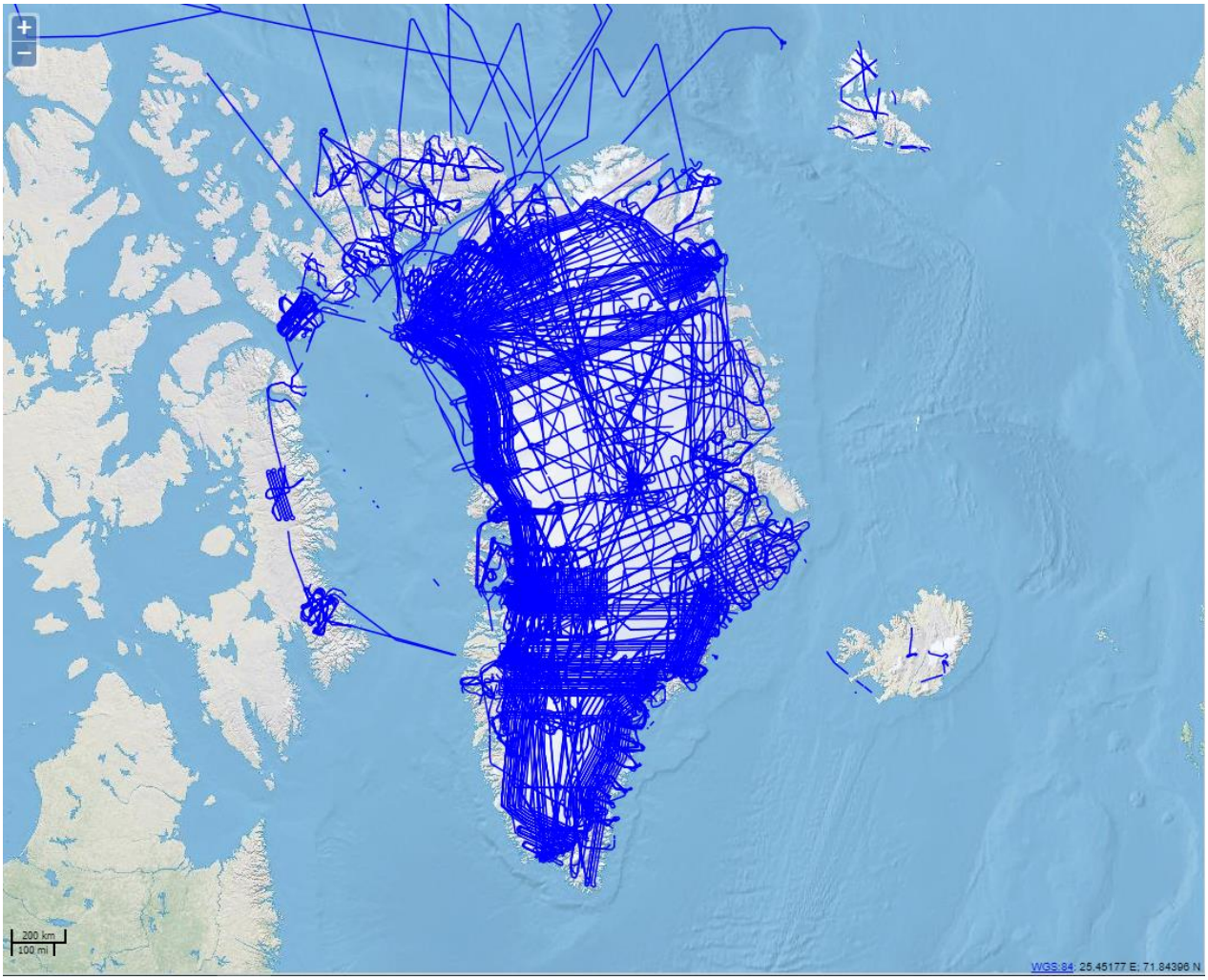


Figure 1-1 *Map of Greenland showing survey flight lines currently available at https://openpolarradar.org/ for one of the radar systems. The blue lines represent the flight path along which subsurface radar data has been collected.*

To further investigate how climate warming affects different sections of the polar ice sheets and the interactions of these effects, several radar systems with specific hardware design and capabilities have been created to study identified sections of the ice column ranging from the top layers of recent snow fall to the underlying bedrock several kilometers under the surface. One such system is the ultra-wideband (UWB) Snow Radar, which captures annual snow accumulation in the top firn layers of polar ice sheets. Its lower operating elevation and large

bandwidth enable it to achieve a vertical resolution that is fine enough to discriminate isochronous layers formed from annual accumulation. The radar echograms produced from these airborne measurements reveal annual accumulation stratigraphy [17], [18] which contains information needed to estimate annual accumulation rates [19], [20].

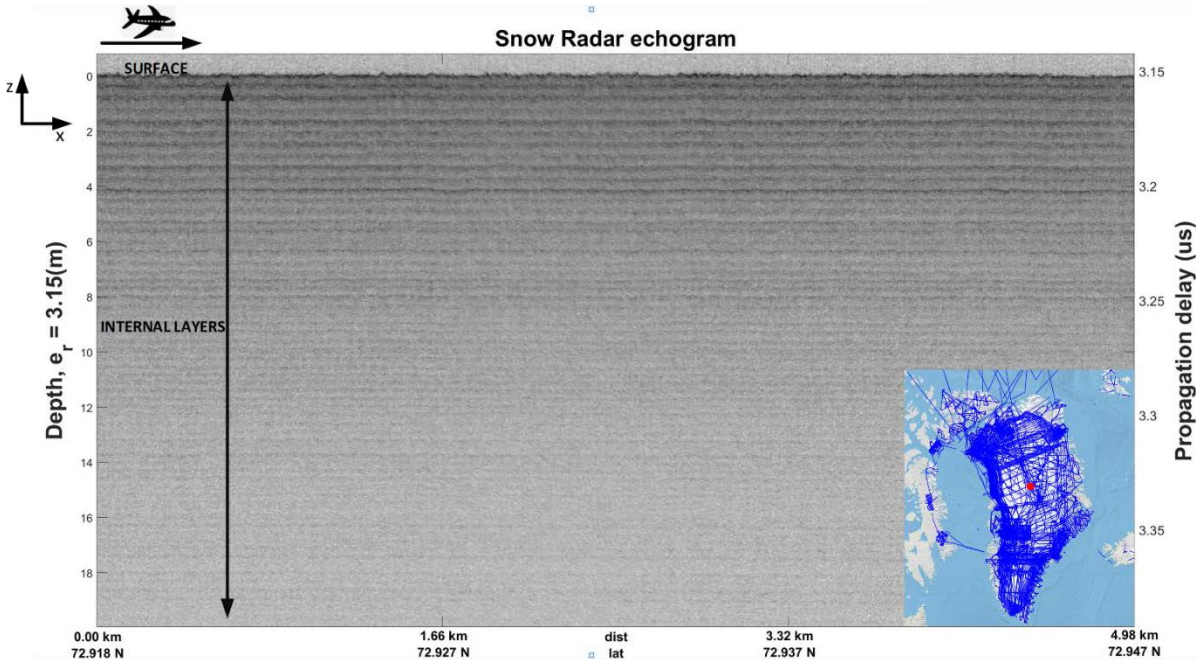


Figure 1-2: *Example Snow Radar echogram from the summit in Greenland showing annual snow layering. Inset is a map of Greenland with the red dot marking the center of Greenland where the echogram data was collected.*

Figure 1-2 is an example of a Snow Radar echogram created from data collected from a 5 km transect at the center of the Greenland ice sheet. Details about radar echograms and how they are created is delayed to the discussion in Section 2-2. Due to chronologically different deposition ages, the snow layers seen in the echogram are a result of contrasting physical properties of the snow layers such as snow density, snow grain size, etc. These layer property changes result in dielectric contrasts that cause scattering that the radar detects. In the echogram image, the

"surface" is the air-snow interface after which are several "internal" roughly annual snow layers. Each of these snow layers needs to be individually identified and accurately tracked to infer the accumulation rates for the mapped geolocations.

Figure 1-3 is an enlarged view of the echogram in Figure 1-2 magnifying the view of the top few internal layers corresponding to about a decade of annual snow fall assuming each layer represents an annual layer. To correctly estimate the annual accumulation rate in the location captured by this echogram, it is important to accurately track each pixel of each layer to obtain the propagation delay to the layer.

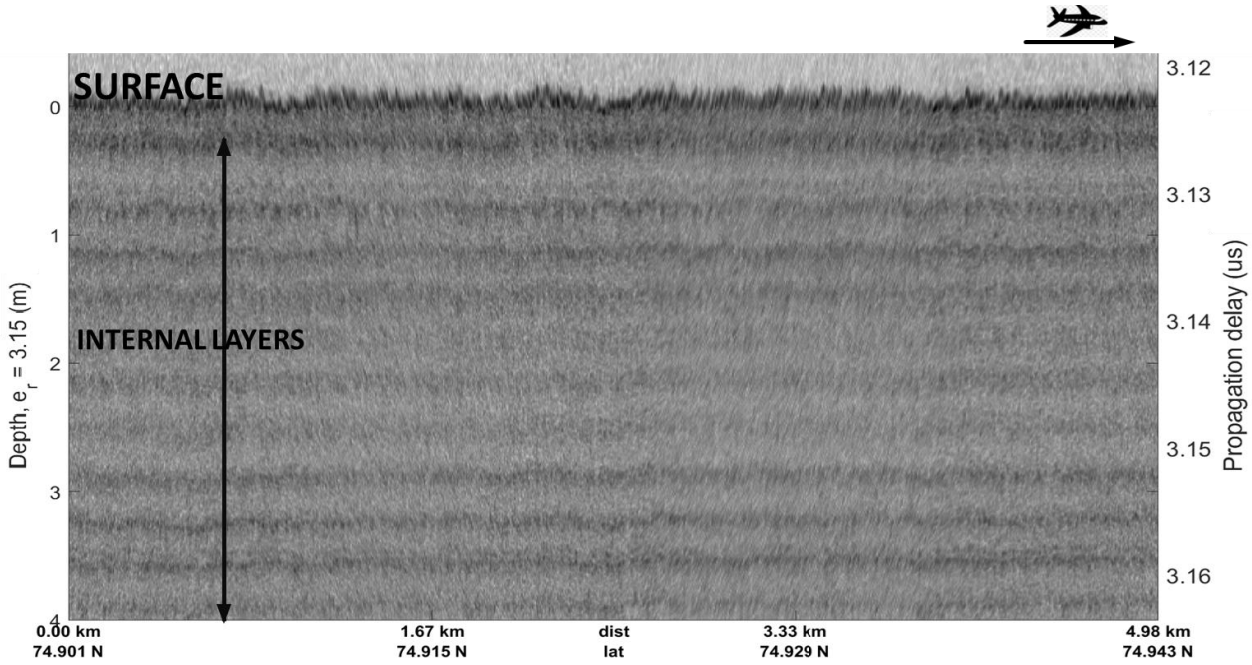


Figure 1-3 *Zoomed in image of earlier echogram to show the internal layers*

Concretely, to estimate the accumulation between successive years, the difference in the radar two-way travel time between adjacent layers is combined with the density-depth-age profile of the location to infer the snow accumulation rate. To illustrate how the accumulation rate is estimated, we provide an approximate estimate of the accumulation between the first two layers in the echogram shown in Figure 1-3. The mean radar propagation delay to the first layer (the

surface) is about 3.122 μs and to the second layer (first internal layer) is about 3.125 μs giving a difference of 0.003 μs. Assuming a constant relative permittivity of $\varepsilon_r = 1.35$ at the captured geolocation assumed to be constant for the top few meters of snow depth, the annual accumulation is estimated using Equation (1) below:

$$d = \frac{c\,\Delta\tau}{2\sqrt{\varepsilon_r}} = 0.38m \quad (1)$$

where $c = 3.00 \times 10^8$ m/s is the speed of light in a vacuum,

$\Delta\tau$ is the propagation delay difference,

ε_r is the assumed relative permittivity in snow and d is the accumulation depth in meters

Given that this data was collected in the year 2012, the tracked surface corresponds to the spring 2012 surface and the immediate lower layer corresponds to the summer-fall 2011 transition. Hence, an approximate mean accumulation of 0.38 m occurred in the mapped 5 km transect shown in the echogram between 2011 and 2012. This estimation process can be repeated for each of the (more than 50) consecutive layers in the full echogram in Figure 1-2. Typically, a precise accumulation estimation is done on a finer spatial resolution scale of a few meters by using the propagation delay for each column in the echogram image.

The accumulation estimate can be combined with the age difference between the layers to estimate the accumulation rate. This information is critical for tuning existing scientific climate models who otherwise suffer from limited spatial data coverage for ground truth. Assimilating radar derived estimation will not only improve the accuracy of the models in predicting future climate changes but also reinforce confidence in the model's predictions by reducing the existing uncertainties.

1-2 Summary of problem statement

The importance of precise tracking and delineation of individual snow layers in echogram images, as exemplified in the illustration above, underscores the necessity of obtaining annual accumulation at very fine spatiotemporal resolution from collected radar data to improve scientific climate weather predictions. However, the larger percentage of existing echograms are yet to be tracked due to the lack of efficient tracking methods. Tracking of layers is primarily done manually due to the complexity of the tracking process. Manual tracking of echogram layers is a tedious and error-prone process, requiring significant investment of the annotator's time.

In recent times, several semi-automated tools have been developed but this still does not scale well to the large data that has been collected since humans are still involved in the annotation process. Artificial intelligence, specifically machine learning and deep learning algorithms, hold great potential for this problem given their well-reported excellent performance in the optical image domain. Deep learning algorithms are currently the state-of-the-art algorithms for performing classification, object detection, and semantic segmentation on optical image data and hitherto complex problems like speech recognition [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]. Unlike traditional signal processing algorithms and probabilistic graphical models, which have demonstrated limited performance and require redesign for different datasets due to varying accumulation patterns, deep learning algorithms possess the potential for broad generalization when properly trained [31], [32]. Moreover, the significant overlap between classical radar signal processing and machine learning techniques [33], [34], [35], [36], suggests that a fusion of concepts from both fields can lead to optimized solutions for the problem of mapping the rapidly varying spatiotemporal accumulation patterns over polar ice sheets. In this work, we explored multiple machine learning and deep learning approaches to create automatic

layer trackers that can uniquely identify and track snow layers in the Snow Radar echogram over several line-kilometers both over dry snow and wet snow zones.

1-3 Summary of original research contributions

In this work, we implemented a fusion of signal processing algorithms for pre-processing and deep learning supervised and self-supervised algorithms to deal with the imperfect and limited-annotated-data problem to achieve accurate tracking of isochronous radiostratigraphic layers in echograms.

Specifically, in this work, we achieved the following research milestones:

1. Successfully designed and tested different streams of deep learning algorithms to understand their strengths and weaknesses.

2. Successfully deployed the first generalizable deep learning algorithms to track persistent snow layers over thousands of line-km over Greenland Ice Sheet.

3. Improved automated tracking (requiring minor quality control) to speed up layer tracking by x5.

4. Creation of a large “deep learning ready” Snow Radar dataset.

5. Synchronization of the Modele Atmospherique Regionale (MAR) regional climate model with radar echograms.

Chapter 2: LITERATURE REVIEW AND DEEP LEARNING BACKGROUND

2-1 Literature review

In over three decades of collecting remote radar measurements over polar ice sheets, a large amount of data has been collected necessitating the need for efficient processing and tracking of snow and ice layers in the echograms. As a result, many scholarly works exploiting different characteristics of the snow surface and subsurface targets to track the snow layers have been developed. These approaches can be broadly grouped as: Semi-automated vs fully automated, surface-bedrock vs internal layer tracking, and machine learning (ML)/deep learning (DL) vs non-ML algorithms.

Semi-automated methods require some form of human input for tracking while fully automated algorithms are designed to operate without any human interaction end-to-end. Surface-bedrock models aim to accurately track only surface (air-snow boundary) and bottom (ice-bedrock boundary) layers while internal layer trackers are designed to track all laterally persistent layers that could appear at any depth within the snow and ice. Surface-bedrock tracking is usually intended for lower frequency radar systems that can penetrate all the way through to the ice bottom but have coarser resolution to distinguish internal layers. Internal layer tracking is needed for most radars, but only sufficiently fine range resolution radar systems can visually distinguish annual layers from one another. The ability to clearly discriminate between each internal layer is dependent on the bandwidth and center frequency of the radar system. Finally, non-ML models deploy methods such as statistical models, level-set, probabilistic graphical methods, etc., while machine or deep learning methods develop artificial neural networks to track the desired layers.

Gifford et al. [37] combined active contour and thresholding edge-based approaches to identify contour boundaries (surface and bedrock) after applying gradient-based edge detection techniques and image processing to reduce noise effects. The consecutive studies in Ferro et al. [38], [39] developed statistical models for characterizing subsurface backscatter categorized into strong layers, weak layers, low returns, and basal returns which were modified and applied to automatically estimate ice thickness of data acquired from Antarctica in [39]. Koenig et al. [40] also developed a high-frequency versus low-frequency semi-automated discriminator algorithm to identify peaks in the returned backscatter power. Rahnemoonfar et al. [41] applied a distance-regularized level-set function to detect the surface and the ice bottom to improve earlier work by Mitchell et al. [42] and D. P. Onana et al.[43] . In another work, Rahnemoonfar et al. [44] took inspiration from Coulomb's law of electrostatic force to detect ice surface and bottom boundaries after performing anisotropic diffusion to enhance layer edges.

Another set of works approaches the problem as a probabilistic inference problem by developing probabilistic graphical models to detect layer boundaries. Crandall et al [45] pioneered the paradigm while Lee et al. [46] employed Markov-Chain Monte Carlo (MCMC) over the joint distribution of all possible layer targets in the inference problem. Xu et al. [47] expanded the scope to include 3D surface and bedrock reconstruction using Markov random fields (MRF) and Berger et al. [48] refined the approach by incorporating additional domain knowledge in the unary and binary loss function terms.

Carrer and Bruzzone [49] introduced machine learning methods by incorporating a support vector machine (SVM) with statistical modeling to classify layers, bedrock, and noise.
Since then, several machine and deep learning models have been designed and applied to the radar echogram layer tracking problem. Kamangir et al. [50] introduced a hybrid wavelet

network for ice boundary detection. Xu et al. [51] designed a multi-task network that avoids the use of meta-data using a combination of recurrent neural networks (RNN) and 3D ConvNets to track layers in 3D radar imagery. In Varshney et al. [52] internal layer tracking was introduced as a semantic segmentation task and deep neural networks were applied to the problem whereas Cai et al. [53] applied an Atrous Spatial Pyramid Pooling (ASPP) module using a ResNet-101 backbone to detect layer and bedrock interfaces in echograms created from Antarctica campaigns. Other efforts by Yari et al. [54] and Wang et al. [55] applied multi-scale transfer learning and tiered-segmentation approaches for internal layer tracking respectively while Rahnemoonfar et al. [56] introduced the addition of synthetic data for model training and multi-scale learning for tracking ice layers in [57]. In Yari et al. [58], physics-driven and GAN methods were used to create simulated data to train a multi-scale network to improve the accuracy of internal layer tracking. Varshney et al. [52] combined the layer tracking task and ice thickness estimation using fully convolutional neural networks and extended the approach in Varshney et al. [59] by incorporating physics-defined labels. More recently, Ghosh and Bovolo [60] combined attention modules with the fusion of TransU-Net and TransFuse modules to segment the combination of ice layers, bedrock, and noise similar to efforts in Cai et al. [61] with the addition of a Multiscale convolution module (MCM) and focal loss for class weight balancing.

2-2 Snow Radar and Snow Radar echograms

The Center for Remote Sensing and Integrated Systems (CReSIS) at the University of Kansas has developed a suite of radar systems suitable for collecting remotely sensed measurements over polar regions in the last few decades [10,13,59, 60]. Depending on the subsurface target or scene of interest and considering the limitations of electromagnetic propagation through that scene, the frequency of operation, transmit power, and bandwidth are chosen to design the appropriate radar system. The primary interest of this work is to focus on radar systems designed

to image the shallow snow and not-too-deep (firn) internal layers in the ice sheet column. However, the overall goal of this research is to identify best practices and develop deep learning algorithms not suited for just a specific radar system but generalizable to multiple radars systems especially if these systems share similar hardware design and image relatively the same section of the ice column.

We begin our efforts by focusing first on the Snow Radar data to train and test the early iterations of the deep neural networks. The Snow Radar is a Frequency Modulated Continuous Wave (FMCW) system that, for the dataset used in this work, operates in the 2-8 GHz frequency band and is designed to image the top firn layers of the polar ice sheets with a vertical resolution of ~4 cm in snow. The echogram images from the Snow Radar data are formed through pre-summing, pulse compression with windowing, coherent noise removal, deconvolution, and incoherent averaging. The Snow Radar is capable of measuring snow thickness over sea and land ice by imaging shallow snow layers. Over land, it can identify annual snow layer interfaces due to its fine vertical resolution which is important for estimating annual accumulation. Over sea ice, it can be used to estimate snow cover on sea ice. This work focuses on the land ice problem of tracking annual layers.

The annual snow layer interfaces (hereafter '*internal layers*') appear as peaks in the pulse compressed range profiles because, at each interface, a portion of the transmitted electromagnetic field is scattered back to the receiving antenna. The interface between late summer and early winter snow fall appears characteristically different to the radar receiver resulting in annual layering within the snow imagery. This difference can be attributed to the difference in the seasonal weather patterns, creating a snow permittivity change that causes detectable backscatter towards the receiving antenna. The distinct peaks in the pulse-compressed backscatter are further

enhanced with various digital filtering and signal processing methods to create the radar echogram.

2-2-1 Brief overview of the Snow Radar system configuration for the 2012 science mission

The first iteration of the deep learning models developed was trained solely on the data collected during the 2012 Operation Ice Bridge Mission. The Snow Radar configuration for this campaign was for the Lockheed P-3 Orion aircraft flying at a nominal altitude of 500 m above ground level. A detailed description of the hardware configuration for this campaign is documented in [15], [16]. Table *1* summarizes the key hardware configuration parameters for the campaign. Datasets from later campaigns are tested in the later stages of this work. The hardware configuration was progressively improved for these later campaigns, but the images are generally similar in nature to the 2012 campaign. Details of some of these improvements can be found in [62], [63].

Table 1: *The Snow Radar parameters used for the 2012 science mission.*

Parameter	**Value**
Transmit power	100 mW
Pulse duration	250 μs
Bandwidth	2-8 GHz
Intermediate frequency ADC sampling rate	125 MSPS
Range resolution	~ 4 cm

Here, we provide a basic overview of the digital signal processing routines performed to aid the understanding of how the echogram images are formed. The digital signal processing starts by loading a two-dimensional data matrix where the row dimension is fast-time and the column dimension is slow-time. This is converted from the receiver's ADC quantized values to the received voltage value at the antenna. This data can be referred to as ***space-time data***. Applying the Fast Fourier Transform (FFT) along the row or fast-time dimension, with Hanning windowing to reduce sidelobes, converts the fast-time axis to the frequency domain which, for FMCW radars, is proportional to the two-way travel time or delay to the target. The delay can be converted to depth (or range) which will ultimately be used to infer the layer thickness between adjacent snow layers. After applying the FFT to pulse compress the data, phase and time corrections are performed to compensate for the effects of altitude variation due to aircraft position and altitude changes. Subsequently, further processing such as coherent noise removal is performed by estimating the noise using a low pass filter in the space axis with a very low cutoff frequency (30 seconds of flight time) and subtracting it from the space-time data. For the survey data used to create the training and test set for this work, a 1x5 depth-by-along-track boxcar filter is applied to the power-detected data and then decimated in the along-track by a factor of 5. This reduces the variance of the speckle noise and accentuates the internal layers since, typically, they are roughly horizontal and aligned with the along-track dimension.

2-2-2 The Snow Radar echogram

The radar echogram introduced earlier in Figure 1-2 can be summarized as a representation of the data matrix produced by taking the logarithm of the power-detected, coherently, and incoherently averaged received backscatter returns. The Snow Radar has limited cross-track elevation angle resolution due to the small antenna used, but the scattering is assumed to come from the nadir elevation angle. The echogram image shows snow accumulation patterns beneath

the ice surface along the flight profile that can be resolved into roughly annual layers. The horizontal axis represents the along-track dimension (direction of aircraft flight) where each column is a “range-line” while the vertical axis is the fast-time dimension, and each row is a “range bin”. The pixel intensity is a function of the received radar scattering from the resolution cell with lighter pixels representing low returns and darker pixels depicting stronger returns from the buried snow layers.

Figure 2-1 shows the same Snow Radar echogram image created from data collected at the ice sheet summit in Greenland. The air-snow interface where the transmitted signal first interacts with the snow layers is referred to as the *“surface”*. Other layer stratigraphy corresponding to the annual snow fall are collectively referred to as *“internal layers”*.

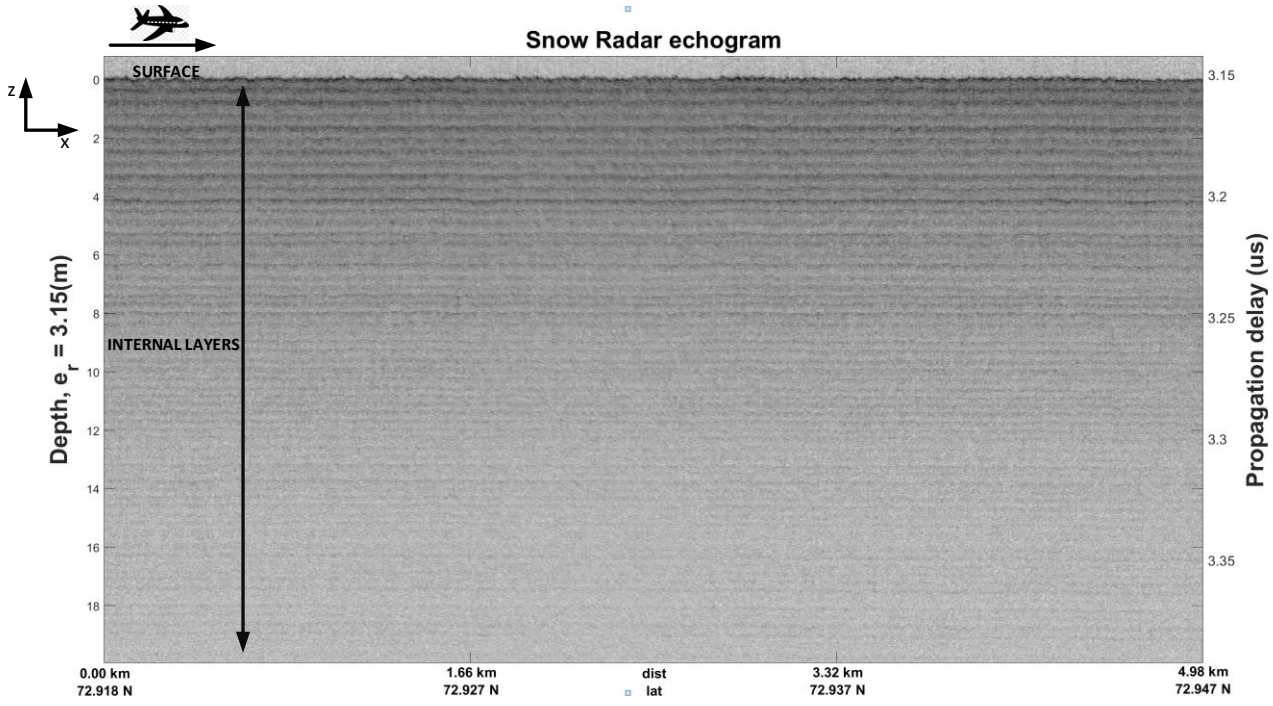


Figure 2-1: *Snow Radar echogram showing several decades of snow accumulation*

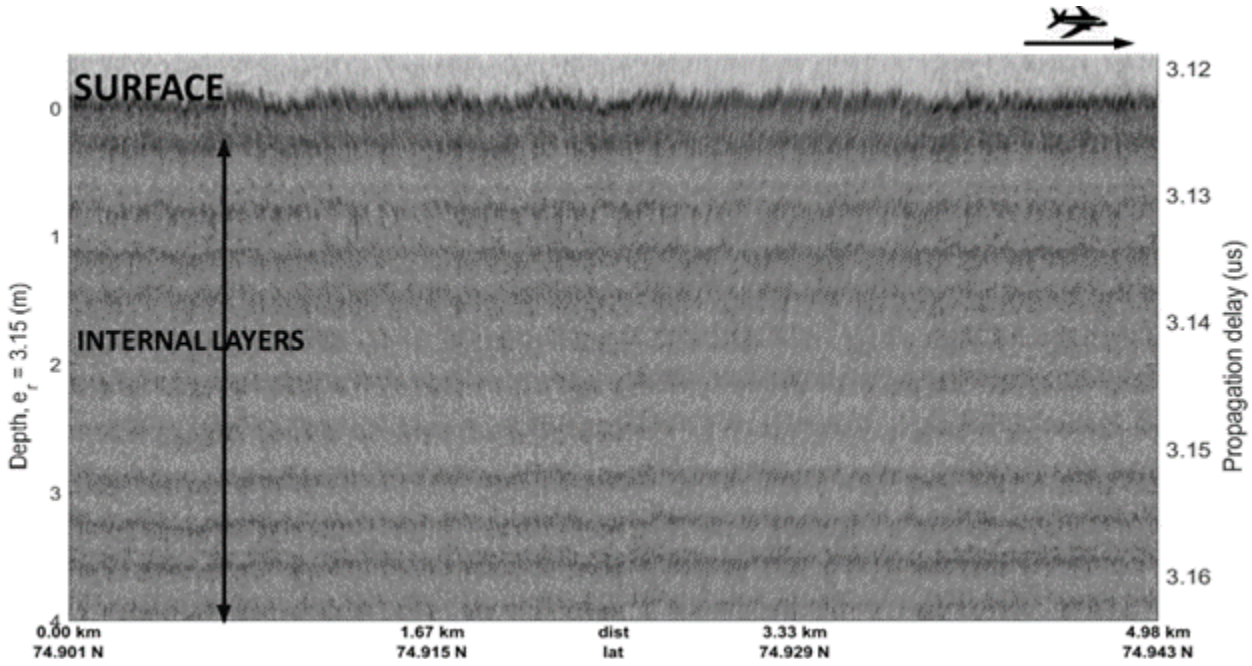


Figure 2-2: *Magnified Snow Radar echogram showing only the first few layers*

The Snow Radar echogram has unique features that distinguish it from echograms created from other radar systems. This is particularly because of the system's S-C frequency band which detects the top firn layers and the associated large bandwidth allowing it to achieve fine vertical resolution in snow. This makes each snow layer clearly distinguishable in the resulting image unlike other contemporary radar systems with coarser resolution that are unable to separate annual layers [64]. The layers can be seen in Figure 2-1 as the "dark" roughly horizontal and laterally persistent pixels along the flight path [9], [34]. The horizontal (x-axis) of the echogram image corresponds to geolocations (latitude/longitude) traveled along the flight path of the aircraft while the vertical (y-axis) represents the radar two-way travel time to the different snow stratigraphy and the depth at which they occur.

Echogram images help visualize subsurface features, particularly snow layers, in the mapped locations. The layer's contour (flat or curved) reveals significant information about the snow's historical deposition and subsequent metamorphosis. The curvature of these layers indicates a

variety of geological and climate processes, such as the flow dynamics of the glaciers, the accumulation patterns of snowfall and the effects of wind redistribution. The echogram images capture the orientation of the layers allowing for detailed analysis of the snowpack structure and the accumulation patterns [65], [66], [67].

The orientation and degree of curvature of the snow layers are influenced by several factors such as the terrain of the mapped location. For instance, in areas where snow accumulates on sloped or uneven terrain, the layers may display significant curvature due to gravitational settling and compaction. Particularly in glacial regions like Greenland, the movement and deformation of the ice can create complex, undulating patterns in the snow layers. These curved layers can also result from differential snow deposition, where wind and topography interact to create variable snow depths. By analyzing these curved contours, inference about past climatic conditions and accumulation rate can be made and this helps make better predictions about future snow behavior.

As described in Section 1-2, annual accumulation can be estimated from the radar two-way travel time (hereafter "*twtt*") and converted to meters using available snow permittivity-depth profiles for that location. However, to achieve this, all the layers in the echogram must be detected and accurately tracked. As such, the goal of echogram layer tracking is to obtain a 2D matrix G of dimensions $N_L \times N_x$ where N_L is the number of snow layers in an echogram image (often unknown apriori by the tracking algorithm) and N_x is the number of rangelines in the echogram. Each row of G contains a vector of the radar two-way travel time of each of the N_L layers sorted from the topmost layer till the last detected annual layer. This matrix G of tracked layer twtt can be plotted over the echogram image to visualize the tracking result. Figure 2-3

shows an example of an echogram where the snow layers have been identified, tracked and plotted in red over the echogram image. This is often used in tracking as a visual sanity check to see how well the result of the tracking algorithm is following the layer peaks/boundaries. These tracked twtt are subsequently used to estimate the annual accumulation for the mapped location.

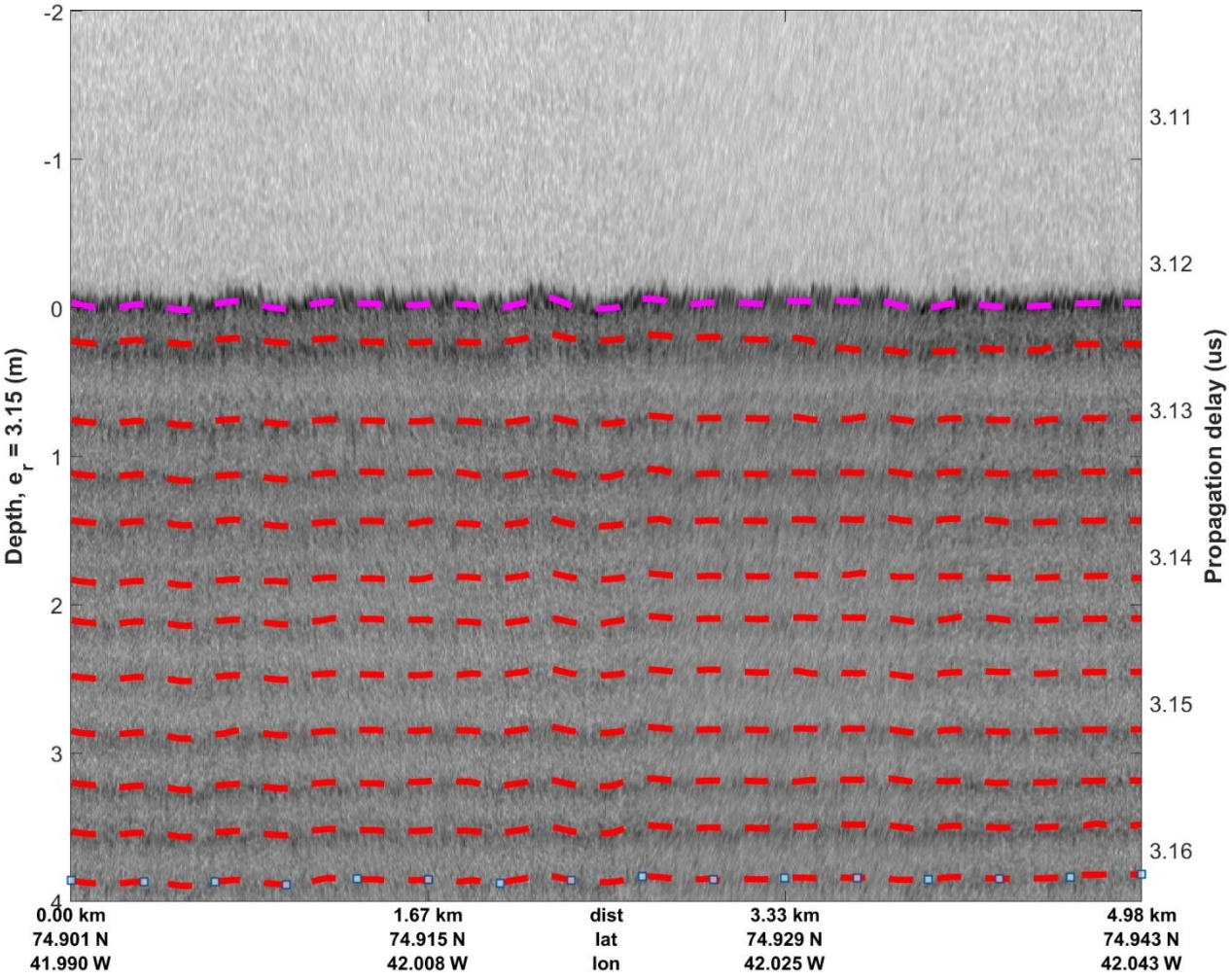


Figure 2-3: *Magnified echogram image with tracked layers plotted in red.*

2-2-3 Some challenges inherent in echogram layer tracking

Automatic layer tracking algorithms have some inherent challenges that need to be addressed to achieve good tracking results while being able to generalize to a broad range of echogram images. The primary input for most tracking algorithms is the echogram image (many times as the only input), hence, issues with the quality of the echogram directly impact the performance of the algorithms. The image quality of an echogram is usually determined by the snow zone it was created from.

The echogram images shown so far are from the dry snow zone whose echograms have the best image quality because of the high layer reflectivity relative to the diffuse volumetric background scattering and the relatively flat, parallel and distinguishable accumulation layers. The dry snow zone is the coldest and highest region of the ice sheet, farthest from the coast, and is usually characterized by extremely low temperatures all year round with little or no melt. The large density contrasts in this zone reflect a strong radar signal back to the aircraft. Other zones such as the percolation, ablation and wet snow zones have echogram images with less trackable layering.

A primary challenge in echogram layer tracking is the significant variability of annual snow layer morphology within echograms. There is a wide variety of shapes and forms the annual snow layering can take in an echogram image. This is usually dictated by the surface topography and geographical features of the imaged location. The rapidly varying spatiotemporal accumulation pattern across the ice sheets means echogram images created from two locations a few hundred meters apart may appear starkly different. Further, the number of layers in each and the morphology of the layers can also differ.

The earlier illustrations in Section 1-1 employed simplified echogram imagery to facilitate easy comprehension of the radar echogram layer tracking process. However, it is important to know that a lot of echogram images exhibit greater complexity. Some examples of echograms with different layer morphology and orientation are shown in Figure 2-4 and Figure 2-5.

Also, the echogram images shown so far have undergone an important "processing" step to remove the effect of the surface relief of the surveyed location. The process of compensating for the surface topography is referred to in this work as "surface flattening" where each range line is adjusted so that the snow surface lies on the same row in the echogram matrix for every column. To perform surface flattening, the 1D surface layer contour (i.e. the index of the pixel containing the surface in all the echogram rangelines) is used. Prior to surface flattening, echogram images reflect the radar trajectory elevation changes and the surface relief of the location, both of which may vary widely across the dataset.

Figure 2-4 and Figure 2-5 show example echograms prior to the surface flattening step. This gives a glimpse into the wide variety of layer orientation, layer interspacing, and the number of layers that can exist in an echogram image.

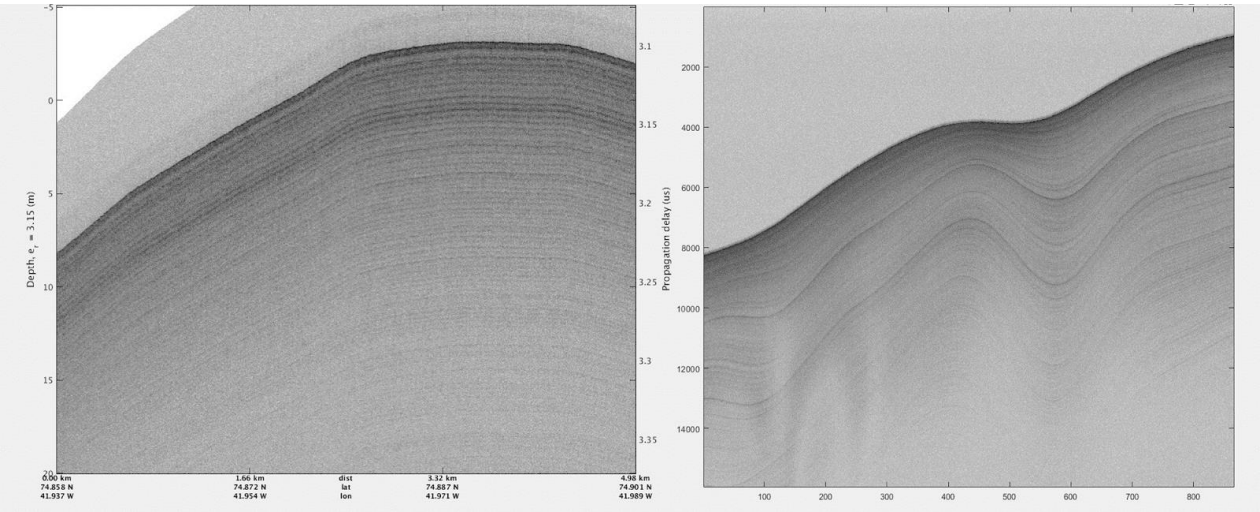


Figure 2-4 *Sample echograms with unflattened surface and varying accumulation patterns*

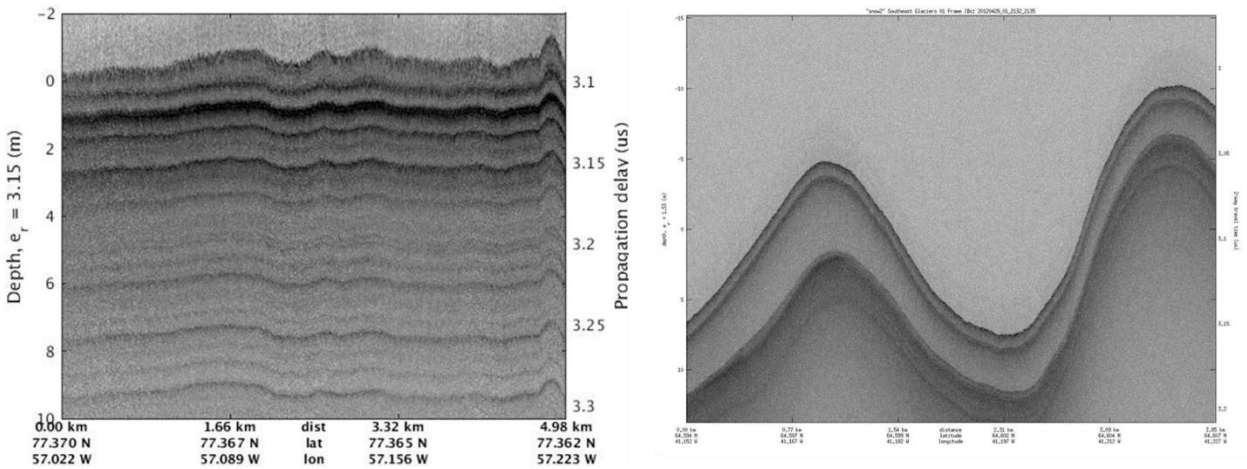


Figure 2-5 *Additional example echograms with unflattened surface and varying accumulation patterns*

Furthermore, it should be noted that even after surface flattening, the internal layers are not always “flat and parallel” to each other. There also exists a range of variability in the geometry of accumulation layers within a unit echogram image. As a result, automatic echogram tracking algorithms need to be robust to handle the diversity in echogram layering.

The geometry of the internal layers, although generally following the structure of the topography, can be unpredictable and is generally unknown a priori. Figure 2-6 shows another echogram where the orientation of the first few layers differs sharply from the deeper layers. The deeper layers are often more closely packed than shallower layers because they have been "burdened" by newer depositions. The weight of the earlier layers compresses the older accumulation to lose air content resulting in smaller density variations and spacing which impacts the received radar reflectivity from them.

The deeper layers in the echogram image also have lower signal-to-noise ratios (SNR) due to signal extinction through scattering and attenuation as the radar wave transverses down and up through the snowpack. This is particularly worse for echograms created from snow regions other than the dry snow zone such as the ablation or wet snow zone where the presence of meltwater diffusely scatters the transmitted signal in all directions lowering the quality of the generated echogram for layer tracking. As a result, some rangelines may have significant fading for some or all their range bins. This has significant consequences on layer tracking as the tracking of faded layers is harder and sometimes ambiguous even for trained experts leading to subjective and different tracking results.

The synergistic interaction of these effects presents a formidable challenge for many automatic tracking algorithms. Consequently, a robust algorithm is necessary to achieve consistent layer tracking across the vast polar ice sheets.

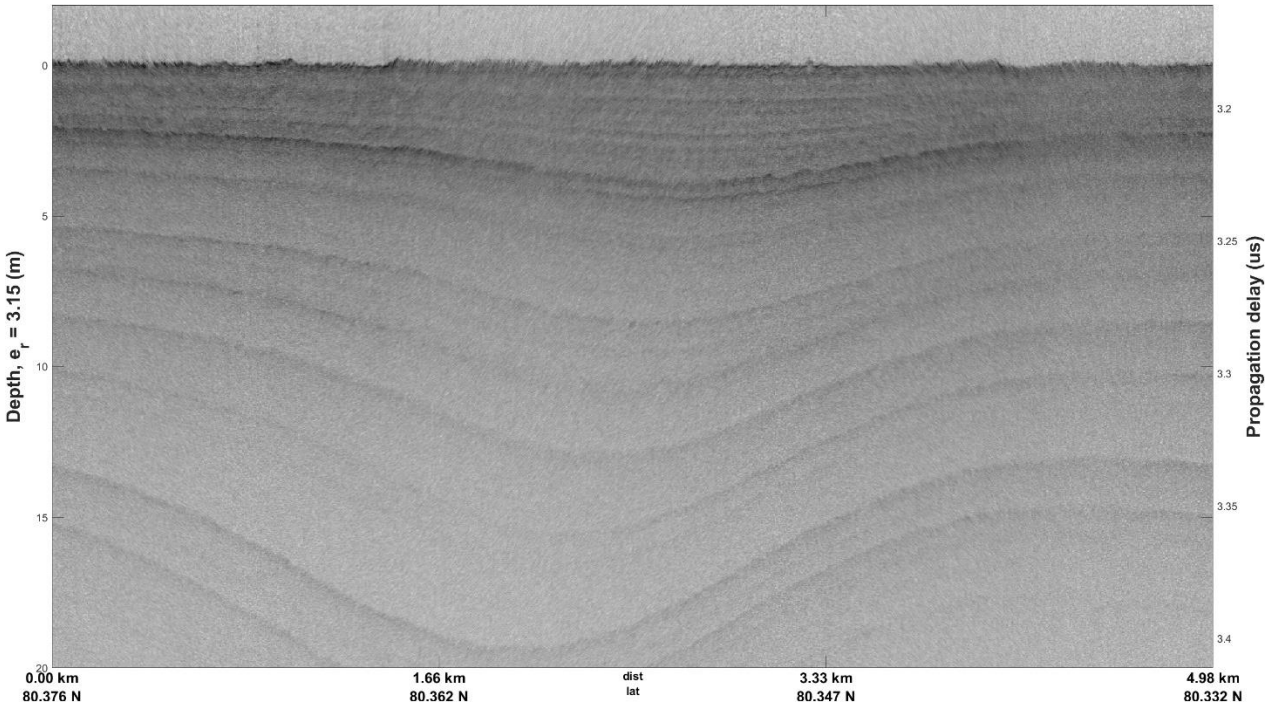


Figure 2-6 *Echogram with different internal layer orientation after surface flattening.*

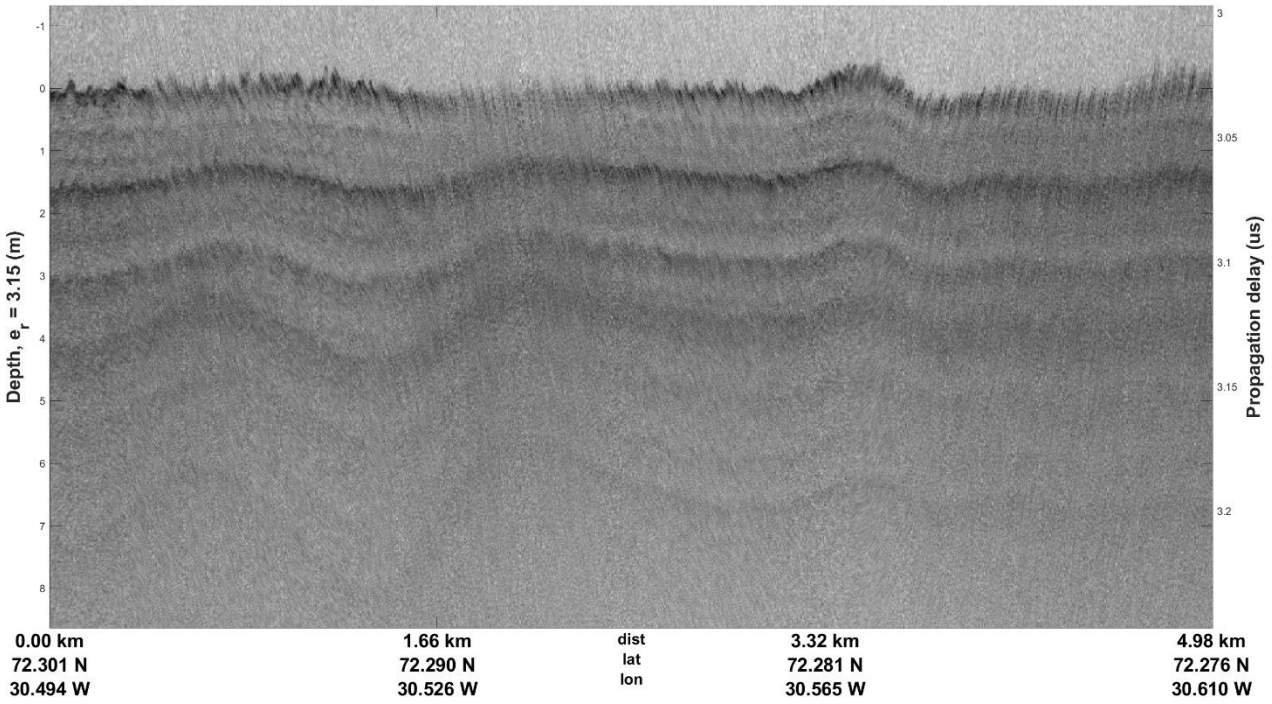


Figure 2-7 *Echogram with different internal layer orientation after surface flattening.*

2-3 Rationale for machine learning adoption in echogram layer tracking

Artificial intelligence, specifically machine learning and deep learning algorithms, hold great potential for this problem given their well-reported performance in other scientific domain with high dimensional and non-linear data such as in the speech recognition problem and man-machine communication. Deep learning algorithms are currently the state-of-the-art algorithms on a variety of hitherto complex scientific fields datasets ranging from astronomy [68], [69], material science [70], [71] to human genomics and bioinformatics [72], [73]. In the optical image domain, which shares similarities with echogram images, deep learning has become the standard for performing classification, object detection and localization, image analysis and generation and semantic segmentation [21], [22], [23], [24], [25], [26], [27], [28], [29], [30].

As evidenced in the echogram layer tracking literature, several traditional signal processing algorithms and statistical models have been applied in the past. Despite the development of many such traditional algorithms, their efficacy remains constrained, particularly on echogram sets with poor image quality and “non-flat” accumulation patterns. Consequently, they are best adopted as semi-automated tools that can be used by humans during the manual tracking process to speed up the tracking. However, semi-automated tracking methods still require significant human interaction for layer tracking which impedes the speed of layer tracking and does not scale well to the large volume of data collected.

A handful of fully automated tracking methods only perform well for a limited number of echograms but need to be redesigned for the dynamic accumulation pattern captured in a large dataset. This is likely due to the limited amount of both modeling data and the expressive power of the assumed model in these approaches. Many of the designed models are too simplistic and fail to fully capture the underlying layer accumulation process and inadequately capture the

nuanced details and complex interactions that characterize the process. This, in part, explains why they achieve limited success only on some echograms.

Furthermore, the analysis of airborne radar data shows that the signal plus noise distribution appears to deviate from "well-described" probability distributions. The non-stationary nature of layer accumulation data, coupled with rapid decorrelation of the accumulation pattern across space, presents a significant challenge. Consequently, constructing signal processing models that fully capture the underlying data distribution, thus facilitating generalization to varied polar regions with distinct spatiotemporal snow accumulation patterns, proves arduous.

Deep learning algorithms, however, hold the potential of generalizing easily across different datasets and variations of the accumulation pattern captured in the echogram when correctly trained [31], [32]. Machine learning (ML) and deep learning (DL) algorithms offer distinct advantages over classical methods when addressing the radar echogram layer tracking problem. Firstly, ML algorithms excel at detecting and modeling intricate patterns within data. This is particularly useful in the echogram layer tracking problem where there exists a highly non-linear relationship between the power-detected echogram pixel values input, and the output snow layer coordinates in the echogram image. The highly parameterized deep learning architecture and number of trainable parameters increases the degrees of freedom of the model leading to robust modeling of the input-output shape. Also, the intermediate insertion of various non-linear activation functions such as the rectified linear unit (ReLU) and the Gaussian Error linear unit (GeLU) further enhances the ability to model non-linearity in the input-output mapping.

Furthermore, these models are adaptable to a variety of input modalities and frequencies such that during inference on input echogram images from slightly different radar systems or

accumulation patterns, the robustness of the trained model is evidenced in good one-shot performance. This reduces the need to develop new models for slight variations in echogram images or accumulation patterns. Even in the case of significant domain difference, the model can be adapted by transfer learning and finetuning on a relatively small new dataset to optimize the weights of the trained model to the differences in the new echogram set.

The potential for end-to-end learning of these algorithms to capture the input-output relationship, without the need for manual feature engineering on the input is very attractive. While the current models developed perform post-processing to extract the visibly identified layers from the output activation map, no extra feature engineering is done on the input. This is critical since all the information in the input echogram is available to the deep learning model during training. Hence, the model can learn subtle relationships that are valuable to achieving good performance on the echogram layer tracking problem but may be unknown to humans. Although the interpretability of these models is currently limited which makes troubleshooting and parameter investigation somewhat difficult, they overwhelmingly make up for their opaqueness by achieving sterling results on a variety of problems.

On the echogram layer tracking problem, the models trained in this work achieve good performance, as detailed in the evaluations in 3-2-4-2, 4-6, 4-6-2, 5-2-6 and 6-2-7-2 in laterally tracking of consistent accumulation layers over several kilometers particularly in the dry snow zones of polar ice sheets.

2-3-1 Fusion of signal processing and machine learning for the layer tracking problem

Machine learning and deep learning algorithms are argued to be extensions of concepts and ideas that originated from adaptive signal processing, statistical modelling, and estimation theory.

However, several modifications, such as improved optimization algorithms, introduction of non-linear activation functions, very large model parameters and complex model architectures in deep learning have helped it achieve better performance on complex problems with large datasets. The significant overlap between both fields suggests that a fusion of ideas from both will lead to better results on the layer tracking problem.

In this work, we combine approaches from both fields by applying signal processing techniques like pulse compression, constant false alarm rate detection of the surface layer, coherent and incoherent integration, and signal detrending in the pre-processing stages to improve the quality of the echogram images. Subsequently, machine learning and deep learning algorithms were developed to identify the layer pixels and track them in the improved images. Finally, a combination of signal processing algorithms and computer vision algorithms were developed to extract the identified layers in the deep learning model outputs.

2-3-2 Challenges in applying deep learning to echogram layer tracking

Despite the appeal of applying machine learning and deep learning algorithms to the layer tracking problem, there are some inherent challenges that need to be addressed. First is identifying the appropriate deep learning paradigm that best suits the problem. The deep learning paradigm in this context refers to one of supervised, unsupervised or deep reinforcement learning. The initial deep learning framework chosen for formulating the layer tracking problem was to set it as "supervised deep learning" because of the flexibility of the approach and reported success on similar problems in other fields. However, supervised deep learning algorithms suffer from the circular problem of first requiring large amounts of labeled data to train the models [74]. These large quality annotations are required to expose the model during the training phase to the true underlying statistical distribution of the data to find the optimal layer weights and

parameters in the solution space. However, at the inception of this research, there is no such fully annotated large radar echogram dataset. This prompts the need to progressively create a radar echogram dataset large enough to train a deep learning model. The alternative of direct zero-shot learning from weights of large models trained on optical images fails largely due to the dissimilarity between the data domains. This precludes the application of off-the-shelf deep learning algorithms which fail on the echogram layer tracking problem largely because of the lack of overlap in the training data and radar echograms.

To train a supervised deep learning model for layer tracking, a key to achieving good performance is the availability of high-quality input echogram images. This will facilitate the ability of the models to understand the noise peculiarities inherent in remotely sensed radar data to correctly discriminate signal and noise signals. However, signal attenuation and fading through the snow medium, non-ideal radar system characteristics, aircraft roll and antenna radiation pattern effects, off-nadir backscatter and multipath scattering are some of the phenomena that combine to form the non-Gaussian noise distribution that result in the observed imperfections in the echogram image. This invariably results in poor input image quality when the effects are severe. These noise effects are particularly noticeable in images created from radar data collected from wet or ablation snow zones where continuous melting introduces melt run-off water that both disrupts the annual snow layering and attenuates the radar signal. For these kinds of images, the presence of along-track fading particularly when the orientation of the accumulation layers in the image is curved or arcuate, results in a challenging layer tracking problem. Hence, the need for a robust custom deep learning algorithm.

Also, existing popular deep learning models are mostly designed for relatively small optical input image sizes or images that can be resized with no harm to performance. For example,

resizing a high quality 1280 x 720 pixel image of a dog to 256 x 256 does not affect a classification model trained to identify the animal in the picture. However, this does not hold true for echogram images. The high resolution of echogram images introduces additional complexities in making the decision for the right training setup such as the choice of deep learning architecture, the computing and memory resource requirements, training time and inference time. Furthermore, the option of resizing requires additional care since arbitrary resizing of the echograms to smaller dimensions can distort the spatiotemporal information, particularly in the depth or fast-time axis information, making it impossible to tie tracked layers back to the physical problem of estimating annual snow accumulation. This prompts the need to either train on the large image size or find creative ways to decimate whilst preserving the output layer resolution.

Additionally, a critical fact inherent in airborne radar subsurface mapping is that snow layer interfaces in echogram images are more than one pixel thick. However, the labeling process used to create the ground truth assigned just one pixel to each column of each layer in the echogram. This has implications for training and evaluating deep learning models for layer tracking. First, the tracking of echogram layers even by trained humans to create the ground truth label is ambiguous since there are multiple layer pixels that can be chosen. Although there exist general unwritten rules and guidelines regarding the selection of consistent layer pixels as ground truth, a universally accepted scientific method remains elusive. Secondly, this ambiguity is significant in model performance evaluation and the definition of evaluation metrics that truly reflect how well the models are tracking the layers. A model might be reported to do poorly based on certain metrics that compare with the human-annotated label even though the model chooses alternative layer pixels that can be said to be equally correct.

More so, the involvement of humans in creating the ground truth labels inevitably introduces errors. Manual labelling of echogram layers is in fact very difficult, and errors are often unavoidable due to human fatigue. This is of consequence in the design of the deep learning architecture because some architectures are more susceptible to these errors than others. These label ground-truth errors become more critical when evaluating the accuracy of automatically tracked layers by different deep learning models.

In summary, the challenges to be considered when deciding the deep learning algorithm to use for the radar echogram layer tracking problem include:

1. Existing ground truth annotations are imperfect, limited, and incomplete for each echogram. Moreover, subtle ambiguity sometimes exists in the definition of a layer's exact pixel location since the layers are generally more than one pixel thick.
2. The fine resolution of the echogram image demands a higher GPU memory budget which competes with the optimization of heavily parameterized algorithms. Although it is possible to resize the images, care is needed to not distort the layer information. Furthermore, when models are trained on decimated images, inference on real full-scale echograms would require extra manipulation that may introduce subtle errors (e.g. odd decimation factors).
3. The dimension of the echogram images in the dataset may vary. Even more critical is that the number of layers that may exist in the image is unknown a priori. Most deep learning algorithms prefer fixed input image sizes with a fixed number of output classes.
4. The output of the models needs to be a layer contour.

Chapter 3: METHODOLOGY

At the inception of this research, the amount of echogram image and annotation pairs available was very limited. Recognizing the strength of deep learning algorithms and their weakness of requiring large amounts of annotation, a heuristic approach is first adopted to decompose the radar echogram layer tracking problem into a simpler iterative problem. This chapter discusses the iterative row block algorithm, the deep learning algorithms designed for this and the post-processing step to reconstruct the decomposed echogram images.

3-1 Iterative RowBlock algorithm

In this paradigm, the dense prediction problem requiring each pixel in the echogram image to be individually classified into different layers all in one go is reformulated as an iterative layer detection problem where the goal is to identify a single layer at a time starting with the surface whose position is known a priori. Concretely, instead of attempting to track all the layers in the echogram image all at once by treating the image as a single input to the algorithm, the echogram image is first broken down into smaller units and layer tracking is done on this one layer at a time. This is achieved by employing the echogram image matrix decomposition routine described below.

Consider the 2D matrix in Figure 3-1 depicting an echogram image. The surface layer (coded in red) is always known a priori since simple thresholding algorithms can be used to track the surface. It is expected that the next layer (the first internal layer in this case) will be a few rows beneath the surface and that the subsequent layer (the second internal layer) will also be a few rows after the first internal layer and so on. This assumption is because each layer is produced from the net surface mass balance from the annual cycle of precipitation and evaporation, which

is assumed to be positive each year in the dry snow zone. Therefore, the internal layers do not cross each other even when there is some ice melting (i.e., the layer thickness is always positive). Hence, to track the internal layers starting from the first internal layer, a fixed number of rows starting from those immediately after the surface are extracted. We term these extracted rows/pixels as a ***RowBlock*** because it is roughly parallel with the rows. Specifically, a RowBlock is the region of the echogram image directly beneath a previously detected layer which would contain the next layer and the algorithm is trained to determine the index of the next layer from these. Usually, the initial layer that defines the first or top row block is the snow surface. It is important to note that the RowBlock's top and bottom edges do not follow constant rows in the original image, since the top and bottom edges follow the layer above that defines the RowBlock.

Thus, we pose the tracking problem as an iterative detection problem which is solved by tracking one layer at a time from the iteratively formed RowBlocks. An example of forming the first two row blocks of an image is shown in Figure 3-2. The number of pixels rows that are in a RowBlock, N_{rb}, is a hyperparameter that is chosen based on statistics of internal layer spacing collected from available manually tracked internal layers. It is chosen to be large enough so that the next layer occurs within the row block. Sometimes, a row block may contain more than one layer i.e., a deeper or subsequent layer after the next layer could be included in the row block. The purpose of this is so that the neural network models can be trained to learn how to ignore these deeper layers and only track the next layer.

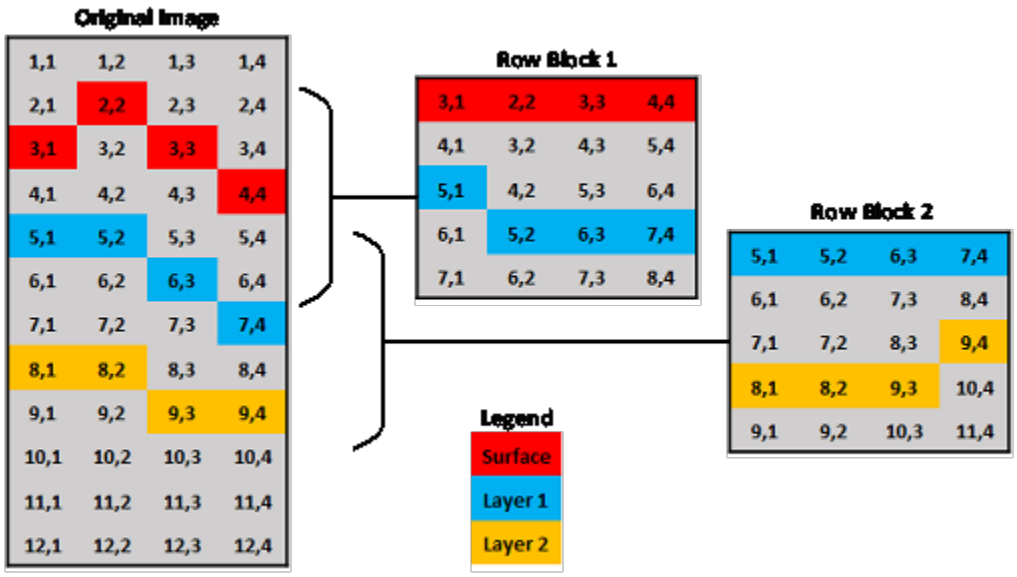


Figure 3-1 Depiction of an echogram image showing iterative RowBlock creation

The example in Figure 3-1 only shows RowBlocks with a single layer in each; if the number of rows N_{rb} in each row block had been increased, then part of layer 2 would have shown up in row block 1. A possible reason for restricting the number of rows in each row block (e.g., to N_{rb} = 5 rows in Figure 3-1) is to reduce the size of the Neural Network (NN) and therefore the learning time of the NN. However, this must be balanced with the need for N_{rb} to be large enough to always ensure that the next layer will be completely contained in the row block.

Rather than passing a single column in a row block as the neural network's input, we form a "ColumnPatch" - a combination of the current column and the adjacent N_{cols} columns to the left and right. Using ColumnPatches as inputs to the model has the advantage of including local spatial information that is shared by adjacent columns which improves the model's performance and ensures a smooth column-to-column transition in the tracked layers. For columns with insufficient support at the edge of the row blocks, available columns are mirrored to complete the

required ColumnPatch size. Figure 3-2 illustrates how ColumnPatches are formed from the row blocks in Figure 3-1

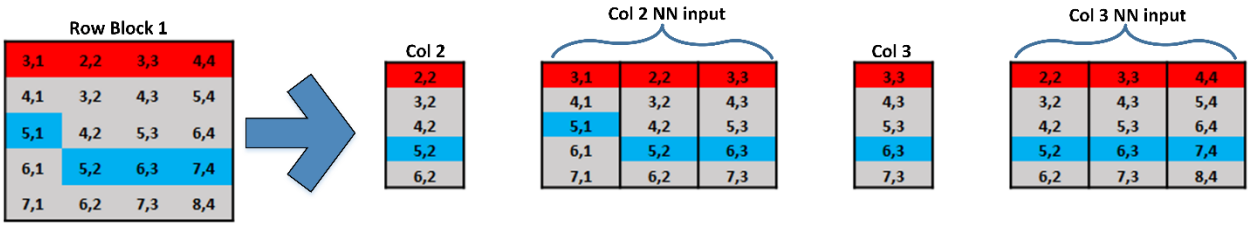


Figure 3-2 ColumnPatch creation process

In this paradigm, the objective of the deep learning model is to identify the index of the next layer in each column of the current row block, one column at a time. Concretely, to track the first internal layers, N_{rb} rows after the first layer are used to form the row block. For each column, the input to the neural network is a ColumnPatch formed with $2N_{cols} + 1$ columns centered on the column for which the next layer is being estimated. The algorithm is trained with the ColumnPatches as inputs and the outputs as the index of the pixel containing the next layer. The tracked location of the first internal layer from all the columns in the row block is then used as the base to form the next row block from which the next internal layer is to be tracked. This process is iterated until a termination condition is met (and ideally all the detectable layers in the echogram are tracked).

The advantage of this algorithm is reconstructing a complex problem of tracking an unknown number of layers in a full echogram into a simpler problem of finding just the next layer in a focused area of the echogram where the next layer is expected to be. Using RowBlocks to create ColumnPatches, the complexity seen by the algorithm is reduced from having to identify and track layer pixels in the entire echogram to only identifying the index of the next layer for a

single column in a ColumnPatch. A direct consequence of this decomposition is substantial expansion of training data by several orders of magnitude. The exact order of expansion depends on the choice of N_{cols} and N_{rb}. By breaking the echogram into RowBlocks and then into ColumnPatches, the unit input to the deep learning algorithm is no longer the entire echogram image but sections of it. This helps circumvent the challenge posed by limited availability of training data by providing millions of training examples.

It is important to note the difference in the algorithm's routine during training and inference. For training, the tracked internal layers (ground truth) are always available, therefore the RowBlock for a layer is formed using the available tracked layer information. However, at inference time, only the echogram and the tracked surface are provided to the models. The algorithm therefore uses the prediction from the last step to form the RowBlock for the next iteration.

An early termination routine was also adopted to stop the iterative layer inference from unnecessarily continuing till the bottom of the echogram when the previously returned prediction suggests that there are no further layers in the echogram image after the current one. Some echograms have fewer layers and as such do not span the entire depth of the echogram. It is therefore not necessary to continue to search for deeper layers until the bottom of the echogram. However, to prevent premature termination of the routine, the inference routine does not quit the first time but after multiple tries with consecutive "no-layer" returns. Section 3-2-4 elaborates on the implementation of the early termination routine used in this work.

Before applying this algorithm to "real" echogram data, we designed simulated echograms in order of increasing complexity to investigate the viability of this deep learning approach. These

simulated echograms were later combined with real echogram dataset to increase the dataset's size and diversity.

3-1-1 Heuristic-based simulated echogram

To assess the feasibility of applying deep learning for real echogram layer tracking, we commence by designing a simulated echogram based on a simple heuristic and employing the iterative layer tracking algorithm on it. The simulated echogram was created based on the following assumptions:

(i) The data was collected from the dry snow zone where there is no melting and annual stratigraphy is preserved

(ii) The surface is flat

(iii) Backscatter from each pixel is not based on any scattering models but just a superposition of many coherent point targets

(iv) To model the stochasticity and incoherence between adjacent rangelines, the coherent point targets are randomly positioned in the image pixel and weighted by complex Gaussian weights

(v) The response is the ideal point target band-limited *sinc* function with no sidelobes.

(vi) Gaussian noise is added.

(vi) The layer statistics and the signal statistics are not rigorously derived from data but are based on educated guesses.

Based on these assumptions, we created echogram images such as the one shown below in Figure 3-3. We then applied the Row Block algorithm to iteratively track the layers one at a time.

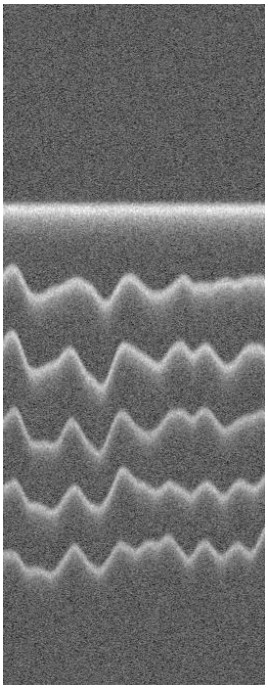


Figure 3-3 Heuristic-based simulated echogram

3-1-1-1 Shallow model for heuristic-based simulated echograms

A lightweight artificial neural network (ANN) with a single hidden layer, and multi-class output layer was designed. The training set has M examples. The training input matrix is $X = \left[x^{(1)}, x^{(2)}, \dots, x^{(M)}\right]$, where example $x^{(i)}$ is a $n_x = N_{rb} \times (2N_{cols} + 1)$ length column vector of image pixels corresponding to the (i) row block example. The associated output matrix is $Y = \left[y^{(1)}, y^{(2)}, \dots, y^{(M)}\right]$, where example $y^{(i)}$ are one-hot encoded $N_{rb} + 1$ length column vectors with the elements of the vector corresponding to {no-layer,layer in row 1, …,layer in row N_{rb}} where every element is zero except for a one in the position corresponding to the correct answer.

The hidden layer has N nodes and the network learns the optimum bias vectors $b^{[1]}$ and $b^{[2]}$ defined below, and weights $W^{[1]}$ and $\mathrm{W}^{[2]}$, also defined below as matrices representing the linear maps through the hidden layer and the output layer respectively.

The output of the hidden layer for input example (i) is given by:

$$a^{[1](i)} = g\left(W^{[1]}x^{(i)} + b^{[1]}\right) \text{ and } A^{[1]} = g\left(W^{[1]}X + b^{[1]}\right) \quad (2)$$

where $g(z) = \frac{1}{1+e^{-z}}$ is the sigmoid activation function, label $[1]$ in $a^{[1](i)}$ refers to the first layer of activation outputs, $a^{[1](i)} = \left[a_1^{[1](i)}, \dots, a_N^{[1](i)}\right]^T$ is a column vector where the subscripts from 1 to N correspond to the outputs of each hidden node, $A = \left[a^{[1](1)}, \dots, a^{[1](M)}\right]$ is the matrix formed from the outputs from each training example, $b^{[1]}$ is a bias vector term of size $N \times 1$, and $W^{[1]}$ is a $N \times n_x$ matrix.

The output of the output layer for input example (i) is given by:

$$a^{[2](i)} = g\left(W^{[2]}a^{[1](i)} + b^{[2]}\right) \text{ and } A^{[2]} = g\left(W^{[2]}X + b^{[2]}\right) \quad (3)$$

where $[2]$ refers to this being the second layer of activation outputs.

$a^{[2](i)} = \left\{a_1^{[2](i)}, \dots, a_{N_{rb}}^{[2](i)}\right\}$ is a column vector where the subscripts from 1 to N_{rb} correspond to the outputs of each output node, $A = \left[a^{[2](1)}, \dots, a^{[2](M)}\right]$ is the matrix formed from the outputs from each training example, $b^{[2]}$ is a bias vector term of size $N_{rb} \times 1$, and $W^{[2]}$ is a $N_{rb} \times N$ matrix.

A regularized logistic regression cost function $J\left(W^{[1]}, W^{[2]}\right)$ using the cross-entropy loss function is used for training the model over the entire training set

$$J(W^{[1]}, W^{[2]}) = \frac{1}{M}\left[-Y^T\, log\left(A^{[2]}\right) - (1-Y)^T\, log\left(1-A^{[2]}\right)\right] + \frac{\lambda}{2M}\left[\left|W^{[1]}\right|_2^2 + \left|W^{[2]}\right|_2^2\right] \quad (4)$$

where λ is a scalar regularization hyperparameter which scales the Frobenius norm of the weight matrices to prevent overfitting during training.

Backpropagation is used to calculate the partial derivatives of the cost function with respect to the model parameters, $W^{[1]}$ and $\mathrm{W}^{[2]}$. The cost function optimization is done using the fmincg conjugate gradient descent algorithm from MATLAB. The optimization goal is to find the model parameters that minimize the overall cost function.

3-1-1-2 Shallow model result

The training set contains 307,200 ColumnPatches taken from a subset of all the RowBlocks generated from 800 simulated Snow Radar echograms. The NN was trained on a 128 GB, 3.3 GHz, 8-core Red Hat Enterprise Linux server using MATLAB. The simulated 1000 by 256 echogram matrices were decimated to 125 by 64 to reduce the number of inputs and outputs of the NN.

For testing, 200 simulated echograms with known surface were created. Using the surface information, the first layer of each echogram was tracked, and this was used to form the row block for the next/second layer and this process continued until the termination condition was met. The following a priori information and hyper-parameters were used:

- The number of rows in a row block is set to N_{rb} = 16.
- A total of $N_{cols} = 15$ neighboring columns were used on each input (7 to the left and 7 to the right of the column being estimated).

- The number of NN layers was set to $L = 3$.
- The number of nodes in hidden layer $N = 50$,
- *no-layer* termination threshold $\gamma = 0.5$, and a regularization term $\lambda = 50$.

An example echogram image is shown in Figure 3-4a with ground truth labels as dashed lines. The NN layer tracks for this image are shown in Figure 3-4b as colored lines identifying each layer in the simulated image.

The model's performance is evaluated quantitatively using root mean squared error (RMSE) defined below where y in this case represents the index of the selected row rather than the one-hot encoded output vector. The term, $y^{(i)} - \hat{y}^{(i)}$, then represents the difference in rows between the NN estimated layer location and the ground truth layer location.

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2} \tag{5}$$

Overall, an accuracy of 92.8 % was achieved with an RMSE of 0.24 pixels. Accuracy here is defined as the percentage of the model predictions that exactly match the ground truth. In other words, the percentage of columns in the test data that the model predicted the exact row/*no-layer* state. Also, of the inexact predictions (prediction pixel errors) made by the model, only 2% of them were greater than 1 pixel.

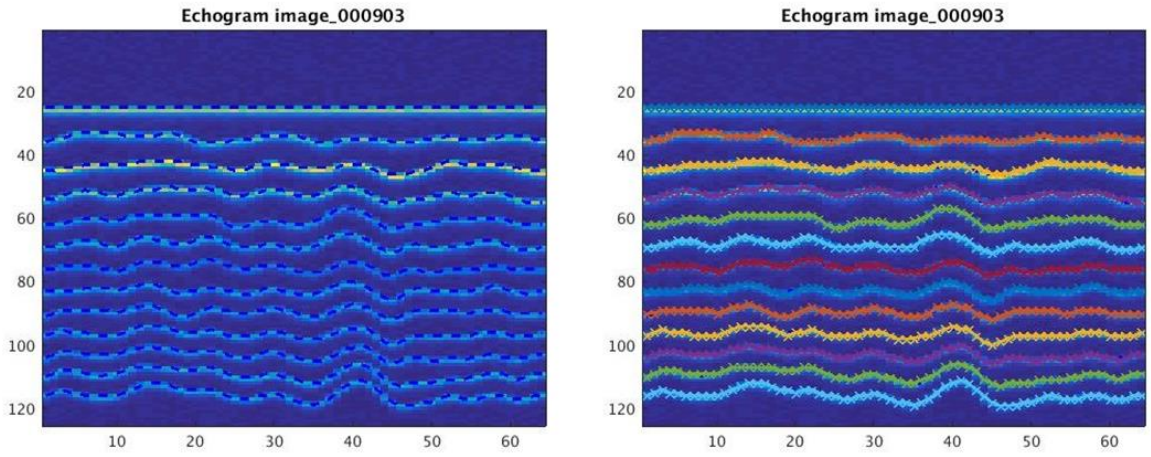


Figure 3-4: (a) Simulated echogram image with ground labels as dashed lines. (b) Image with overlaid prediction for each layer in the echogram

3-1-2 Improved simulated echograms

Following the successful application of the RowBlock algorithm for layer tracking in a simplified simulated echogram, efforts were made to enhance the simulated echogram's fidelity to match actual echogram images. The approach adopted involved identifying and replicating key physical models underlying the layering and backscatter power of the Snow Radar. This was achieved by incorporating noise, layer geometry, and echogram signal statistics, all derived from actual echogram data. The simulator model was then parameterized using statistical estimates obtained from data collected along a flight line extending from Central to Northwest Greenland, which spans diverse snow accumulation conditions within the dry snow zone.

3-1-2-1 Layer generation

To simulate echogram images with "near-real" layers, we investigated the layer geometries that exist in a sample dataset with over 200,000 rangelines (after stacking and other post-processing).

The geometry of the layers in an echogram image depends largely on the geographic and climatic conditions of the measurement site. Wet-snow and ablation snow zones usually have fewer internal layers, with markedly sloped and undulating layer geometry while dry snow zones have more internal layers which are relatively flat and parallel to each other. It should be noted that most of the sample dataset used in parameterizing our model comes from the dry snow zone. However, this is the zone of greatest interest for automated tracking due to the higher number of persistent layers compared to other snow zones.

Simulating Layer Geometries

Using the manual annotations of the sample dataset, we computed the thickness (snow accumulation) between consecutive tracked layers to understand the underlying accumulation random process. The histogram of the layer thickness reveals a Gaussian process with a slowly changing along-track mean. This suggests that, although there are local variations in each layer thickness from rangeline to rangeline, there is also a slowly varying trend in the average layer thickness over space. Another important process is the correlation between the layer thickness of different layers at a particular location. This is seen qualitatively in an echogram image such that the layers often share a similar trend in their layer geometry, and this can be attributed to the weather and topography of the imaged location. This tends to be consistent over time as each layer is deposited at a site. Therefore, to create layers with similar geometry to that in real data, all three processes need to be incorporated.

To model the high-frequency local variations, the sample dataset was divided into blocks of 1000 rangelines. We then computed the power spectral density of each layer thickness after subtracting the along-track mean in each section. Similarly, to capture the slowly varying along-

track accumulation mean that describes how the mean thickness varies across space, we computed the mean thickness and variance for each section. In future versions of the simulator, the autocorrelation function of the low frequency (slowly varying) component of the layer geometry could be considered instead of mean and variance. Given the different accumulation rates over Greenland, accumulation can vary a lot between different locations, and this is seen in the sample dataset. We, therefore, re-grouped the dataset into 4 groups of accumulation zones, based on the accumulation/thickness of the first layer: shallow, medium, high, and very high accumulation zones.

Lastly, to partly model the similar trend that exists in the thickness of the first layer and subsequent layers and the geometry of layers in an echogram, we normalized the layer thicknesses of all the layers relative to the thickness of the first layer to simulate the correlation between the layers in an echogram. This was done for all four zones and since the distribution of the normalized layer thickness is approximately Gaussian, we estimated the mean and variance of each.

One additional image attribute to note is the number of traceable layers. This parameter is defined as the number of layers in the echogram with a sufficiently high signal-to-noise ratio that detection is possible. This parameter is treated as a wide-ranging value from a uniform distribution for the training to avoid biasing the network's model parameters to a limited number of layers. The number of layers is randomly chosen from a uniform distribution between a minimum of 5 layers to a maximum of 40 for each simulated echogram. The mean thickness of the first layer is then generated from a uniform distribution based on the distribution of the first layer in the sample dataset and this determines the accumulation zone of the simulated echogram. Next, the mean thickness of each of the deeper layers is computed by multiplying the

mean thickness of the first layer with the multiplicative factor drawn from the normalized layer thickness distribution of the accumulation zone. To add the high frequency rangeline to the rangeline variation, a random Gaussian sequence with zero mean and unit variance is filtered to match the power spectral density of the layer thickness as seen in the sample dataset.

The images in Figure 3-5 and Figure 3-6 show the power spectral density of the simulated layer and real sample data respectively. Although the length of the simulated layer thickness sequence for each layer is shorter — resulting in a sparser power spectral density compared to the real data— the overall trend is consistent between them. This matching power spectral density profile demonstrates that the simulated layer thickness closely resembles that of real data, which significantly enhances the simulator's ability to generate echogram images that appear more realistic.

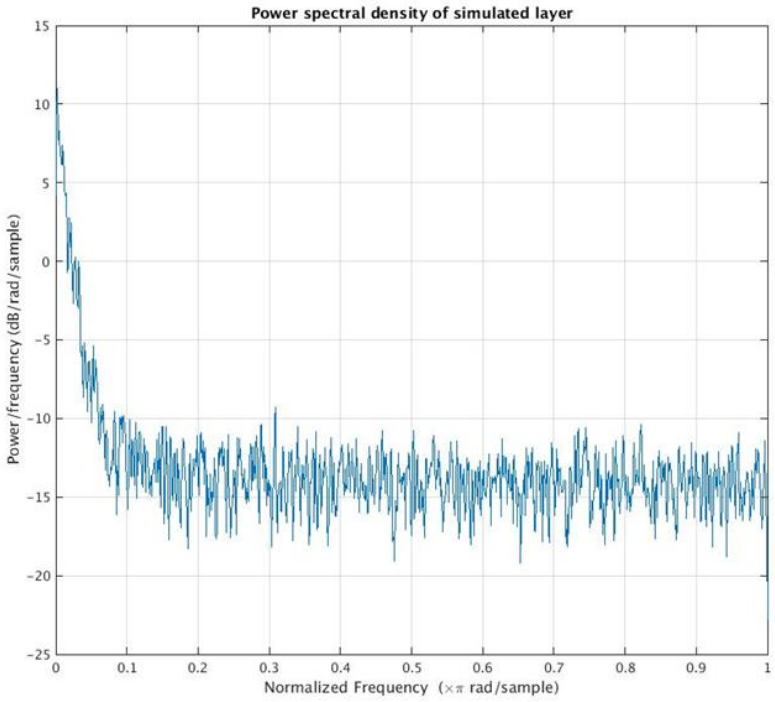


Figure 3-5: Power spectral density of simulated layer

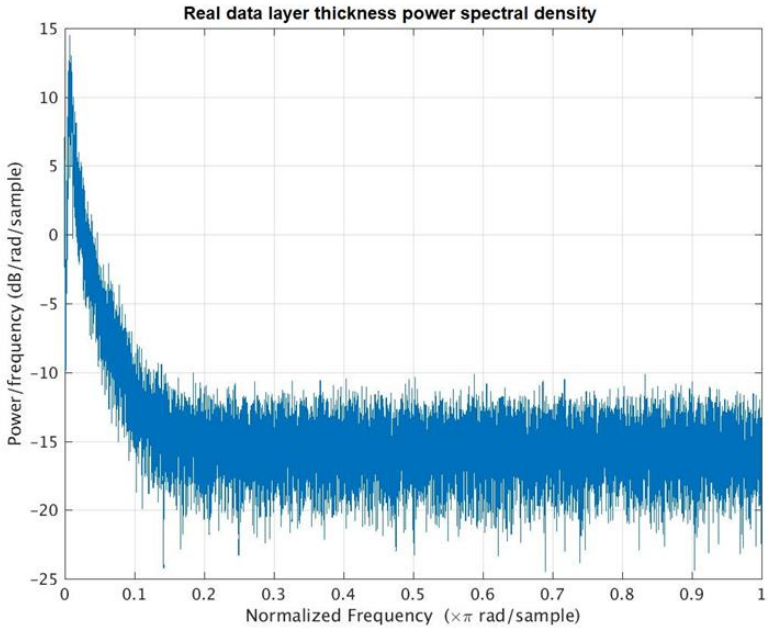


Figure 3-6: Power spectral density of real data.

3-1-2-2 Layer Power Generation

Like the layer generation process described above, we modeled the backscatter power of the simulated echogram using statistics derived from the received backscatter in the sample dataset. The layer scattering response generally follows the shape predicted by the Moore convolution model for surface altimetry [64]. The shape starts with a fast-rising edge (attributed to the RMS height of the surface) followed by a slower, exponential-like, decay (attributed to the off-nadir backscatter) with the tracked layer centered on the peak as shown in Figure 3-7.

Moore and Williams [64] show that the expectation of the power detected waveform for the surface can be modeled as the convolution of several constitutive elements including the height distribution of the layer, the pulse-limited footprint, and the expected backscatter. We assume the Born approximation [65] so that the interaction or multipath between layers can be ignored. We

model the set of layers as the superposition of each layer independently generated and then incoherently summed.

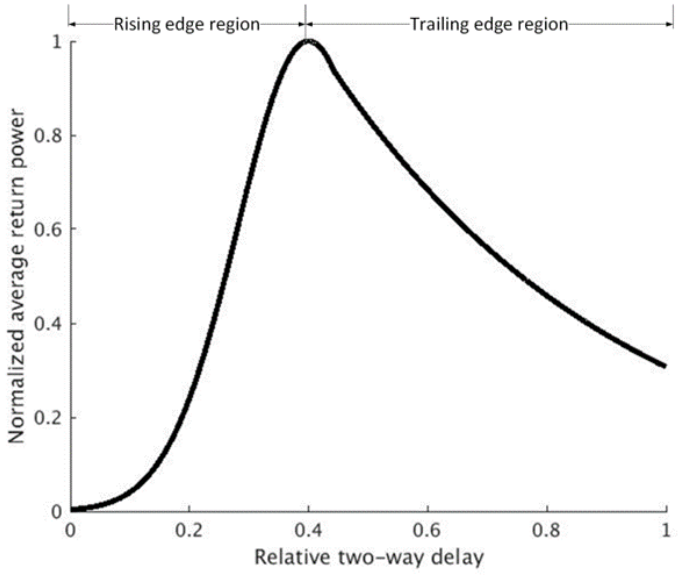


Figure 3-7: Illustration of the normalized average or expected backscatter from a layer.

Following the convolution model, we simulated the response of a layer by convolving a Gaussian waveform, which approximates the height distribution of the layer, and an exponential decay waveform, which approximates the combined pulse limited area and layer backscatter roll-off. The peak of the Gaussian is aligned with the tracked layer location and the combined return for a range line is the linear superposition of these convolved waveforms – one for each layer in that range line. Snow Radar scattering tends to be incoherent because scatterers that form a layer vary randomly throughout the snow volume that constitutes the layer. Therefore, rangelines typically have minimal to no phase correlation with neighboring rangelines.

This incoherent backscattering assumption, which is the basis of the above convolution model, is supported by the Doppler spectra along snow layers. In general, the spectrum is broad without distinct coherent peaks. The histogram of the Snow Radar data in the sample dataset, which

includes along-track incoherent averaging, shows that the distribution of the peak power along a layer fits a Chi-square distribution. Using the scaled superposition waveform as the mean power, a Gaussian random process was created to simulate the expected power returns from each rangeline. We then power detect and incoherently averaged multiple simulated rangelines to create the final simulated rangeline with the appropriate Chi-square distribution.

To parameterize the layer power generation, the mean power for layer l, denoted, m_l, is found by taking the average power of the bin that is manually tracked across all rangelines where the layer is defined. The mean power represents the backscatter received by the radar for each layer and therefore encapsulates backscatter cross-section, attenuation, and other effects. We estimated the width of a layer by calculating the range bins it took for the peak power to decline to e^{-1} of the layer peak return for all well-defined and tracked layers in the sample dataset. Based on the resulting histogram of the estimated layer width, we approximate the width of the layer peaks, d_l, by a uniform distribution between 10 to 15 range bins or rows.

The estimated along track mean peak power, m_l, for each layer is used to scale the convolved exponential and Gaussian waveform and the resulting range line signal power is given by:

$$P_{s(x)} = \sum_{l=1}^{L} m_l \, exp \frac{-(x-\mu_l)^2}{2{d_l}^2} \otimes U(x) \, exp(-\alpha_l x) \qquad \textbf{(6)}$$

where

- P_s is the expected backscatter power waveform for each rangeline.
- x is the fast time pixel index
- $U(x)$ is the unit step function

- L is the number of layers in the range line
- m_l is the mean peak power for layer
- μ_l is the location or row of layer l,
- d_l is the width of layer *l* pulled from a uniform distribution from 10 to 15 rows, and
- α_l is the exponential decay rate of the layer.

The values of m_l, d_l, and α_l are all estimated from the sample data. To estimate the decay rate of each layer, all rangelines in the sample data are grouped into *K* groups of 20 consecutive rangelines each. Each group was then incoherently averaged in the along-track dimension to produce a single filtered range line per group. Note that the data are already incoherently averaged and decimated by five during the process to generate the echograms, so the total number of incoherent averages is 100. This ensemble of *K* filtered rangelines is then used to find the backscatter peak power, $P_{peak,lk}$, and the minimum power, $P_{min,lk}$, between this and the next peak, for each filtered range line $k \in 1,...,K$ and each layer $l \in 1,...,L$.

The backscatter peak power $P_{peak,lk}$ generally corresponds to the location of the layer since the tracked layer follows the peak power. Using the range bin distance between the peak and the minimum for each layer and filtered range line, b_{lk}, we compute the estimated decay rate for each layer as follows:

$$\alpha_l = \frac{1}{K}\sum_{k=1}^{K}(-1 \,/\, b_{lk}) * \log\left(\frac{P_{min,lk}}{P_{peak,lk}}\right) \tag{7}$$

As described in Equation (7), the decay rate, α_l, is estimated from the average of the exponential curve fitted between peak power $P_{peak,lk}$ and minimum power, $P_{min,lk}$, for K adjacent rangelines.

We concluded the signal statistics analysis by estimating the background thermal noise power, P_n, from the sample dataset by estimating the power of the received signal before the surface return arrives under the assumption that there are no targets above the surface so that only thermal noise is present in the part of the image used to estimate the thermal noise. The expected thermal noise power is also assumed to be constant. The complex signal and additive noise are both pulled from additive white complex circular Gaussian noise which is then scaled by the expected signal, $P_s(x)$, and noise power P_n. Thus, the power detected rangeline with signal and noise is given below as

$$p_{(x)} = \left| s(x)\sqrt{P_s(x)} + n(x)\sqrt{P_n} \right|^2 \tag{8}$$

The distribution of the power detected signal follows an exponential distribution. The final step in the simulator is to incoherently average $\mathrm{M} = 100$ rangelines together followed by decimation in along-track by M. This results in a Chi-squared distribution with 2M degrees of freedom. This is done in the data processing to reduce signal fading which helps produce smoother and better-delineated layering in the images. Examples of echogram images simulated from this approach are shown in Figure 3-8 and Figure 3-9.

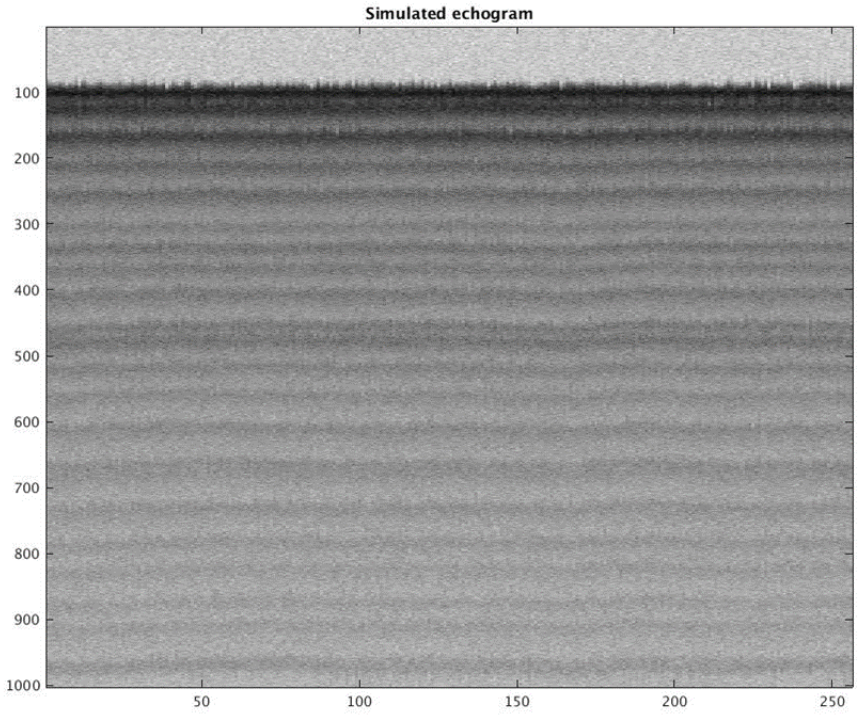


Figure 3-8: Simulated echogram image using improved model

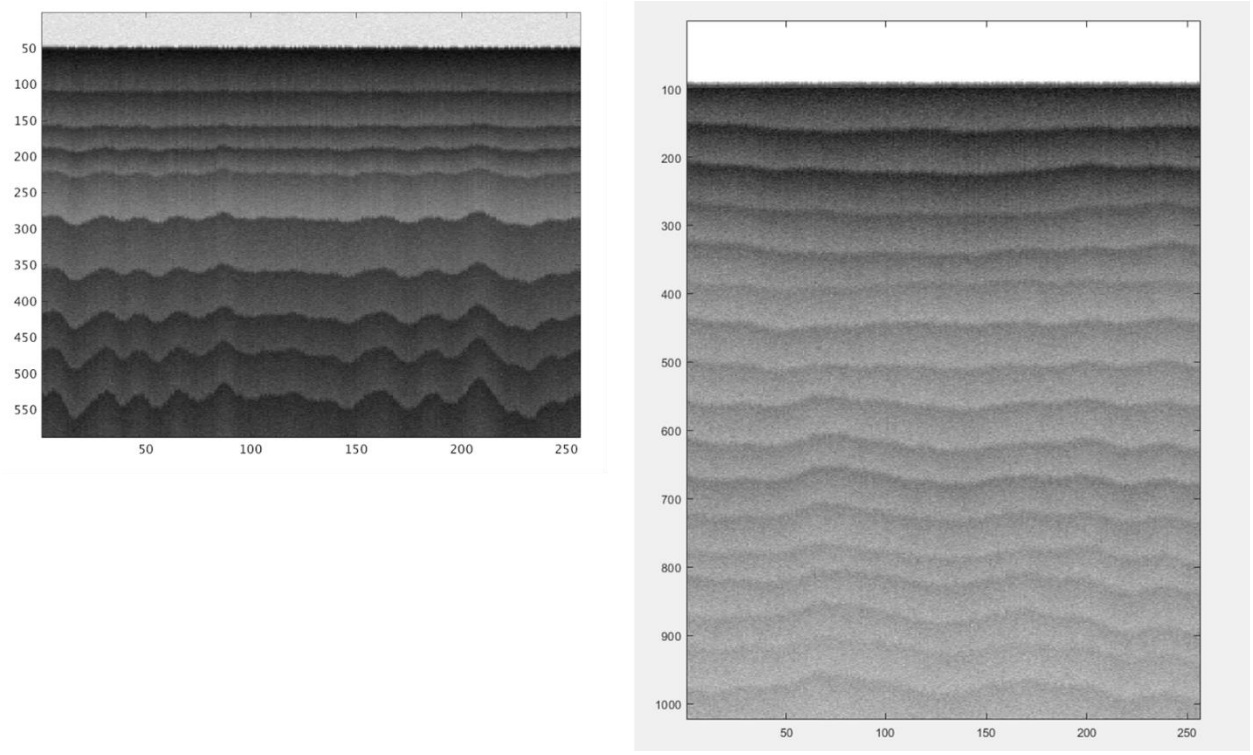


Figure 3-9: More simulated echograms with different number of layers and layer shape

3-2 Row Block on actual echogram images

The enhanced simulated echograms exhibit greater realism, closely resembling actual echograms. This improvement also increases the flexibility in manipulating echogram features, such as the number of layers, the shape of internal layers, and the along-track distance of the echogram, enabling the creation of a diverse simulated echogram dataset. To further supplement the limited supply of manually annotated real echograms, these simulated echograms were combined with additional synthetic images generated by conditional Generative Adversarial Networks (cGANs) to produce "near-real" echogram images.

However, despite these improvements, applying models trained on simulated images directly to real echograms did not yield the anticipated results, with minimal performance gains observed. This outcome, though somewhat expected, underscores the models' insufficient robustness in capturing the complex features inherent in echogram images. While visually improved, the simulated echograms still fail to encapsulate the underlying random processes necessary for deep learning models to effectively track layers.

Echograms, unlike optical images, are characterized by specific challenges such as signal fading, speckle noise, interference from non-nadir scatterers, and other system imperfections typical of SAR data. These factors, when severe, degrade image quality and complicate layer tracking, even for human analysts. Although further investigation could potentially improve the simulator and clarify why simulated images did not significantly enhance performance, the research focus was shifted to working directly with real echogram images.

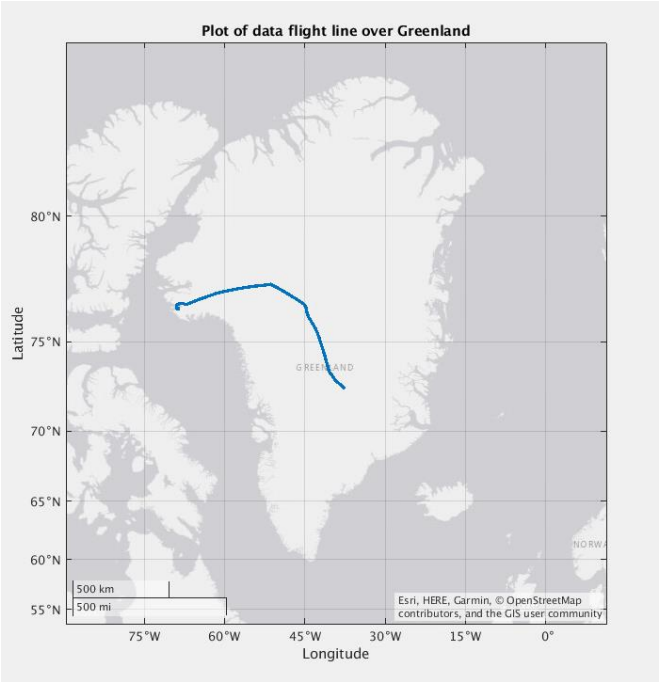


Figure 3-10: Map of Greenland showing the flight line used for estimating the statistics used for the simulated RowBlock echograms

Next, we began experimenting with various deep learning architectures using manually annotated datasets to identify the most effective architecture for modeling and learning from real echogram data. To begin, a pilot dataset with echogram image and annotated ground truth was selected. Figure 3-10 shows the flightline from which the initial manually annotated echogram dataset was derived. This dataset comprises of 1,272 training echograms created from one flightline. The performance outcomes and limitations of tested architectures are detailed in the following sections.

3-2-1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specialized class of deep learning models designed to process and analyze visual data, though they have also been effectively applied to other data

types such as audio and text. They have become the cornerstone of modern computer vision tasks such as image classification, object detection and image segmentation. They are particularly effective because they can automatically and adaptively learn spatial hierarchies of features from input images, which allows them to handle the complexity of visual data more efficiently than traditional neural networks.

CNNs are composed of multiple layers, each responsible for extracting different levels of abstraction from the input data. The key layers in a CNN include:

1. **Convolutional Layer**: This layer is the core building block of a CNN. The convolutional layer applies a set of filters (also known as kernels) to the input image. This set of learnable filters scans the input image, performing a dot product between the filter weights and the input data. These filters are usually small matrices (e.g., 3x3, 5x5) but could be larger for specific applications. They slide over the input data to extract features such as edges, textures, or patterns. This operation is called a "convolution," which gives CNNs their name. The output of the convolution operation is the “feature map” that represents the presence of specific features in the input image.

2. **Activation layer**: After the convolution operation, the output is passed through an activation layer (usually the ReLU activation function). This is an element-wise operation that replaces all negative pixel values in the feature map with zeros, which introduces non-linearity into the model, allowing it to learn more complex patterns. The sigmoid activation is also a common choice particularly on the output layer.

3. **Pooling layer**: This layer is used to reduce the spatial dimensions of the feature maps while retaining the most important information in the input. The most common type of

pooling operation is max pooling, where the maximum value is selected from a group of values in the feature map.

4. **Fully connected layers:** This layer is used in classification CNN models but not in segmentation networks. After the series of convolutional and pooling layers with intermediate non-linear activation layer insertion, the final high-level reasoning in the neural network to determine the appropriate class is done with the fully connected layer in the classification network. The alternative is the segmentation head in segmentation networks.

CNNs are widely used in deep learning due to their ability to perform template matching at different image scales, translation and illumination invariance to achieve remarkable results. Fundamental to the success of CNNs is the pooling operation. Its key function is to reduce the spatial size (width and height) of feature maps while trying to preserve the most important features within them. This reduction in dimensionality offers several advantages.

Pooling makes CNNs more efficient by lowering the number of parameters and computations after convolution. It reduces the number of values needing processing in subsequent layers, leading to faster training and reduced processing power demands. It also improves the robustness of the network by capturing the essential features within a local area making the network less sensitive to slight positional variations in the input data. This property, known as translation invariance, allows the network to recognize the same object even if it appears slightly shifted in the image. This translational invariance property of convolutional neural network is mostly due to the convolution operation performed at each image scale, but the pooling operation also contributes to this. Importantly, pooling helps to control overfitting which happens when a model

memorizes the training data too well and performs poorly on unseen examples. Pooling reduces the complexity of the data representation, making it harder for the model to simply memorize details and encouraging it to learn more generalizable features.

However, the attempt to implement 1D and 2D CNNs did not yield good performance. This is most likely due to the small dimensions of the input ColumnPatch. This precludes the application of successive pooling operations which ultimately limits the performance of the CNN architecture. Exploring larger ColumnPatch sizes by removing the undersampling in their generation or inclusion of a greater portion of the image is a potential avenue for future work that would allow testing this hypothesis of the CNNs poor performance.

3-2-2 Multilayer Perceptron (MLP)

MLP is a supervised learning algorithm architecture consisting of fully connected feed-forward artificial neurons with an input layer, one or more hidden layer(s) and an output layer. It is a specific type of artificial neural network with all the nodes in the layers densely connected. During training, through a series of forward pass and error back propagation, the network learns to approximate the latent input-output distribution. The MLP is known to be close to the universal approximator when given sufficient depth and/or width. This property makes it a good candidate for modeling input-output relationships between rangeline backscatter and the internal layer range index.

However, there are limitations to the vanilla implementation of the MLP. Having just one hidden layer with a lot of nodes makes it susceptible to overfitting the training data and will also suffer performance degradation with increased layer depth [75]. To combat this limitation, we trained a

variant of the classical MLP termed “Skip-MLP” which uses ResNet blocks and successive skip connections between all adjacent blocks. The skip connections serve as identity functions that easily allow gradient flow during model forward and backward optimization pass, effectively increasing convergence speed while mitigating vanishing or exploding gradients.

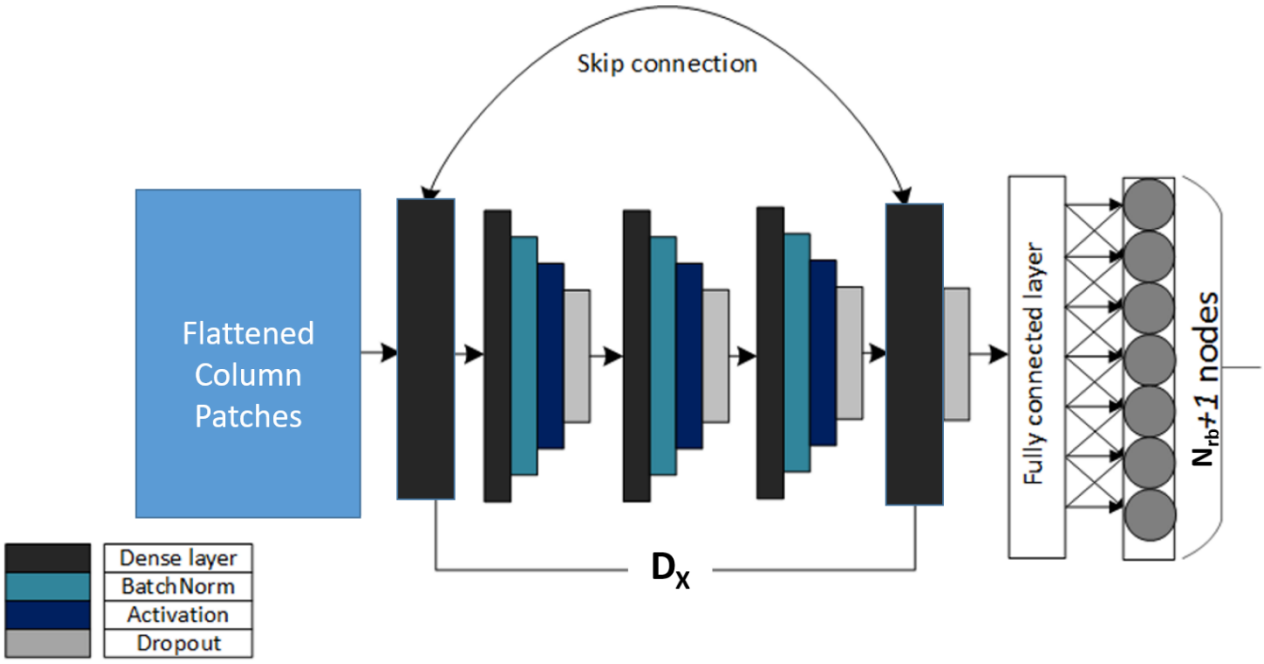


Figure 3-11: Architecture of the SkipMLP model

The SkipMLP network (shown in Figure 3-11) consists of D_x repeated blocks of sandwiched dense layers, batch normalization layers and non-linear activation functions. The nodes in the dense layers act as an approximation of a universal estimator whose weights are optimized during the training. The batch normalization layer normalizes the network weights every training batch to settle the learning process, thereby drastically reducing the number of training epochs. A Rectified Linear Unit (ReLU) differentiable activation function is applied before the dropout layer to help the model learn nonlinear mappings that may exist in the data.

The depth D_x of the network and the width (number of nodes in the hidden layer) are hyperparameters carefully chosen to achieve desired model performance. To avoid overfitting the training data due to increased parameterization, both batch normalization layer and dropout layer are used to provide regularization in each block.

Taking inspiration from the popular ResNet architecture, each block is connected to previous blocks through the skip connections. However, we extend this to form interconnection between all the blocks instead of just adjacent blocks i.e. all the blocks are connected to one another through skip connections. Skip connections have been shown to significantly reduce model inference cost by as much as 50% on datasets such as CIFAR-10 [76]. The strength of the SkipMLP architecture is that with the interconnected skip connections between all the blocks, every layer of the model can learn from all prior layers thereby reducing the chance of information loss during training. The skip connections also serve the additional purpose of preventing exploding or diminishing gradients while also preventing information degradation problem due to architecture depth since earlier inputs are always available. We used the softmax activation function on the output layer consisting of $N_{rb} + 1$ nodes as described in the basic single-hidden layer ANN earlier in this chapter.

3-2-3 Long Short-Term Memory with Position Embedding (LSTM_PE)

Given the inherently sequential nature of the geospatial information captured in the columns of a ColumnPatch, a recurrent neural architecture is a good fit to exploit this implicit property. Of the known Recurrent Neural Network (RNN) deep learning architectures, the LSTM is reported to have better performance on similar tasks because of its ability to model longer-range patterns [77], [78]. The long-term dependency modeling ability of LSTM helps it achieve better

performance by allowing the network to effectively capture and remember important information from earlier in the sequence, which is crucial for making accurate predictions or decisions later in the sequence. Therefore, we designed a version of the LSTM to track the index of layers in each training ColumnPatch.

LSTM's major component is the memory cell also known as "cell state" which keeps track of the information state in the network as weights are updated during training. LSTMs also have the forget, input and output gates that act as filters to control information that can be added or removed from the cell state based on the current input x_t and the output from previous time step h_{t-1}.

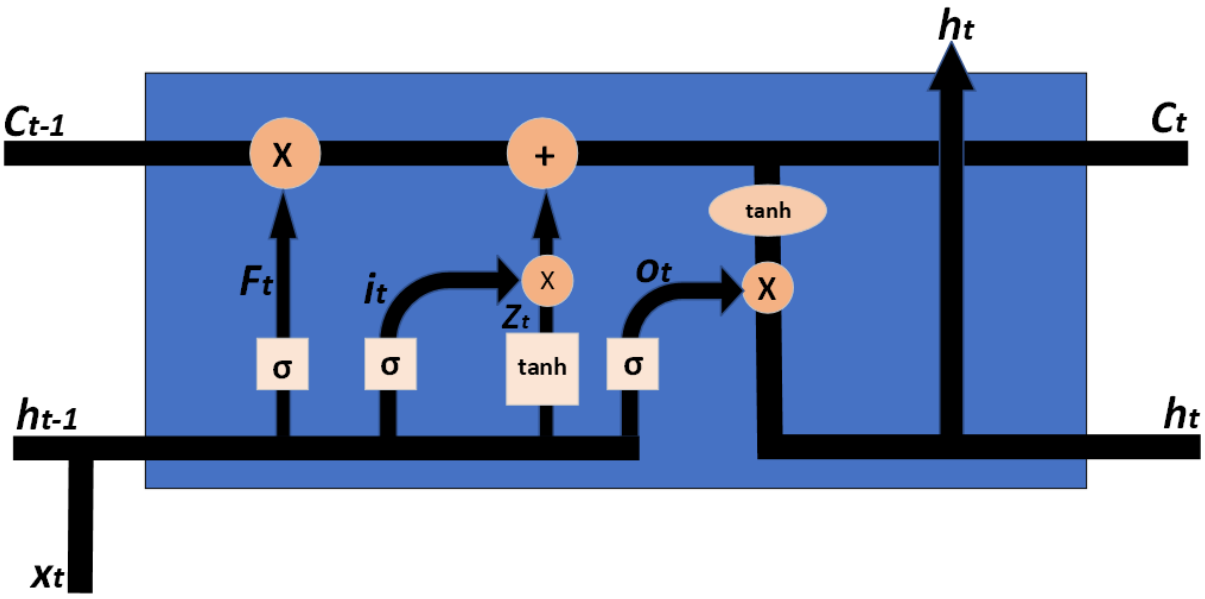


Figure 3-12: Architecture of the LSTM model

The equations in (9) describes all the gates, their interconnections and activation functions used in a LSTM model based on the seminal paper [79].

$$
\begin{aligned}
F_t &= \sigma(x_t U^f + h_{t-1} W^f) \\
i_t &= \sigma\left(x_t U^i + h_{t-1} W^i\right) \\
Z_t &= tanh(x_t U^z + h_{t-1} W^z) \\
o_t &= \sigma(x_t U^o + h_{t-1} W^o) \\
C_t &= \sigma(F_t * C_{t-1} + i_t * Z_t) \\
h_t &= tanh(C_t) * o_t
\end{aligned}
\tag{9}
$$

where t is the time step, x is the current input, and

h_{t-1} is the previous hidden state.

U is the weight matrix that connects the inputs to the hidden layer while W is the recurrent connection between the previous hidden layer and the current one.

F, i, Z describe the forget gate, input gate and the candidate hidden state. C is the next cell state formed while o is the output gate which is also the next hidden state.

Although the RowBlock creation and the number of resulting ColumnPatch's drastically expands available training data needed for deep neural network training, there is a tradeoff in the depth information captured by each ColumnPatch. However, this can be easily compensated for. A ColumnPatch from a RowBlock captures spatial information from adjacent columns that helps in tracking the layers in the RowBlock columns but does not fully capture depth correlation with deeper layers contained in the echograms. Concretely, if two ColumnPatches come from different RowBlocks (layers), but the same echogram, are provided as input to the neural network, the ColumnPatches, by themselves do not contain explicit information that would help the network learn the relationship that exists between layers at a given geographic location. This

distinction is important because of the radar power fading phenomena associated with returns for deeper internal layers. In this sense, although ColumnPatch's can be viewed as independent and identically distributed (i.i.d) inputs to the model, auxiliary depth information will improve the model's performance.

We therefore supplement each ColumnPatch with its location index in the originating echogram and provide this as auxiliary input to the LSTM. The integer location index is derived from the originating echogram as the index of the total ColumnPatch formed from the echogram. For example, for an echogram with a total of 1500 ColumnPatches, the location index of the first and last ColumnPatch are 1 and 1500 respectively. The location index effectively adds depth and additional spatial information to each ColumnPatch input which considerably improves the model's performance.

Similar to image patches used in Vision Transformers [30], the derived location index is projected onto a learnable position embedding space spanning the entire training ColumnPatches. The weights of the embedding are randomly initialized and eventually learned during training. Conclusively, the input to the LSTM_PE model is the sum of the ColumnPatch and its embedded location index.

Deep LSTMs are usually prone to overfitting because of the large number of parameters when the model is unrolled. To avoid this, we apply recurrent dropout which uses the same pattern of dropped units instead of varying dropout mask at every timestep. Finally, similar to SkipMLP, we apply a softmax activation function on the final layer to output the index of the next layer in a given input ColumnPatch.

3-2-4 Experimental setup

As shown in Figure 3-10, the dataset used for creating the RowBlock dataset was collected in the Spring of 2012 from the dry snow facies around the ridge of the Greenland ice sheet. Owing to high elevation and low temperature, little to no annual melt occurs in this region resulting in well-preserved annual stratigraphy. As a result, the imaged echograms capture historical accumulation spanning over four decades. The internal layers in the echograms were manually tracked using semi-automated layer picking software in the Open Polar Radar Toolbox developed at CReSIS [80]. The picking tool is based on the Viterbi algorithm [81] with implementation details in [45], [48]. A total of 1786 echogram blocks were tracked, with 50% overlap between consecutive blocks. Also, only the top 28 layers were tracked to maintain consistency in the tracked echograms.

Splitting the echogram matrix into RowBlocks and ColumnPatches effectively expands the limited manually annotated ground truth data to a total of 2,094,400 ColumnPatches which forms a dataset large enough to train a deep neural network. However, a methodological approach is required to divide the data into echogram training, validation, and test sets.

First, the ColumnPatches in each set should be diverse and represent the different accumulation patterns in the entire dataset to train a “generalizable” model. Secondly, and more importantly, it is crucial to carefully separate ColumnPatches derived from echograms in the training set from those in the test and validation sets. Creating ColumnPatches first from all available echograms and dividing them into train and evaluation sets can result in data leakage because ColumnPatches overlap each other so that using a ColumnPatch in training that has a lot of overlap with a ColumnPatch in the other sets could lead to biased results.

To avoid both issues, we employed a simple interspersing of the echograms following the order of the echograms along the contiguous flight path which transverse different accumulation conditions, although the flight path was mostly over the dry snow zone. Concretely, the final test set was created by alternating every 20 echograms to make a total of 200 echogram test set. We followed the same pattern to create the training and validation set to give a split of 1232 training echograms and 354 validation echograms. This approach ensures that each set is independent with no leakage of information between sets and that each set covered a diverse set of snow accumulation patterns in order to reduce the chance of covariance shifts between the sets

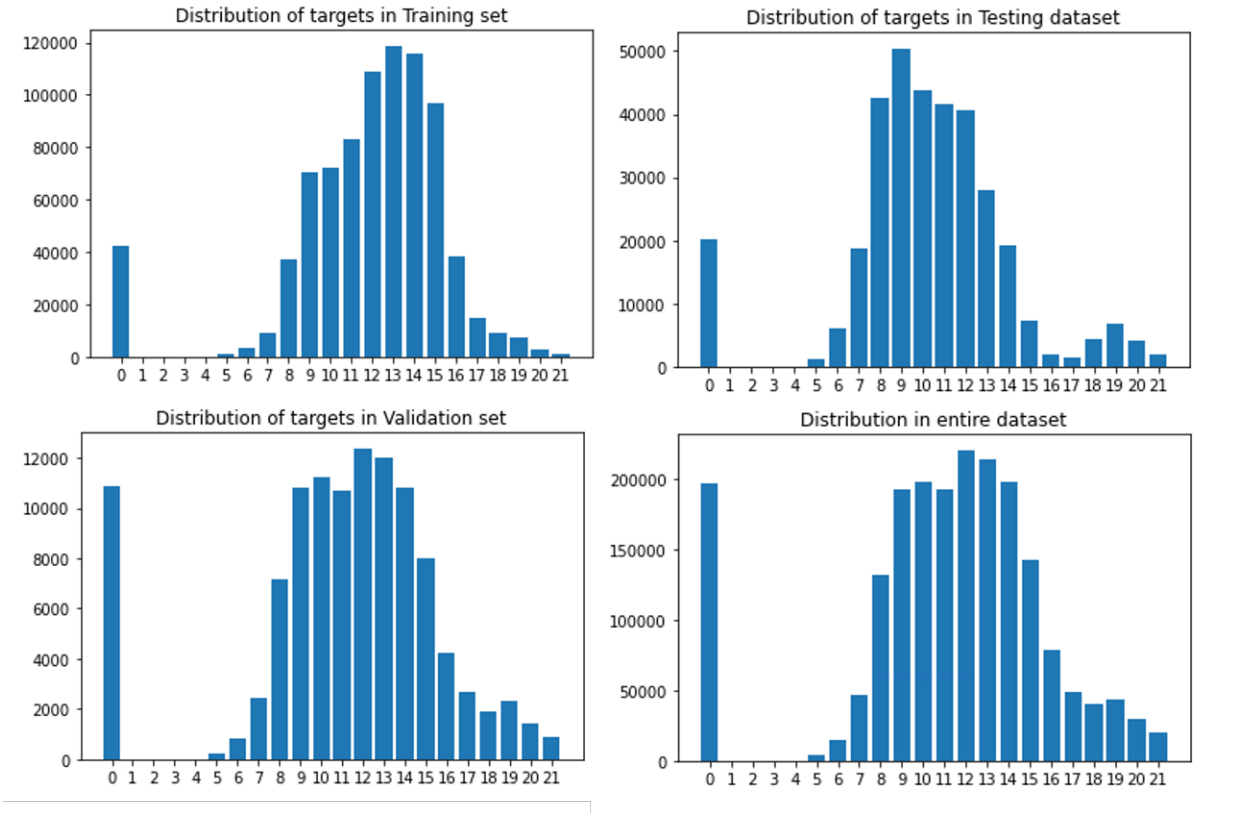


Figure 3-13: Distribution of targets (number of rows to the next layer) in the training set, test set, validation set and the entire dataset.

3-2-4-1 Implementation details

Initial echogram matrices of size 1664x256 were decimated by 4 to 416x64 to keep the training time and compute expense tractable. The RowBlocks are formed from the decimated echograms invariably dictating the layer spacing and the value of N_{rb}. We set $N_{rb} = 21$ and $N_{cols} = 7$ resulting in an input shape of 21x15 for both models. The training and validation set consist of 1,508,660 and 377,164 samples respectively. For SkipMLP, we used 512 nodes in each dense layer and a dropout of 0.3 in each block. A total of $D_x = 10$ blocks were used to increase the representation power of the model.

Similar hyperparameters were chosen for the LSTM_PE model. Each layer has 512 hidden nodes and a recurrent dropout rate of 0.3 between each layer. We limited the depth of layers in LSTM_PE to D_x = 3 to avoid overfitting the training set. This resulted in 7M and 10M trainable parameters for the SkipMLP and LSTM_PE models respectively. Adam optimizer was used for both models with an initial learning rate of 1e-3 that is reduced by a factor of 0.25 after 30 epochs at a plateau and trained for a total of 200 epochs. We trained both models with a batch size of 128 on a single Intel(R) Core(TM) i9-10900K 5.3GHz CPU with NVIDIA RTX A5000 GPU.

3-2-4-2 Model Results

We report Accuracy, Precision, Recall, and F1-score for both models in Table 2. The accuracy is defined as the percentage of predictions made by the model that exactly match the ground truth labels (i.e. the total of all the correct predictions divided by the total of all predictions). This domain-specific accuracy provides a direct measure of the model's performance in correctly

identifying the correct layer pixel in each test ColumnPatch. Precision refers to the proportion of correct predictions for a particular class (true positives) divided by all predictions to that class including predictions that should have predicted a different class (true positives plus false positives). Precision indicates how well the model avoids false positives. Recall is similar but divides the correct predictions (true positives) for a class by the total number of instances for that class including predictions that were assigned to other classes (true positives plus false negatives). Recall reflects the model's ability to capture all instances of a class. The reported values of Recall and Precision are the weighted average of the recall and precision of each class where the weighting is based on the frequency of each class in the test dataset. This weighted approach ensures that the performance metrics reflect the distribution of classes, providing a more balanced evaluation of the model's ability to generalize across different classes. Finally, the F1-score is the harmonic mean of precision and recall (i.e. the reciprocal of the mean of the reciprocals of precision and recall) – the harmonic mean is designed for averaging rates or percentages.

Table 2: *Performance metrics for the Skip-MLP and LSTM-PE*

Metric	**Skip-MLP**	**LSTM-PE**
Accuracy	0.81	**0.88**
Precision	0.78	**0.89**

Metric	Skip-MLP	LSTM-PE
Recall	0.81	**0.88**
F1-score	0.80	**0.89**

For all the metrics reported, the LSTM_PE model achieves better performance than the Skip_MLP. This is likely because of the latent sequential structure in rangeline information that is exploited by the LSTM architecture. While the Skip_MLP model's performance is fair, the flattening of the input ColumnPatches obliterates spatial information in the data and this might be responsible for the slight performance dip. This proves that incorporating “some” of our knowledge of the data in the design of the model architecture has potential gains. The recurrence of the LSTM architecture is able to take advantage of the sequential layer information in the columns of the input ColumnPatch to achieve better performance. However, attempts to increase the modeling power of the LSTM by increasing $D_x > 3$ led to the exploding gradient RNN issue.

3-2-4-3 Reconstruction of Echograms from ColumnPatch predictions, result and discussion

The goal of the automatic trackers is to track the internal layers in radar echograms; not in the independent ColumnPatches. Therefore, we apply the models to track internal layers in the 200 test echograms forming RowBlocks and ColumnPatches and report the performance.

Again, we highlight the difference in the algorithm's routine during training and inference. For training, the tracked internal layers (ground truth) are available and the row block for a layer is formed using the tracked layer from ground truth. However, at inference time, only the echogram and the tracked surface are available. The routine to reconstruct the echograms from the ColumnPatches and track the internal layers is outlined below.

Given an echogram and its tracked surface, the goal is to track all the internal layers with the total number of layers unknown apriori. The first row block is formed using the available surface and the model is used to predict the next layer location in each ColumnPatch. The result of this is then used to form the next row block from which the next (second) internal layer will be traced. This continues until all the internal layers in the echogram are traced and the model returns "no-layer" for all the columns in the last row block formed. However, there are few situations that might affect the overall tracking performance in an echogram.

First, due to the sequential nature of the layer predictions, an error in an earlier row block can cascade to deeper layers resulting in overall poor performance. An example of this is when the prediction for a ColumnPatch in the current row block is wrong e.g. it is several pixels different from what it should be. In simple cases where there are only rare occurrences of this, it is somewhat easy to detect and correct. Since it is expected that the predictions of adjacent columns should be similar, then, a sudden spike in layer location is not physically meaningful and can be detected. We detect such few occurrences using a simple local mode filter applied to each layer prediction to detect predictions that may be off. The identified wrong predictions are replaced with the modal values of neighboring ColumnPatches. This solution is not robust to larger groups of errors.

In cases where some ColumnPatches in a row block incorrectly return a "no-layer" class, it becomes difficult to create the next row block since a complete prior layer prediction is needed to create the next row block. We can categorize these missing layer detections into three main scenarios:

a. **Deeper Layer:** The “no layer” might be correct and the true layer is located deeper in the snowpack than the current row block size can capture. This can happen when there is a lot of snow accumulation in a single year so that the layer is thick enough to be thicker/deeper than our row block is.

b. **Truly Missing Layer:** The scattering from an annual layer may not be enough to distinguish it from the background.

c. **Model Prediction Error:** In some cases, the model might simply make a mistake.

The adjacency of the missing layers can help distinguish which situation is at play - if all the columns return the "no-layer" class, it is believed that there truly is no layer in those columns, otherwise, it is deemed a false negative prediction. To correct the issue and form the next row block, we use $\frac{N_{rb}}{2}$ as a "placeholder layer" but do not include this in the saved tracked layer.

Lastly, the tracker could stop too early if it receives a "no-layer" class for all the ColumnPatches in a row block. To avoid this, the inference routine does not stop the first time all the ColumnPatch predictions return "no-layer" except when the prediction is already at the bottom of the input echogram. Again, we use $\frac{N_{rb}}{2}$ as a placeholder layer to form a "tentative row block" to search if there are still deeper layers. It is when this also returns all "no-layer" class that the iterative routine quits.

3-2-4-4 Tracking error analysis

The metrics reported in Table 2 highlights the performance of the models on the ColumnPatches and not directly on the tracking accuracy in echogram images. Hence, new metrics that investigate the tracking efficiency of the models are introduced. We used a variant of the well-known mean absolute error termed N-pixel accuracies. The N-pixel accuracy is calculated by comparing the prediction with the ground truth and reporting the percentage of absolute errors that are above a certain number of pixels.

$$N_{pixel-accuracy} = \frac{1}{M}\sum_{i=1}^{M} |y^{(i)} - \hat{y}^{(i)}| <= N \tag{10}$$

where $y^{(i)}$ is the ground truth and $\hat{y}^{(i)}$ is the model's prediction both for the (i)th example. M is the total number of echograms in the test set.

Table 3 shows the N-pixel accuracy for the 200 test echograms for both models. This result is on the decimated echogram of size 256 x 64. In all 3 reported accuracies, LSTM_PE performs better than Skip_MLP. The 1-pixel accuracy is the toughest to achieve because it requires that the prediction and the ground truth label match exactly. 2-pixel and 3-pixel accuracies allow some room for imprecision in the models' predictions.

Table 3: *N-pixel accuracy for the Skip-MLP and LSTM-PE*

Metric	Skip-MLP	LSTM-PE
1- pixel accuracy	0.725	**0.782**

Metric	Skip-MLP	LSTM-PE
2- pixel accuracy	0.762	**0.798**
3- pixel accuracy	0.805	**0.830**

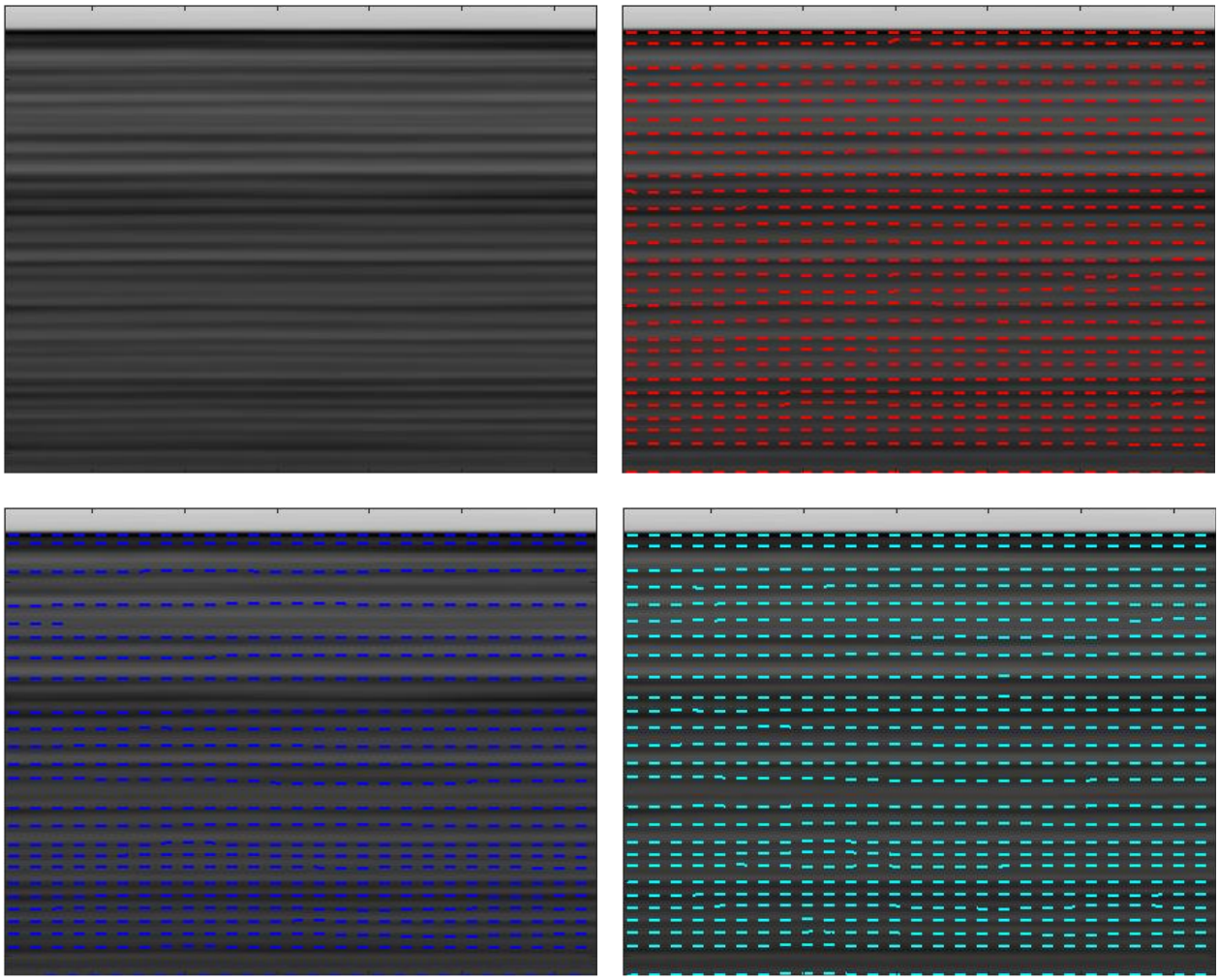


Figure 3-14: (a) Decimated and filtered echogram (b.) Annotated Ground truth (c) Skip_MLP predictions and (d) LSTM_PE predictions

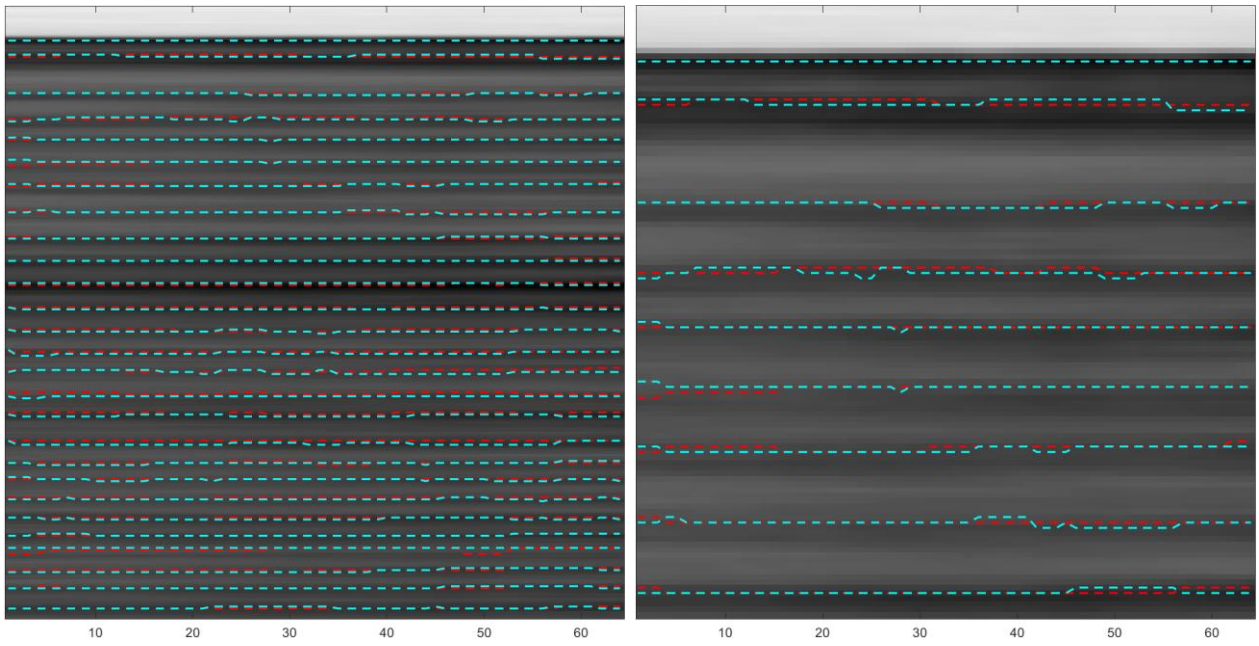


Figure 3-15: (a) Echogram with Skip_MLP predictions (in red) and LSTM_PE predictions (in cyan) (b) magnified view

3-2-4-5 Application to other untracked flight lines

There were setbacks when attempting to apply the models trained with the RowBlock algorithm to echograms from flightlines other than the one used for training. Initially, these flightline echograms needed to be decimated to match the training configuration. The model performed

moderately well on a few handpicked echograms that were very similar to the training set but was very brittle when the echograms slightly differed. This poor generalization is likely because the N_{rb}= 21 used for training did not match the layer statistics of the new dataset. While extensive research can be done to optimize the choice of N_{rb} that will generalize broadly to the large untracked dataset, the high variability of spatial accumulation over the polar ice sheet suggests that there is likely not one value that can generalize to multiple flightlines.

3-2-5 Initial efforts of applying deep learning models to 5km echograms frames

The success of the RowBlock algorithm on its test set demonstrated that deep learning algorithms are a viable option for snow layer tracking. However, given the limitations of the current implementation of the RowBlock algorithm, the focus was shifted toward exploring deep learning models that can process "full echograms" without the need to first disassemble them into smaller units. Such algorithms will take full advantage of long-range spatiotemporal information inherent in full echograms, enabling them to not only identify the layer pixels but learn generalizable features such as the spatiotemporal correlation between layer pixel values and layer spacing in both along-track and layer depth axis.

This approach, however, comes with the associated challenge of requiring a large dataset to effectively train the models. Along with this are the required computational resources needed to train such models. To begin this effort, we experimented with decimated echograms to mitigate the computational needs.

The initial goal at this stage of the research is to train using the available decimated echograms to assess the feasibility of using deep learning models for accurate snow layer tracking in full echogram images. If successful, these models can be scaled to handle full-sized echograms, with the expectation that more ground-truth echograms will be generated to support further model development and validation.

3-2-5-1 Binary segmentation on decimated echograms

Two models were designed to investigate two different image segmentation paradigms: namely binary image segmentation and deep-tiered multi-layer segmentation. Details of the training paradigm and model architecture are delayed to sections 4-4-1 and 4-4-2 respectively. Here, we only show the qualitative performance of the designed model as proof of concept for applying image segmentation deep learning models directly to “whole” echogram images.

For binary segmentation, we designed the well-known U-Net [82] architecture with slight modifications. The U-Net architecture is so-called because the encoder and decoder network are stacked beside each other to form a U-shaped architecture with skip connections between corresponding levels. Figure 3-16 shows the symmetrical U-Net encoder-decoder architecture with shared skip connections between contemporary feature map stages.

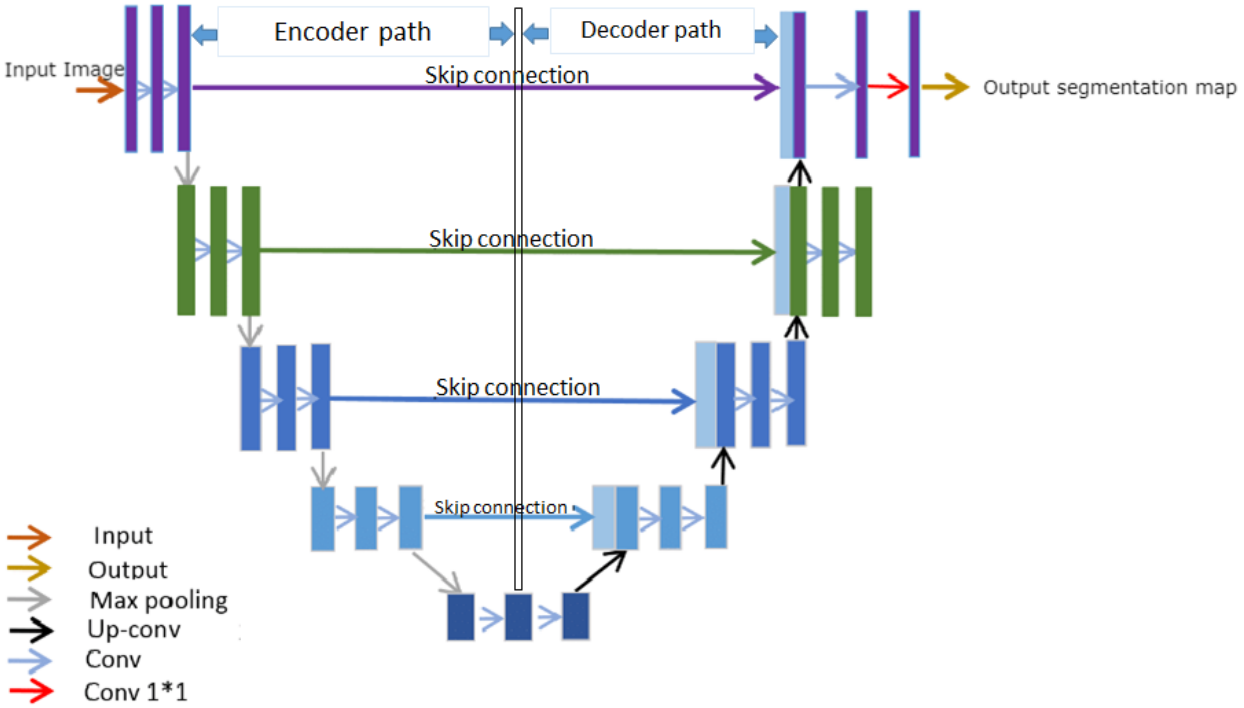


Figure 3-16: U-Net binary segmentation architecture

At the input of the encoder, decimated echogram images of size 416x64 are fed in and this is followed by blocks of convolutional layers. The first convolutional layer has an asymmetric filter size of 7x5 to account for the difference in the height and width of the radar image. Other filters used in the convolutional layers in the contraction and expansion path are symmetric 3x3 filters. The subsequent convolutional layers have a dropout layer, batch normalization and ReLU activation function but these are not shown in the figure to avoid overcomplicating the diagram.

Each convolution block, dropout, and batch normalization layer in the encoder path is followed by a 2x2 max pooling represented by the gray arrow to downsize the image so that the next convolutional layers can learn local features from a downsized image. Similarly, each block in the decoder path is first concatenated with the corresponding image of equal resolution in the encoder path, followed by two convolutional layers, and then is upsampled by a 2x2 kernel to

resample the image to finer resolution. The up-convolution operation is depicted by the “black” arrow in the network image. The series of convolution, downsampling, upsampling and concatenation of the encoder and decoder blocks helps the model to learn both global features at low resolution and local features at fine resolution and the correlation between them – hence the ability of the network to perform the segmentation of the layers.

The model is trained with the Nesterov-accelerated Adaptive Moment Estimation (NAdam) optimizer using a binary-cross entropy loss function for 100 epochs. To combat the effect of class imbalance between the less abundant layer pixels and more abundant no-layer pixels, we applied a 1:10 ratio class weighting and also used focal loss in addition to cross entropy loss as the model’s objective function.

Figure 3-17 and Figure 3-18 are qualitative outputs of the model when used for inference on the test set. As can be seen from both images, the model shows promising results in tracking the layers in the decimated echograms. This indicates that training the models on the complete echogram is a viable approach. In Figure 3-17, the echogram has a very simplistic and almost perfectly straight layer structure which may suggest that the trained model can only track those but Figure 3-18 shows that the model can learn sloped orientations too.

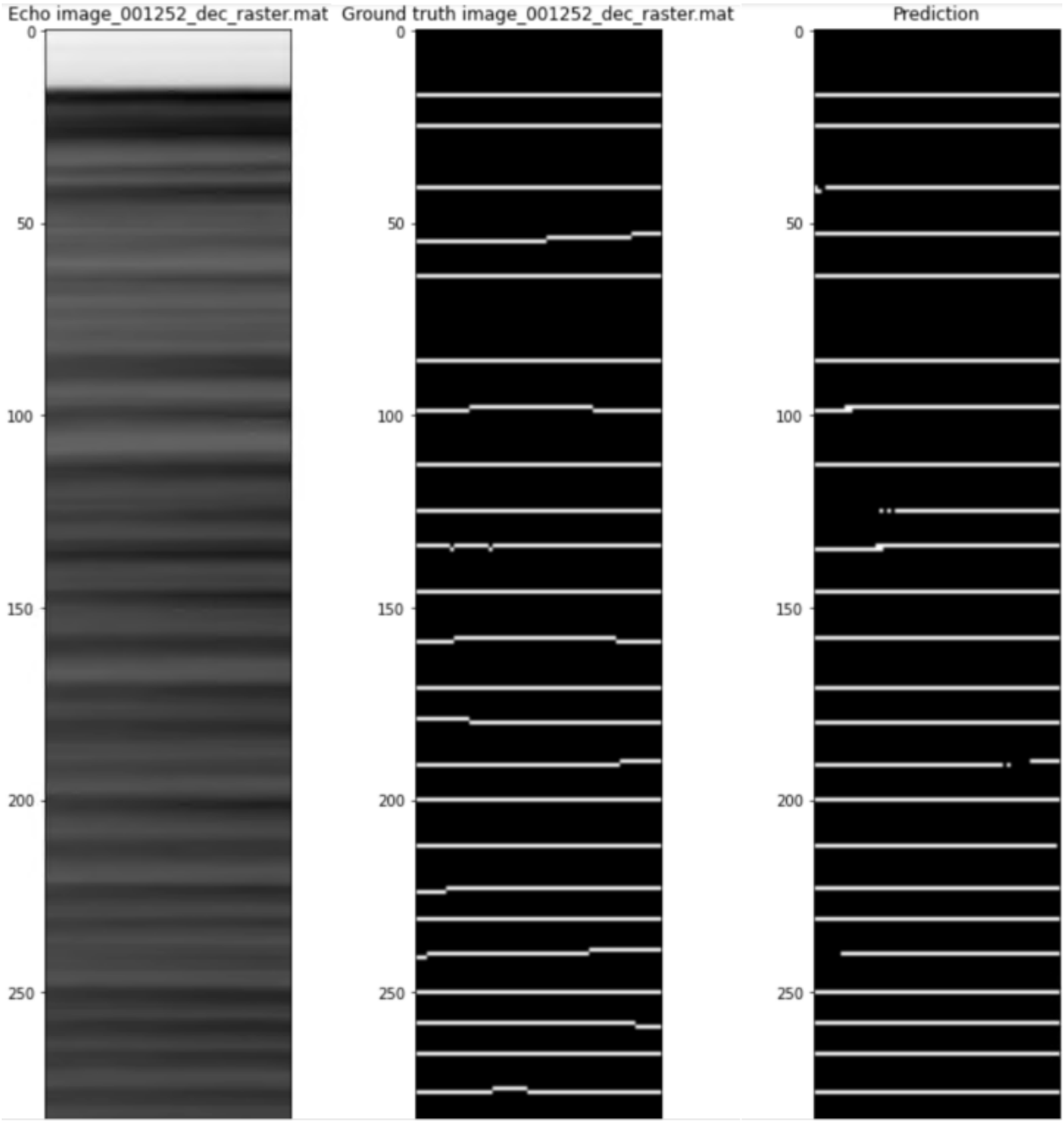


Figure 3-17: Example binary segmentation algorithm qualitative output on a decimated image

(a.) Decimated image (b.) Ground truth (c.) Model Output

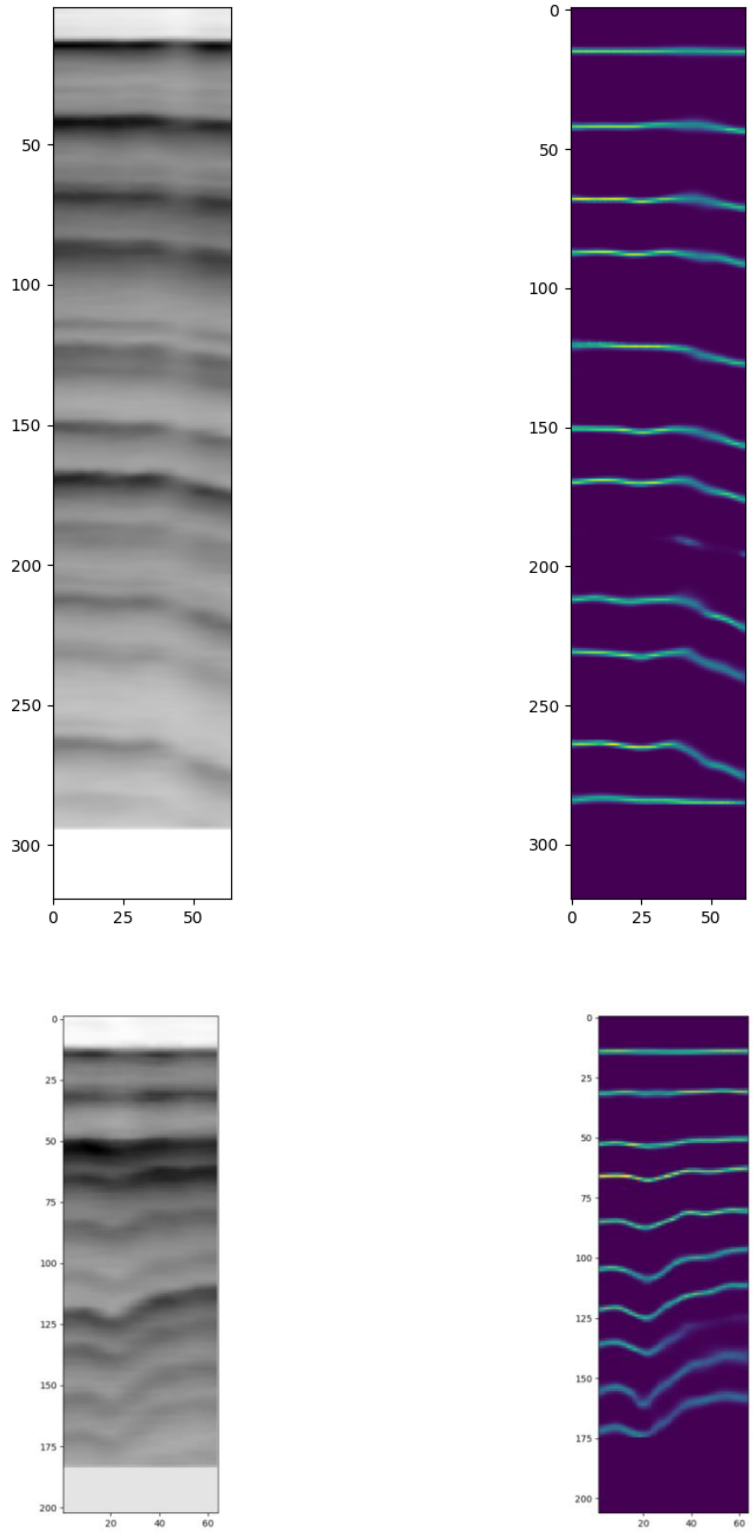


Figure 3-18: Two more examples of binary segmentation algorithm qualitative output on decimated images. (a.) Decimated image (b.) Model output

3-2-5-2 Multiclass semantic segmentation on decimated images

Similarly, a vision-transformer based model was developed to apply multi-class semantic segmentation architecture on the decimated echogram. Again, although this architecture is introduced here, a comprehensive discussion of the multi-class semantic segmentation architecture is deferred to Sections 4-4-2 and 5-2.

The vision-transformer architecture employs the layer isomorphism paradigm such that no decimation was done in the architecture to ensure that the input dimension matches the output dimension needed for pixel-wise dense classification.

The number of multiclass layers in the training echograms was limited to the top 30 layers in each echogram. Table 4 shows the training hyperparameters.

Table 4: *EchoViT training hyperparameters*

Echogram Vision Transformer	**Training Hyperparameters**
Batch size	4
Learning rate	1e-3
Number of heads	20
Number of Transformer layers	10
Input image shape	416x64
Embedding dimension	416
MLP dense units	[2048, 1024, 512, 64]
Convolution stem activation function	GeLU
Prediction Full Convolution activation function	Softmax
Number of prediction classes	30

Number of epochs	250
Loss function	Categorical Cross Entropy
Test Accuracy	93%

Accuracy is usually not regarded as a good metric for segmentation tasks because of the class imbalance and possible misalignment of ground truth and prediction pixel values. Instead, other metrics that more accurately reflect model performance are presented in the table below.

Table 5: *Semantic segmentation metrics for semantic segmentation of decimated echograms*

Layer	*Precision*	*Recall*	*F1-score*	*Class support*
0	0.990	0.990	0.990	2486464
1	0.970	0.970	0.970	142912
2	0.950	0.970	0.960	196928
3	0.950	0.950	0.950	154112
4	0.920	0.940	0.930	115520
5	0.940	0.930	0.940	109056
6	0.930	0.950	0.940	87232
7	0.970	0.950	0.960	135936
8	0.960	0.960	0.960	125888
9	0.950	0.950	0.950	123776
10	0.940	0.940	0.940	126720
11	0.940	0.940	0.940	129600
12	0.940	0.940	0.940	143168

13	0.950	0.950	0.950	104320
14	0.940	0.930	0.930	101568
15	0.920	0.930	0.930	104768
16	0.900	0.920	0.910	87040
17	0.940	0.940	0.940	133376
18	0.930	0.930	0.930	100864
19	0.920	0.910	0.920	75008
20	0.900	0.920	0.910	70016
21	0.930	0.920	0.920	83328
22	0.930	0.920	0.920	83264
23	0.860	0.900	0.880	47104
24	0.890	0.890	0.890	29056
25	0.910	0.890	0.900	31680
26	0.900	0.910	0.900	38336
27	1.000	1.000	1.000	0
28	1.000	1.000	1.000	0
29	0.880	0.930	0.900	24640

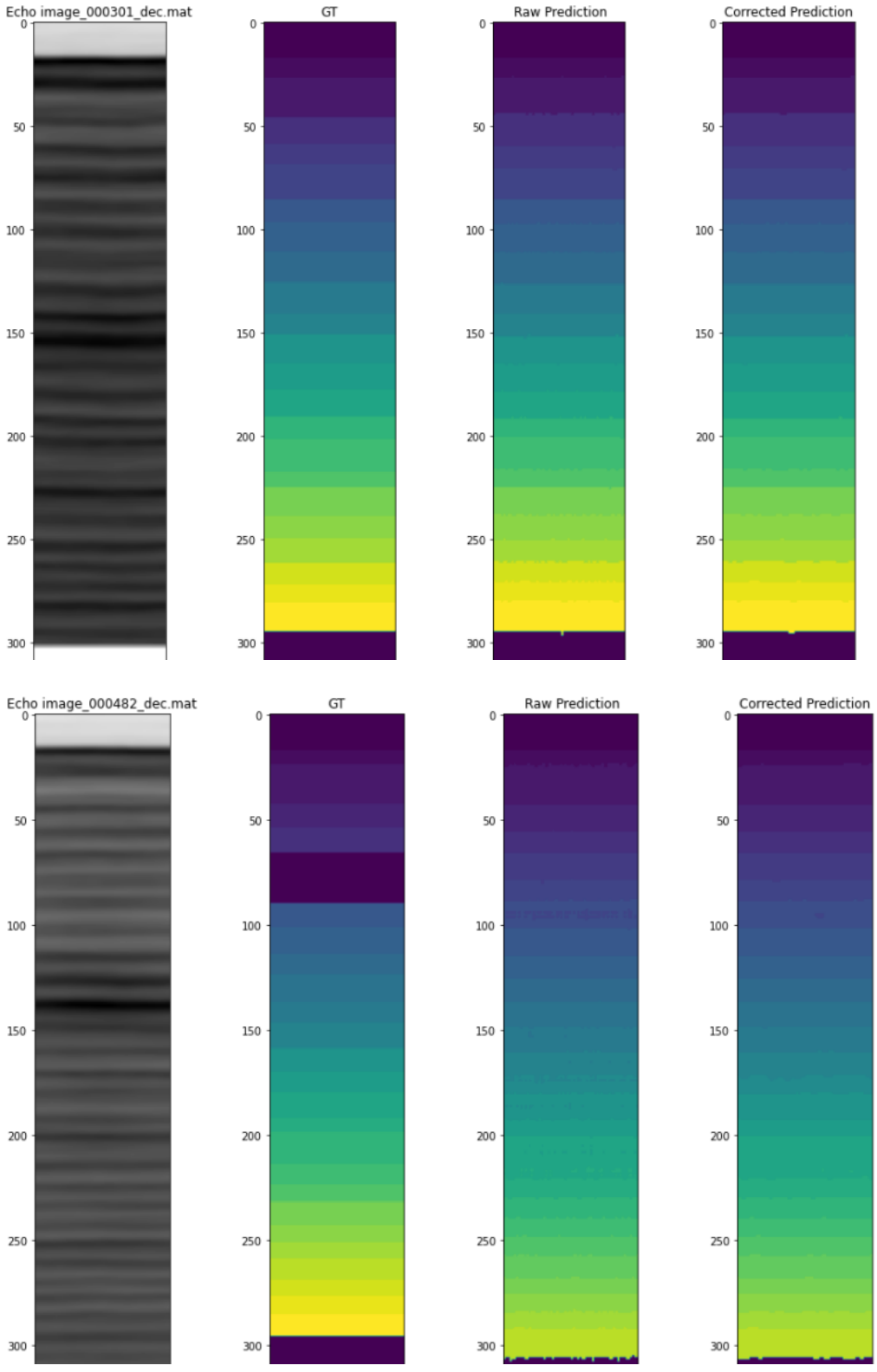


Figure 3-19: Two examples of semantic segmentation algorithm qualitative output on decimated images. (a.) Decimated image (b.) GT (c.) Raw prediction (d.) Filtered prediction

The promising results of the binary segmentation and multiclass segmentation on the decimated echogram images motivated the pursuit of a more extensive implementation on full echograms.

However, a detailed look at the segmentation results, particularly the semantic segmentation, suggests overfitting of the model but further work was not done to verify or correct this since the initial goal was only to confirm the viability of the approach using the decimated images but to train a larger dataset with full sized echograms.

Chapter 4 METHODOLOGY (2)

4-1 – Creating large-scale dataset of full echograms.

An important goal of this research is to create a larger and standardized echogram dataset consisting of well delineated training, test, and validation sets that can support the training of various deep learning algorithms. Standardized datasets are crucial for applying deep learning to novel scientific problems. They establish a common ground for researchers, enabling fair comparisons of deep learning model performance across different architectures and training methods. Reproducibility of results is ensured, fostering scientific progress through verification and iterative development. Moreso, collaboration and data sharing become efficient which is particularly valuable for data-intensive problems such as the radar echogram layer tracking. The hope is that providing such standardized datasets will streamline the future research efforts by eliminating the need for individual data pre-processing, saving researchers valuable time and effort and paving the way for efficient scientific discovery through deep learning.

While the volume of training data is undeniably important for the success of deep learning, true progress depends equally on the quality and diversity of that data. This implies leveraging prior knowledge gained from earlier experiments to better condition the data used to create the dataset, thereby achieving performance gains. The integration of domain-specific insights into the data preparation process is crucial for optimizing the dataset's utility in training effective models

For instance, an examination of echogram images reveals certain characteristics that may be challenging for algorithms to correctly identify and track internal layers. Using the echogram image in Figure 4-1 as an example, although some layers beneath the surface can be seen, many

are faint and may be difficult to track by traditional methods. This suggests that additional signal processing-based preconditioning of the echograms could enhance the delineation of the layers, thereby improving the performance of the proposed automatic layer trackers. Such preprocessing steps are essential to highlight the features in the echograms that are critical for accurate layer tracking.

Therefore, prior to applying the deep learning algorithm to the echograms, the following preprocessing steps were applied in the order outlined below to better delineate the interface between successive snow layers.

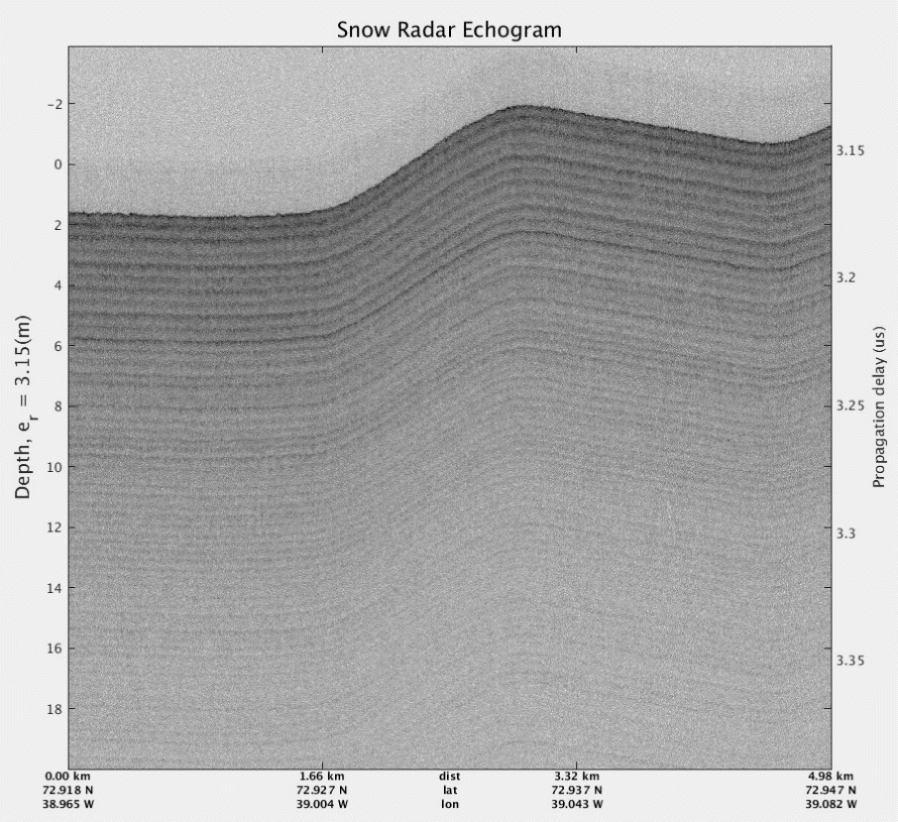


Figure 4-1: Snow Radar echogram image

4-1-1 Surface tracking

Figure 4-1 depicts an echogram before surface flattening. Variations in terrain relief and aircraft elevation cause the surface index of the received backscatter to vary along the flight line. However, to enhance the visualization of internal layers and facilitate along-track processing - such as incoherent averaging of neighboring rangelines - it is crucial to represent the surface in the echogram image as perfectly flat.

Since the surface bin in a rangeline typically corresponds to the first significant peak in received backscatter, adaptive threshold techniques can be readily employed for its detection and tracking. Consequently, traditional signal processing approaches are used for surface tracking, while deep learning is reserved for the more challenging task of tracking closely spaced internal layers.

The air-snow interface, or surface, is the first layer detected within an echogram data matrix. This surface marks the initial point of interaction between the radar signal transmitted from the aircraft and the ground snow, resulting in a significant increase in backscatter power compared to earlier received signals. Although the surface return is not confined to a single fast-time bin, there is a noticeable rise in backscatter as the radar pulse encounters the surface.

Due to variations in backscatter power and surface return across different rangelines, an adaptive threshold is required to accurately track the surface bin. To address this, an adaptive detection algorithm was developed, utilizing each rangeline's data to estimate the threshold. This approach ensures reliable and precise surface tracking despite the inherent variability in the echogram data

Concretely, we employ a form of Cell Averaging - Constant False Alarm Rate (CA-CFAR) algorithm to set the dynamic threshold for each rangeline. The noise floor estimation for each

rangeline is achieved by applying a finite median filter to returns preceding the surface which comprise only noise since there are no detectable targets above the surface. A constant power offset (P_offset) is then added to this estimated noise floor to dynamically set the minimum surface bin power threshold for each rangeline, corresponding to the rising edge of the peak return.

These thresholds are further constrained using the surface index estimated from a digital elevation model (DEM) obtained from radar altimetry. Each rangeline's DEM surface bin estimate is derived by synchronizing DEM data with the radar's location and fast time sampling to get the corresponding radar two-way travel time and range bin index. The CA-CFAR threshold and the DEM estimate are combined to create a constrained search window from which the maximum return index is chosen as the rangeline's surface bin. Finally, a Savitzky-Golay filter is applied to the tracked surface bins from all the rangelines in the echogram to create a smoothed surface-bin 1-D contour vector.

4-1-2 Surface flattening

The tracked surface serves as the reference layer for subsequent layers that may exist in an echogram. After tracking the surface bin in each rangeline, these bins are aligned to create an echogram representation that mimics a perfectly flat surface. This improves visualization of internal layers, their correspondence to annual accumulation, and facilitates subsequent signal processing like along-track filtering. To align the surface bin, a rangeline (typically the first in the echogram matrix), is set as the reference and other rangelines or image columns are shifted up or down so that the fast time index of their surface bin matches the reference. The shift is implemented using linear interpolation. This operation is also reversible allowing the flattened echogram to be returned to its original state if necessary.

4-1-3 Averaging and filtering

The backscatter from a resolution cell of snow volumetric scattering at S and C-band is usually modeled as a random process [83] due to the snow grain size and its interaction with the radar signal wavelength. This stochasticity is seen in the high variance of the backscattered power for each rangeline which can obscure the snow internal layers, particularly for weak deeper layers. Prior to coherent and incoherent averaging, it is difficult or sometimes impossible to identify the layer peaks. To enhance the visibility of the layers and the performance of the layer tracker algorithms, a boxcar moving-average filter is applied in both fast-time and slow-time dimensions to smooth the returns. Filtering is done in the linear power domain. We chose low-order (order 3 and order 5) boxcar filters in along-track and fast-time dimensions, respectively, to improve the delineation of the internal layers since an aggressive filter can broaden the layer peaks and introduce subtle artifacts.

4-1-4 Detrending

One of the challenges associated with automated tracking of layers in echogram data is the inherent power loss experienced by the radar signal as it penetrates to deeper snow layers. As the transmitted radar signal propagates through the snow layers, the combination of spherical spreading loss, media attenuation, and scattering causes the signal power to be attenuated. Hence, the radar backscatter power from deeper layers is often lower than those from earlier layers. This results in a large dynamic range for each rangeline, which if not mitigated, can make it difficult for deep learning models to effectively detect and track deeper layers. The detrending step attempts to remove the power loss as a function of depth trend. By pre-processing the echogram data with detrending algorithms, we can create a more uniform signal profile down the entire rangeline. This pre-processing step allows the deep learning model to focus on the

variations in backscatter power that are more likely to be associated with the actual internal layers of interest, leading to improved tracking performance.

A low order polynomial fit to the log-power data is used to remove the trend in the received power. Given an echogram with a tracked surface and the deepest layer index for each rangeline, we identify three distinct regions (see Figure 4-2) in the backscatter data - (I) signal in air before the surface, (II) signal in snow/ice (with possible internal layers) and (III) signal after the deepest layer. Due to the distinct characteristics of each region, a different trend is estimated for each region as shown in Figure 4-2.

First, the trend for region (II) is found by fitting a low-order (5^{th} degree) polynomial (in a least-squares sense) to the backscatter in region (II). After this is found, the polynomial is evaluated at the surface bin which corresponds to the top of region (II) – this evaluation is used to set the detrend value for region (I). Since the signal prior to the surface is mostly dominated by the thermal noise of the system, the trend for region (I) is forced to be constant so that the noise is not amplified during detrending and the constant value is set equal to the polynomial evaluation at the surface bin to avoid a discontinuity here. Region III was not used in this work since the echograms were truncated to mostly not include the deeper regions and the low-order polynomial proved effective in fitting the noise region when present.

4-1-5 Normalization

Inputs to most machine learning models are usually normalized to either [0,1] or [-1,1]. This is done to achieve similar scales between the input images and the model's output to facilitate fast convergence during training. We apply a linear transformation to the original log-domain echogram image to map the signal portion of the echogram to a normalized scale of [0,1].

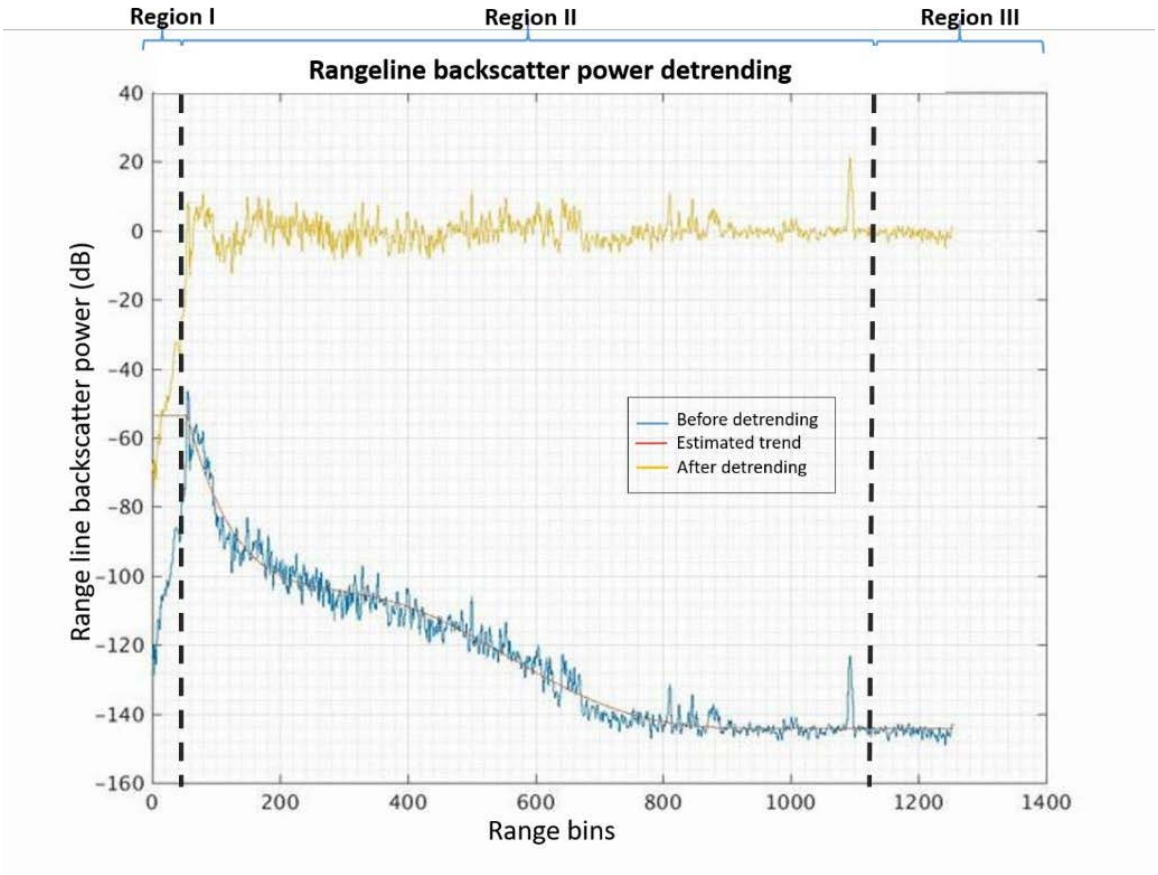


Figure 4-2: Plot of backscatter power of a rangeline showing detrending regions

4-1-6 Multi-distance multi-looking echogram blocks

To further diversify the dataset and maximize the available manually annotated echograms, a multi-distance multi-looking approach was implemented. Given that the data generally comes from long flightlines, the data can be segmented up in different ways with varying lengths and overlap. This strategy helps in constructing a larger and more diverse dataset. Consequently, we treat the coverage distance of each echogram as a flexible pre-training hyperparameter rather than maintaining a uniform distance across the dataset.

This method of using staggered flight line distances to form echogram blocks for training deep learning models has several advantages. It simulates a range of spatial accumulation patterns

within the dataset, which enhances the robustness of the models against "unseen" spatial distributions. By exposing the models to varied spatial accumulation patterns, we improve their ability to learn and accurately represent the true underlying snow layering phenomena. This approach significantly boosts the models' potential to generalize to new echograms that were not part of the training set, thereby increasing the likelihood of successful inference on echograms from future surveys. We also evaluate the performance of the models as a function of the along-track length.

To create the SnowRadar_Dataset_v1, echogram blocks of along-track distances of 2 km, 5 km, 10 km, 20 km and 50 km were created. The training, validation and test sets all include a mixture of these different lengths of images. However, to ensure uniformity of dimensions of all the echograms in the training set, along-track averaging (multi-looking), and linear interpolation were used to resample the data to create fixed size images of $N_t = 1664$ and $N_x = 256$.

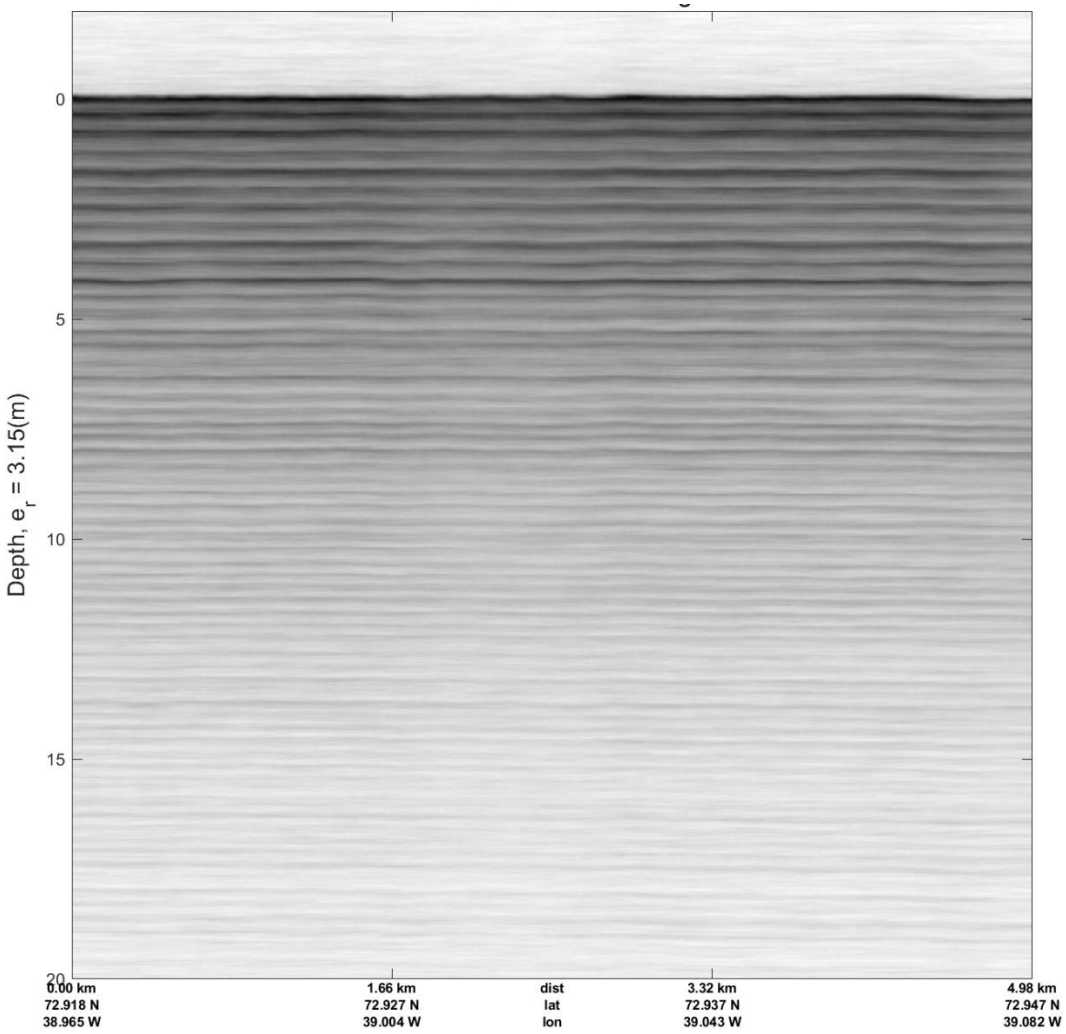


Figure 4-3: Image of echogram after applying steps 1- 5

The dimensional uniformity is crucial because it standardizes the dataset samples, enabling an accurate assessment of the performance of different deep learning models. By ensuring that each echogram block adheres to a consistent format, we eliminate variability that could otherwise skew the evaluation of model performance, thus allowing for a more reliable comparison across different architectures and training methodologies.

To facilitate easy distinction and identification, the specific distance associated with each echogram block is appended to its filename. This practice not only aids in organizing the dataset

but also allows researchers to quickly reference and analyze the spatial coverage of each echogram block, contributing to more efficient data management and model training processes.

Figure 4-4 compares 2 km and 5 km echogram blocks created from approximately the same geographical location. Although the echogram images appear visually similar due to their overlapping geolocation, a closer examination reveals differences in the along-track spatial accumulation patterns. These variations are encoded in the echogram layer variance, which is significantly different for the two images.

During training, the models are exposed to this subtle yet critical information, allowing them to better identify and track accumulation layers. This nuanced variance serves as a valuable source of data, enhancing the model's ability to accurately learn and predict snow layering patterns across different spatial scales.

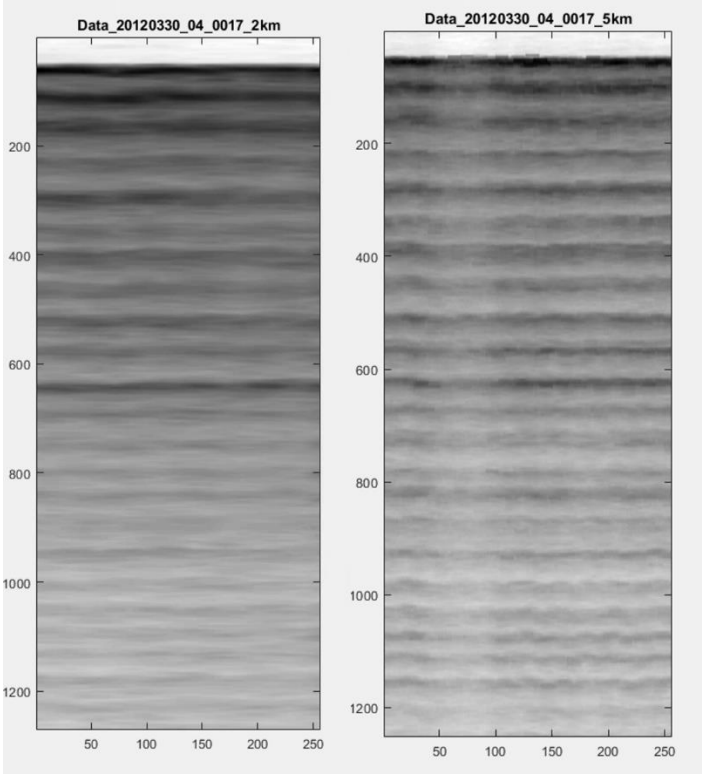


Figure 4-4: Echogram blocks created from 2km and 5km line distances respectively

4-2 Snow Radar ML_Dataset_v1

4-2-1 Dataset echograms

The Snow Radar Dataset_v1 is created from selected flight lines from the NASA Operation Ice Bridge (OIB) campaigns in 2012. The dataset comprises 11 flight lines, covering a total distance of 28,369 line kilometers. While the primary objective was to create a dataset that captures the rapidly varying spatiotemporal accumulation patterns across the ice sheet, the majority of the available annotated data originates from the dry snow zone.

The training and validation set uses 9 of the 11 flight lines spanning the different snow zones and accumulation patterns across Greenland. Consequently, only a small portion of the data is from other zones such as the wet snow zone and transition zones.

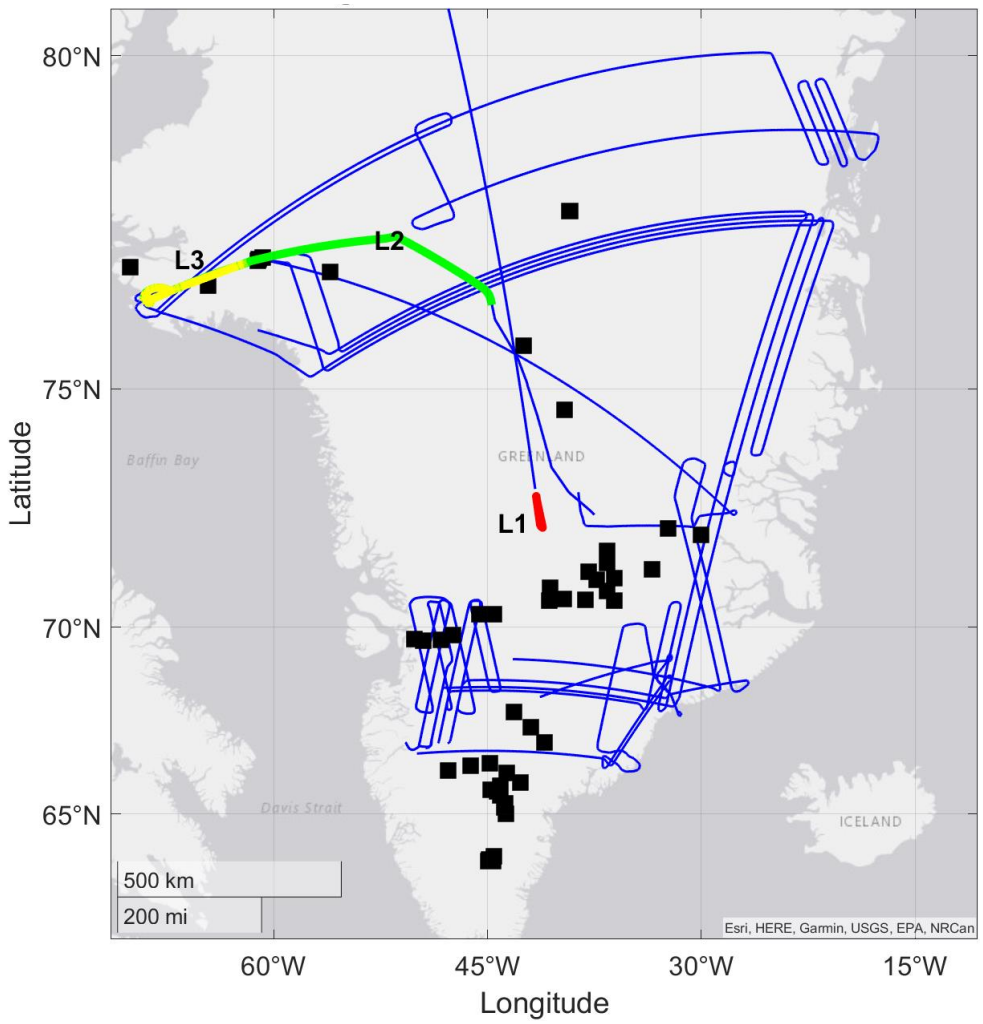


Figure 4-5: *Spatial plot of dataset flight lines and neighboring ice cores. Flight lines in blue represent training data while those in red (L1), green (L2), and yellow (L3) are the test data. Black squares mark the locations of some of the existing ice cores and snow pits in Greenland*

The test set is carefully divided into 3 groups; L1, L2, and L3 each represent different snow accumulation patterns that exist in most polar ice sheets. The L1 test set is derived from the dry snow zone characterized by well-preserved annual accumulation stratigraphy. L3, on the other hand, is close to the coast and contains echograms from the wet snow zone where older stratigraphy might have been eroded due to melting. The L2 test echograms capture the transition between both zones.

The quality of the echograms varies across these zones. Echograms from the L1 section of the test data are of the highest quality, exhibiting clear visibility of snow layers. Conversely, the radar backscatter from L2 and L3 zones is often diffused due to the presence of refrozen melted snow. This reduction in visual quality is important for assessing the performance of deep-learning models on echograms from different snow zones with varying image quality. The split of the test data into these 3 sections is done to investigate how deep-learning models perform on echograms from different snow zones and of varying image quality.

Additionally, there are some test echograms (in L2 and L3) that coincide closely with existing ice cores which can be used for corroborating ice-depth-age measurements for radar data facilitating synchronization between radar data and ice core data. This alignment is intended to encourage further studies comparing radar measurements with coinciding and neighboring ice core and ice pit measurements.

Summarily, the Snow Radar dataset_v1 has 11,302 training set echograms, 1,322 validation echograms and a total of 1,292 test echograms. The test echograms are further subdivided into 127 L1 echograms, 1049 L2 echograms and 116 L3 test echograms.

Since L3 test echograms primarily originate from the wet snow zone, some images capture no internal snow layers or exhibit very low quality. Despite their poor quality, they are important for model training because most flight lines include at least one of these echograms. The objective is for deep learning algorithms to learn to correctly identify these low-quality echogram images and either ignore them or track only the surface layer when possible. The overall distribution of the dataset is shown in Figure 4-6

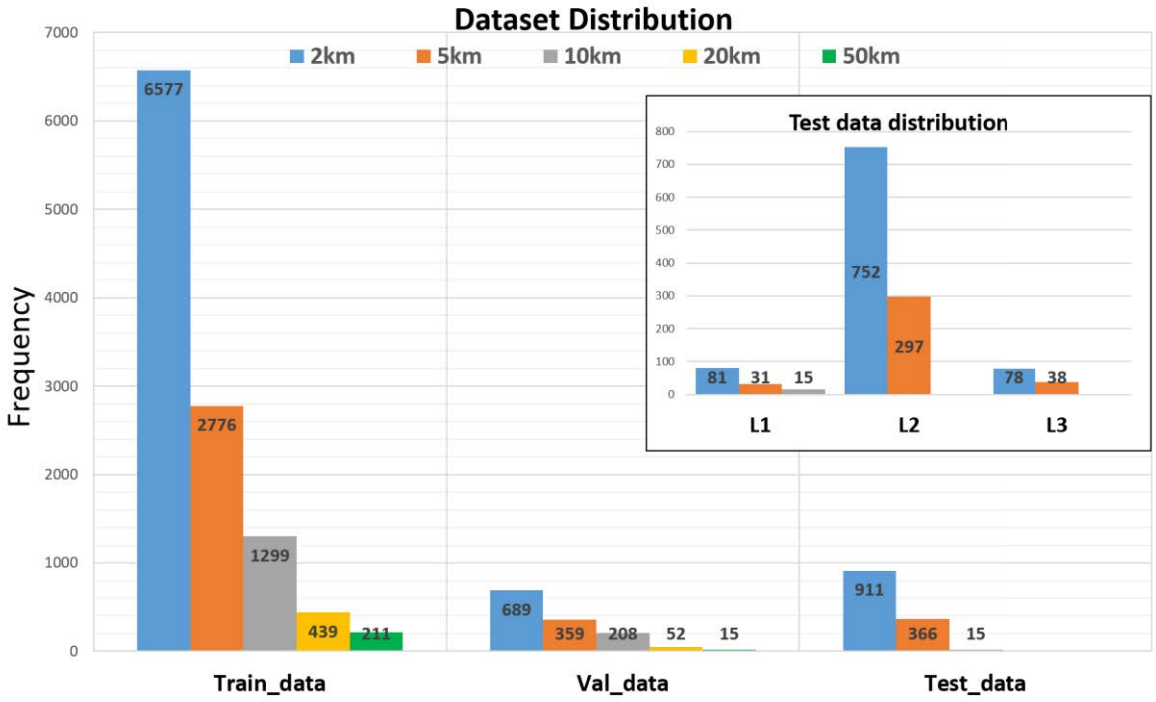


Figure 4-6: *Distribution of the dataset along-track distances in the train, validation, and testing sets. The testing set subdivision into 3 levels L1, L2, and L3 corresponding to different Snow Zones are also shown in the insert plot. The numbers in each bar represent the frequency of occurrence of each along-track distance in each of the set*

4-2-2 Dataset metadata

In creating the Snow Radar Dataset_v1, we included metadata for each echogram. The Snow Radar is equipped with Inertial Measurement Unit (IMU) and Global Positioning System (GPS) to provide precise auxiliary information about the surveyed location and aircraft's orientation at every point during the data collection. These additional meta-data, such as the antenna phase center geolocation, aircraft elevation, and other GPS data have been synchronized with the radar echogram to provide additional information for each rangeline. This provides additional context and description to further ground the understanding of the users what each echogram represents and provides context for accurate interpretation. Moreso, there are new deep learning paradigms where relevant metadata are included in training to achieve performance gains.

Each entry in the dataset is provided as a .PNG file and has an associated .MAT file. Some of the important fields in the files are described below. The data fields are described first followed by metadata fields.

- Data [$N_t \times N_x$]: This is the echogram image and is the primary training input. It is the log-normalized backscatter received from the surface and other internal layers at the imaged location.
- Layers [$N_L \times N_x$]: This is a form of the ground truth created through a combination of semi-automated and manually corrected annotation. It is provided as a 2D matrix that is a stack of N_L rows of 1D contour vector representing each tracked layer. Regardless of the deep learning architecture or tracking method, this is the final and preferred format for saving the tracked snow layers. In this form, the accumulation between successive annual layers can be estimated. The values are oftentimes in the range bin domain depicting the fast time bin the layer is localized in each rangeline. However, this can be converted to radar two-way travel time using the ADC fast time sampling rate (also provided in the metadata) which is eventually used to estimate the layer depth or accumulation in meters.

- Layers bitmap or raster [$N_t \times N_x$]: This is another form of the ground truth annotation containing the same information as the "Layers" field. It is, however, provided in a binary raster format. This format is typically used as the training "target" for a binary segmentation task. It is a sparse binary matrix of equal dimensions with the echogram image, but its pixel only contains one (1) when there is a layer and zero (0) otherwise. This form of the ground truth has an imbalanced class distribution with many more zeros than ones which might need to be accounted for during model training.

- Layer segment $[N_t \times N_x]$: This is a dense matrix and is another form of ground truth. It has the same dimension as the echogram image and is created as the label for training deep multi-tiered segmentation models. In this 2D matrix, each accumulation layer is assigned a unique index starting from the topmost layer as 0 and the index is repeated until the next accumulation layer.

Other important GPS and IMU metadata

- Latitude $[N_x \times 1]$: This is the associated WGS-84 geodetic latitude coordinate reported by the GPS and IMU and is synchronized with each rangeline of the echogram. The position fields can be used to synchronize radar measurements with other sources such as LIDAR, satellite image, weather models, etc.
- Longitude $[N_x \times 1]$: This is the corresponding pair of Latitude.
- Elevation $[N_x \times 1]$: This is the elevation relative to the WGS-84 ellipsoid surface.
- Time $[N_x \times 1]$: This is the radar system Analog-to-Digital Converter (ADC) fast time axis corresponding to each row of the image. Zero time corresponds to the time the transmit signal was sent. This field is important for converting tracked echogram layers in range bins (or row-index) back to radar two-way-travel time.
- Roll $[N_x \times 1]$: This measures the rotation of the aircraft about its longitudinal axis. Positive roll corresponds to ring wing tip down. It is provided in radians and has a relationship with the backscatter received because the radiation pattern of the antenna rotates with the aircraft since the antenna is fixed (i.e. not mounted on a gimbal). Non-zero roll angles mean that the antenna's boresight or mainlobe points away from nadir resulting in a reduction in received power.

- GPS Time $[N_x \times 1]$: This is the GPS time corresponding to when each rangeline's data was collected. It is saved in the ANSI C standard (seconds since Jan 1, 1970 00:00:00).

4-3 Synchronization of dataset with Model Atmospheric Regional (MAR) weather model data

Regional weather models, such as the Modelle Atmospherique Regionale (MAR), provide historical meteorological and climatic data that can complement radar measurements to enhance the accuracy of annual snow accumulation estimates in Greenland. To further enrich the Snow Radar dataset, MAR model data is synchronized with radar measurements, providing auxiliary information that deep-learning models can potentially leverage to improve layer detection and tracking performance. This integration also opens avenues for comparative studies between radar imagery and weather model outputs.

In this work, we utilize MAR model data version 3.10, which provides climate data at a 15 km grid resolution across Greenland. The synchronization of MAR outputs (e.g., density, temperature, etc.) with radar data is achieved using the latitude and longitude coordinates of each rangeline in the echograms. A 2D Delaunay triangulation interpolation algorithm is employed to align the gridded MAR outputs with the radar data.

Surface Mass Balance (SMB) is selected as a primary measurement to be synchronized with radar data, as it provides estimates of annual accumulation layers that correspond to the internal layers observed in echogram images. The annual SMB is computed by summing daily measurements from the weather model for each accumulation cycle, covering the past three

decades up to the year of the radar measurement. The MAR daily SMB is calculated as the sum of daily snowfall and rainfall, minus sublimation, evaporation, runoff meltwater, and surface water. The accumulation cycle used spans from September of one year to September of the following year, reflecting the snow layer captured by the Snow Radar during the summer-to-winter transition. In cases where summer melt exceeds winter accumulation, resulting in a negative annual SMB, the deficit is subtracted from the previous year's mass balance.

We estimate annual accumulation from the weather model and convert it into equivalent Snow Radar internal layers, creating a model-derived product equivalent to the annotated ground truth. To achieve this, the SMB is accumulated from the date of acquisition to the date of each summer-to-winter transition. These net accumulations, measured in millimeters of water equivalent (mmWe), are then converted to snow layer depth in meters. By combining this information with a depth-density profile and a density-permittivity conversion model, an estimate of the radar's two-way travel time to each weather model-estimated annual layer can be derived.

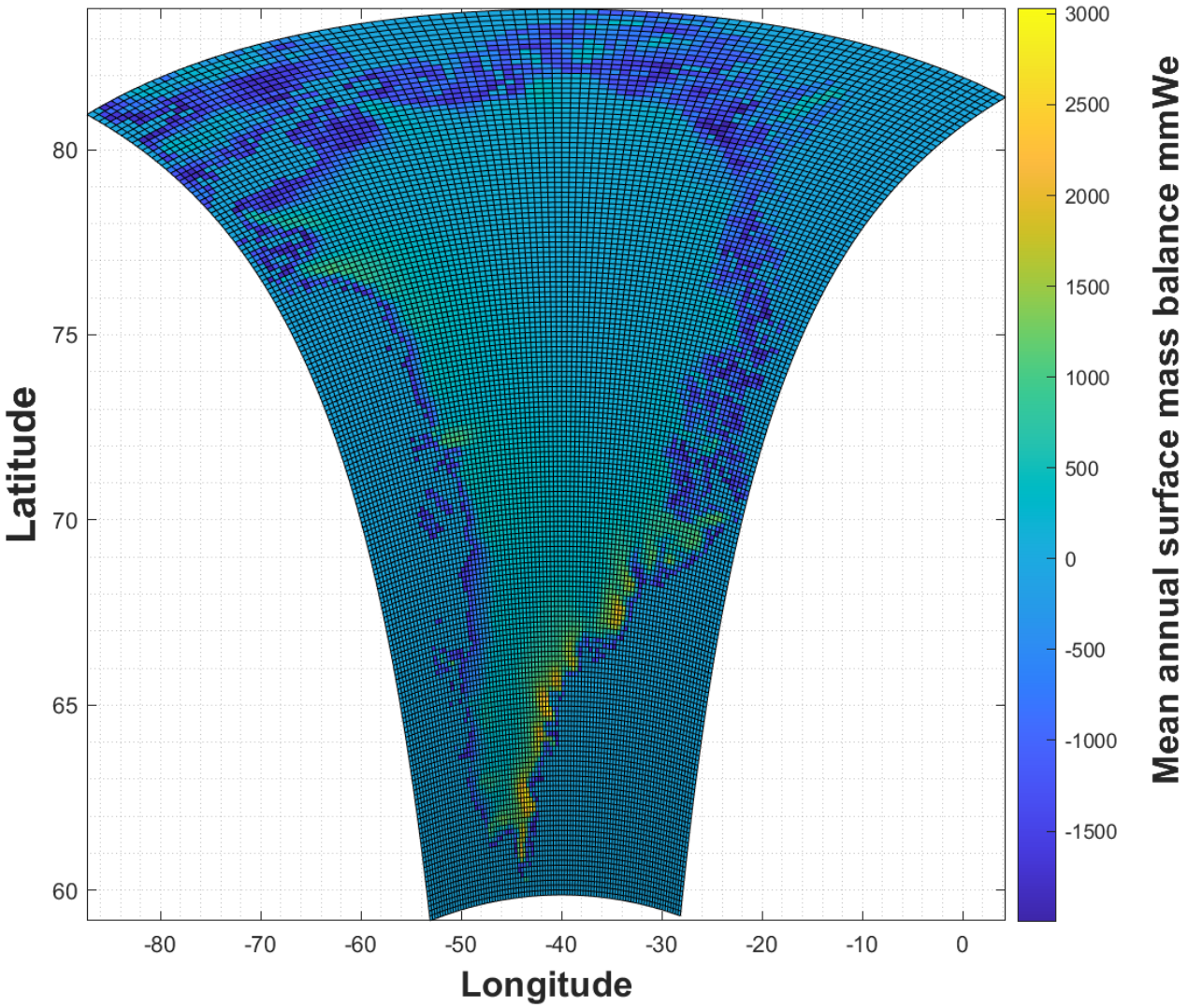


Figure 4-7: *MAR map of mean annual surface mass balance for the Greenland ice sheet*

To estimate how the initial snow density varies with snow depth for each flightline, the mean annual accumulation, initial snow density, and mean annual surface temperature from MAR data are used as inputs to the Herron and Langway firn densification model [84]. The output of this is a density versus depth profile (ρ_d) for the imaged location. The derived density-depth profile is subsequently used as input to the simple mixture model in Equation (11) to estimate the snow permittivity vs snow depth profile.

$$\epsilon_d = (1 + 0.85{\rho_d}^2) \tag{11}$$

$$TWTT = (2 \times d \times \sqrt{\epsilon_d}) / c$$

Finally, the estimated two-way travel time is synchronized with the radar imagery layering using the radar ADC fast-time sampling rate. This estimate can be considered as equivalent to the annotated echogram layer vector ground truth, albeit, from the MAR model's perspective. It is provided as "weather layers" metadata associated with each echogram. However, due to the coarse spatial resolution of the MAR data, the MAR weather layers fail to capture fine along track layer details that are on the order of tens to hundreds of meters. Like the annual weather data layer estimated from SMB, other measurements from MAR were interpolated onto the radar measurement flight lines to create annual measurements for echograms in the datasets. These include annual meltwater production (ME), mean surface temperature, mean surface density, run-off of meltwater and rainwater (RU), meltwater refreezing and deposition (RZ), Snow height change due to melt (SHC), Snowpack height total (SNHS), etc.

Each row of these fields in the echogram data corresponds to annual MAR output synchronized to the radar rangelines starting from the date of radar data collection back to the summer-to-winter transition of each earlier year in the dataset (in this case 2012 back to 1980) making a total of 32 annual measurements. For example, the first row in the "curr_smb" field is the estimated SMB from the MAR output for the year 2012 while the 5th row is the estimation for the year 2008. It is important to note that while the number of snow layers seen by the radar (i.e. the number of layers in the echogram) might vary from one echogram to the other depending on

the snow zone the echogram measurement was taken from, the number of MAR “layers" estimated from MAR data remains the same for all the echograms in the dataset.

More details about the MAR model and its outputs can be found in [85].

4-4 - Full echogram image layer tracking

With the relative increase in the training dataset, we set out to train and predict on the entire echogram at one go as opposed to iteratively tracking the layers one at a time. However, it is important to correctly define and formulate the problem within the appropriate deep learning framework.

Many of the state-of-art performance reported in computer vision tasks are from classification models. Deep learning classification models are those trained to identify a single class for each input image. They are a type of supervised learning algorithm used to categorize data points into predefined classes by learning a mapping from input features to discrete output labels, essentially building a decision boundary between the different classes it was exposed to during training. When probed with an input image during inference, the objective is to identify the correct class of the input from the set of inputs it was trained on. As an example, if a model is trained on a set containing different types of cat and dog images, at inference, the model is supplied a new cat image, one not used for training, the performance of the model is evaluated based on how well it is able to correctly classify the new image. These classification models do not output the pixels associated with each object but just the class of the object in the test image. This paradigm does not fit the echogram layer tracking problem which needs to identify the pixels associated with each layer.

The goal of the echogram layer tracking problem is to design a deep vision algorithm that can identify each snow accumulation layer in the echogram as a 1-dimensional contour which describes which image pixels the layer passes through and has the same cardinality as the number of rangelines in the echogram image. The final output will therefore be a 2D matrix of all the identified 1D contours vectors corresponding to the orientation of all the snow layers in the echogram image.

Formally, given an input 2D grayscale echogram image $\mathbf{E} \in \{\mathbb{R}^{N_t \times N_x} : 0 \leq E(m,n) < 1\}$, $\mathbf{E}$ represents the two-dimensional spatial distribution of the firn layer backscatter in the along-track or slow time (N_x) axis and depth layer or fast time (N_t) axis. A deep learning model is to identify which of the 2D echogram matrix $\mathbf{E}$ pixels contain a snow layer and track at most N_x consecutive columns for each layer. The output of the algorithm for L unique layers would then be $\mathbf{O} \in \mathbb{R}^{N_L \times N_x}$

In the context of supervised deep learning, we cast the echogram layer tracking problem in two different ways:

(i) binary segmentation and

(ii) deep-tiered multi-layer segmentation problem.

4-4-1 Binary image segmentation

In the binary segmentation paradigm, the model is trained to classify each pixel in the image as either containing a layer (1) or not (0) using the associated ground truth binary matrix annotation

of identical dimension: $G_b \in \{\mathbb{R}^{N_t \times N_x} : G_b(m,n) = \{0,1\}\}$ (see Figure 4-8b). It is important to note that in binary image segmentation, only one pixel is designated as the layer pixel for each layer in each rangeline which differentiates it from traditional segmentation tasks that generally do not explicitly constrain the shape or number of pixels associated with any given class.

While there can be a variety of inner architectures for the model, the output layer of the binary segmentation neural network generally has a sigmoid activation function for each pixel that is subsequently thresholded to produce binary outputs. This thresholding step is critical, as it reveals the sensitivity and specificity of the model in correctly identifying layer pixels, directly impacting the accuracy of layer tracking. During inference, the trained model is given a test grayscale echogram image **E**, to output a binary matrix classifying each input pixel into one of "layer" or "no-layer" classes depending on whether or not the pixel contains a layer. Since accumulation layers in the echogram image have spatial correlation along-track, correctly identified pixel layers in adjacent columns naturally trace out the layer geometry. This inherent spatial coherence allows the model to effectively capture the underlying structure of the snow layers. This approach is simple, relatively easy to train, but powerful and generalizable because it makes less assumptions about the task.

4-4-2 Deep-tiered multi-layer segmentation:

Given that each layer in the echogram image corresponds to a chronological snow deposition, typically assumed to be annual, the multi-layer segmentation task aims to uniquely identify the pixels associated with each year's deposition. The most recent accumulation prior to the data collection is seen as the first layer, with older years arranged sequentially in chronological order. This structure informs the ground truth annotation as depicted in Figure 4-8 where each snow layer

is assigned a unique class label. Hence, the deep learning model is trained to identify pixels corresponding to each year's accumulation and to delineate boundaries between adjacent layers.

Concretely, the ground truth annotation is a 2D matrix $G_d \in \{\mathbb{R}^{N_t \times N_x} : G_d(m,n) \in \{0,1,2,L_{max}\}\}$. Each pixel is assigned a tiered label L based on its associated year of deposition:

- $L = 0$ represents the pixels in the signal-in-air portion before the transmit signal interacts with the surface,
- $L = 1$ represents the first year's accumulation,
- $L = 2$ represents the second year's accumulation,
- And so on until $L = L_{max}$ which corresponds to the deepest accumulation layer in the input echogram image.

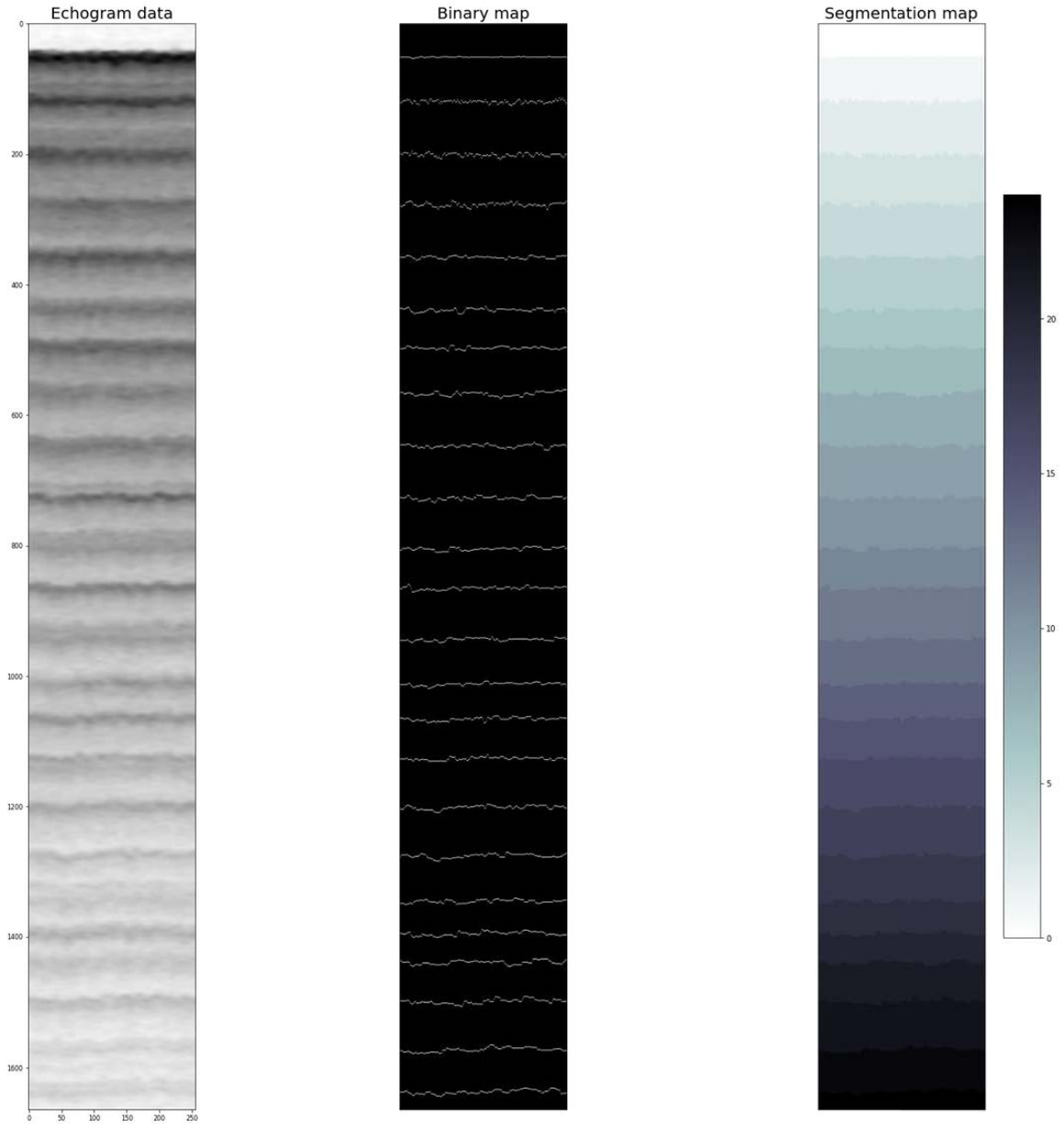


Figure 4-8: Example image echogram and the corresponding binary segmentation and multi-class segmentation labels

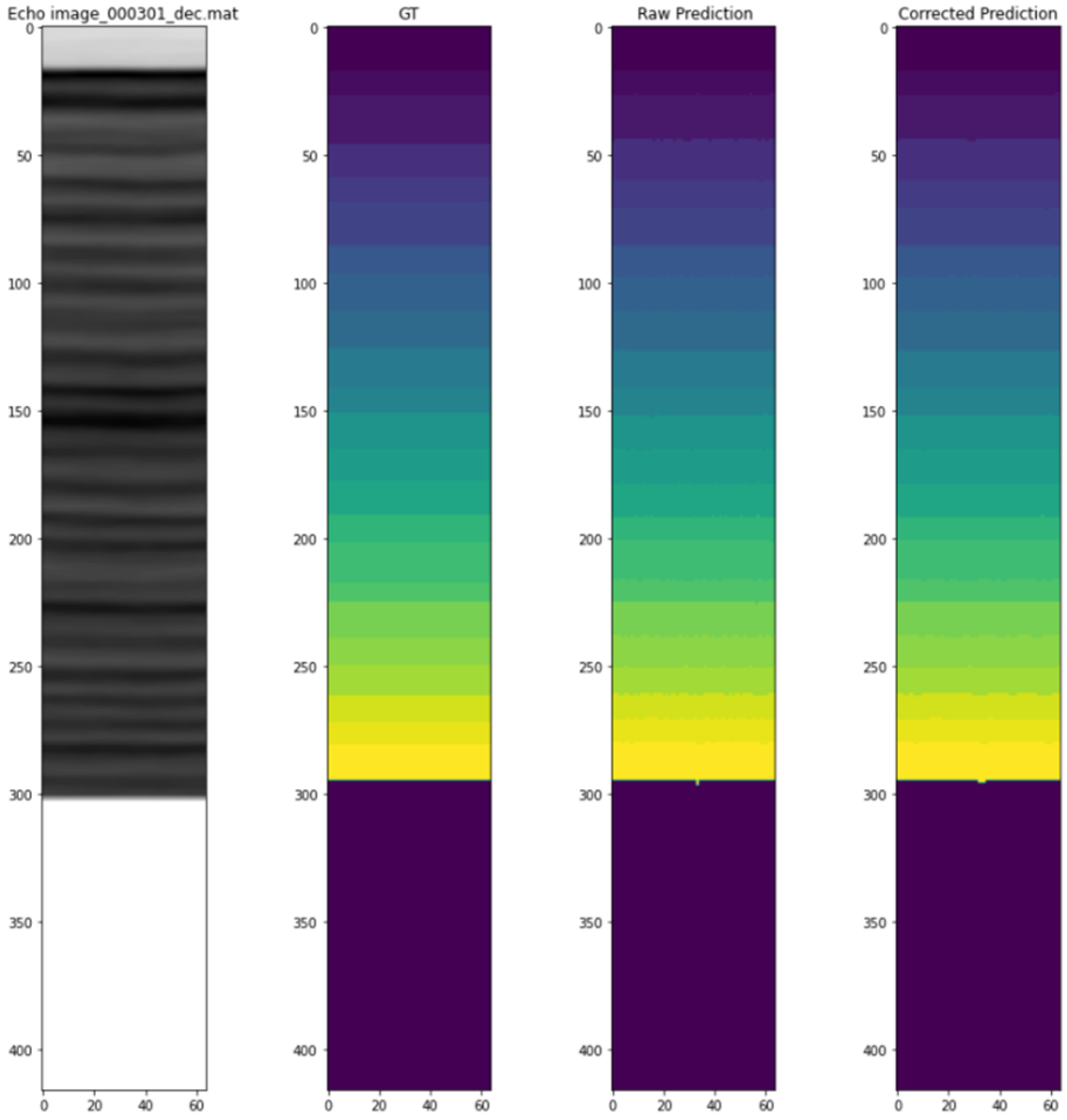


Figure 4-9: *Example qualitative result of multi-layer transformer-based architecture*

This approach resembles multi-class image segmentation of optical images, with the distinction that, in deep-tiered multi-layer segmentation, accumulation layers are naturally ordered by the year of the snowfall. Also, layers share horizontal boundaries only with adjacent accumulation layers (the year before and after) but with no other layer. This approach to the radar echogram layer

tracking problem has the advantage of directly estimating each year's annual accumulation range bins since it uniquely identifies each layer pixel in the along-track axis and delineates its boundaries with adjacent accumulation layers.

However, this approach comes at the cost of increased parameterization of the model and the inherent need for more training data, and consequent longer training time.

For the preliminary stages of this work, we truncated the echograms to only contain the top 30 layers with the intention to extend this to include all deeper snow layers in the echogram after the initial analysis. Figure 4-8 shows a sample echogram image, the binary mask, and the multi-layer segmentation mask.

4-5 Deep learning models for full echogram image tracking

Detecting and tracking snow layers in echogram images can be conceptualized as a semantic segmentation task where the model is designed to classify each pixel in the input image instead of just assigning a label to the entire image as in image classification. To provide the desired pixel-wise dense prediction output, a fully convolutional output layer is pivotal in the design of semantic segmentation models. The fully convolutional layer uses a 1x1 filter kernel on the penultimate convolutional filter outputs to produce dense output prediction. This paradigm is employed in most well-known semantic segmentation architectures.

To benchmark the performance of the Snow Radar Dataset_v1 dataset, we trained the following well-known segmentation models with slight adaptations to their original architecture for our particular problem.

1. Fully Convolutional Networks (FCN) [86],
2. U-Net [82],
3. Attention-U-Net [87],
4. DeepLabv3+ [88],
5. and a soft ensemble of all four models.

We train these models for the dense binary segmentation tasks, i.e. to predict for every pixel in the echogram image if it contains a layer or not. A brief description of each of the models is provided below.

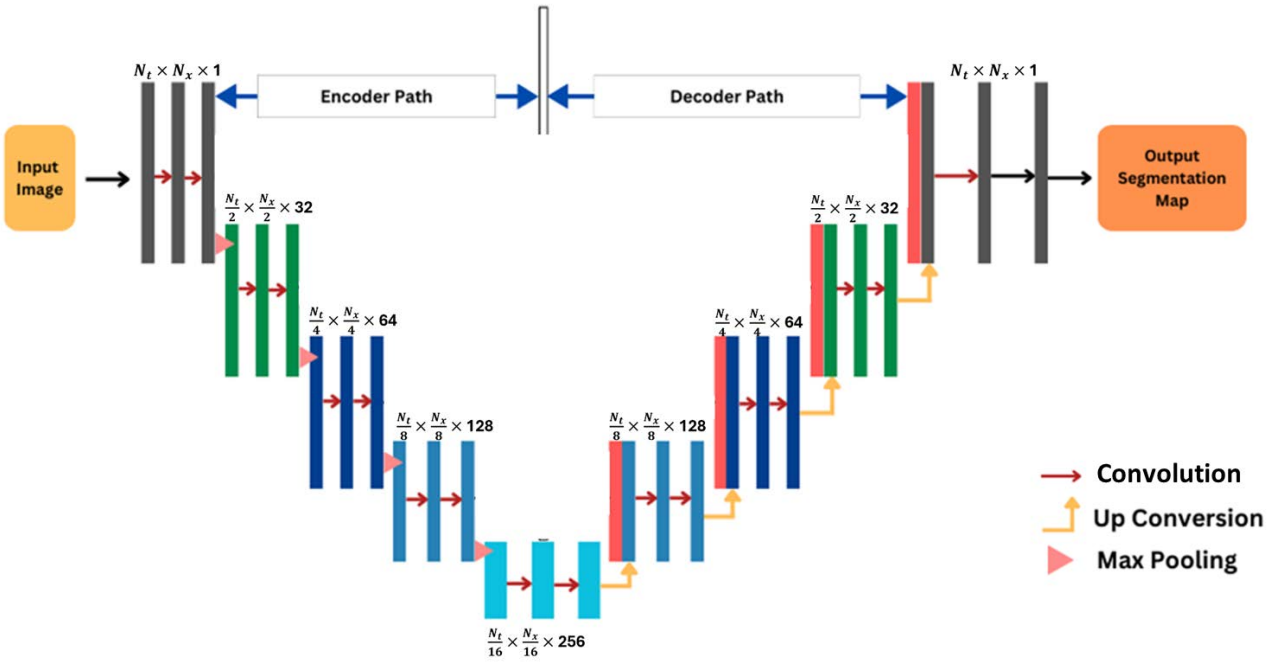
4-5-1 Models' description

1. **Fully Convolutional Network:** This model is generally regarded as one of the pioneer deep learning architectures for binary and multi-class image segmentation and was proposed in [86]. The work represents a significant advancement in the field of computer vision, particularly for the task of semantic segmentation. Traditional convolutional neural networks (CNNs) end with fully connected layers which produce a small, fixed number of outputs, unsuitable for pixel-wise predictions required in segmentation tasks. FCNs address this by replacing these fully connected layers with convolutional layers that maintain spatial information, allowing the network to output a spatially dense prediction map that corresponds directly to the input image dimensions. This architecture effectively transforms a classification network into a dense, pixel-wise prediction model.

   The FCN architecture starts with an encoder-decoder structure where the encoder part consists of convolutional and pooling layers. The encoder stage is made from a sequence of 2D convolutional layers applied to the input image with intermediate spatial

downsampling. After each stage of 2D convolution, the number of applied filters is increased while the spatial dimension is reduced to increase the representation power of the model while progressively learning local-to-global features in the input image.

For echogram layer tracking, a key part of the original architecture was altered in our implementation. Instead of upsampling directly from the bottle-neck layer in the encoder to the decoder output, the transposed convolution was done in equal amounts of stages as it was in the encoder. This approach helps preserve low-level information learned by the network, such as curves and lines, which are crucial for the echogram layer tracking task. However, no skip connection was inserted between the corresponding encoder layer. Finally, the decoder fully-convolutional layer is used as the output layer to produce pixel-wise segmentation prediction.



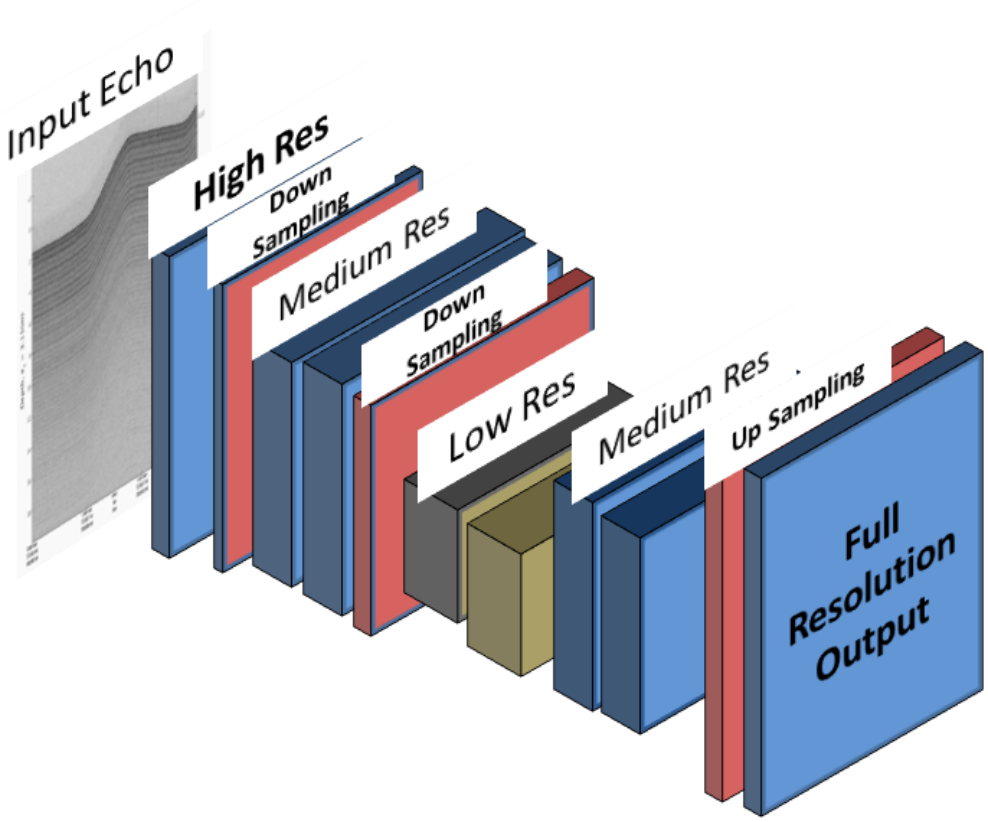


Figure 4-10: *Schematic diagram of the Fully Convolutional Network (FCN) architecture*

2. **U-Net model** – The U-Net model remains one of the classic segmentation models till date. Prior to this work, traditional approaches to segmentation often struggle with limited training data and the challenges posed by complex structures and variations in niche domain images. Although the original U-Net architecture was specifically tailored for biomedical image segmentation tasks, the solution proffered to these issues by employing an encoder-decoder structure with skip connections addresses the underlying challenge for a number of related fields.

   The U-Net architecture consists of a contracting path, which captures context and reduces spatial dimensions, and an expansive path, which enables precise localization. The

contracting path comprises a series of convolutional and max-pooling layers that progressively downsample the input image, extracting high-level features. The expansive path consists of upsampling layers followed by convolutional layers that increase the spatial resolution of feature maps. Crucially, skip connections are introduced to connect corresponding layers between the contracting and expansive paths, allowing the network to retain fine-grained spatial information from earlier stages while incorporating high-level context.

This unique architecture enables the U-Net to effectively learn intricate spatial relationships within input images and produce accurate segmentation masks. The original authors demonstrated the efficacy of U-Net on various biomedical segmentation tasks, including cell tracking in microscopy images and delineation of neuronal structures in electron microscopy data. U-Net's ability to leverage both local information and global context, facilitated by skip connections, proved instrumental in achieving state-of-the-art performance in biomedical image segmentation then.

For the binary segmentation task for echogram layer tracking, we designed four stages of the encoder-decoder with double convolution at each stage. Each block has a sandwich of convolution, max-pooling, batch normalization and ReLU activation function. A progressive number of filter channels of size 32, 64, 128 and 256 was applied at each stage with skip connections between feature maps of similar spatial resolution.

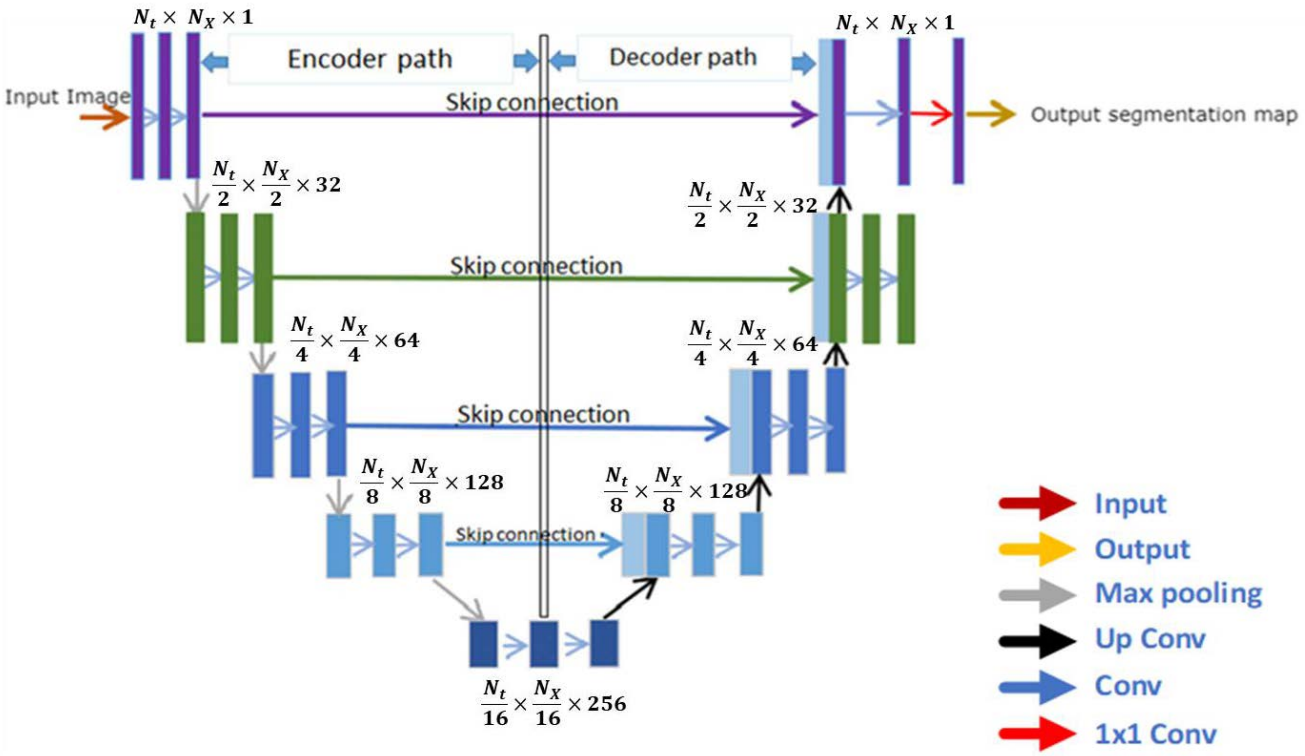


Figure 4-11: *U-Net segmentation architecture showing each block and the skip connections*

3. **Attention U-Net:** The Attention U-Net, as introduced in the paper "Attention U-Net: Learning Where to Look for the Pancreas," [87] represents a significant advancement in medical image segmentation, particularly for identifying the pancreas in abdominal CT scans. Traditional segmentation approaches often struggle with accurately delineating small and intricate structures like the pancreas due to its variability in shape, size, and appearance. The Attention U-Net addresses this challenge by incorporating an attention mechanism into the U-Net architecture, allowing the network to dynamically focus on relevant regions while suppressing irrelevant background information. This ability is critical in echogram layer tracking.

The Attention U-Net architecture extends the standard U-Net by integrating an attention gate module into its contracting path. This attention gate module learns to assign different weights to feature maps based on their importance, directing the network's focus towards informative regions. During training, the attention mechanism enables the model to learn where to concentrate its attention for accurate important local features necessary for segmentation. Importantly, the attention gate module is designed to be lightweight, ensuring computational efficiency while enhancing the network's ability to capture fine-grained details crucial for pancreas localization.

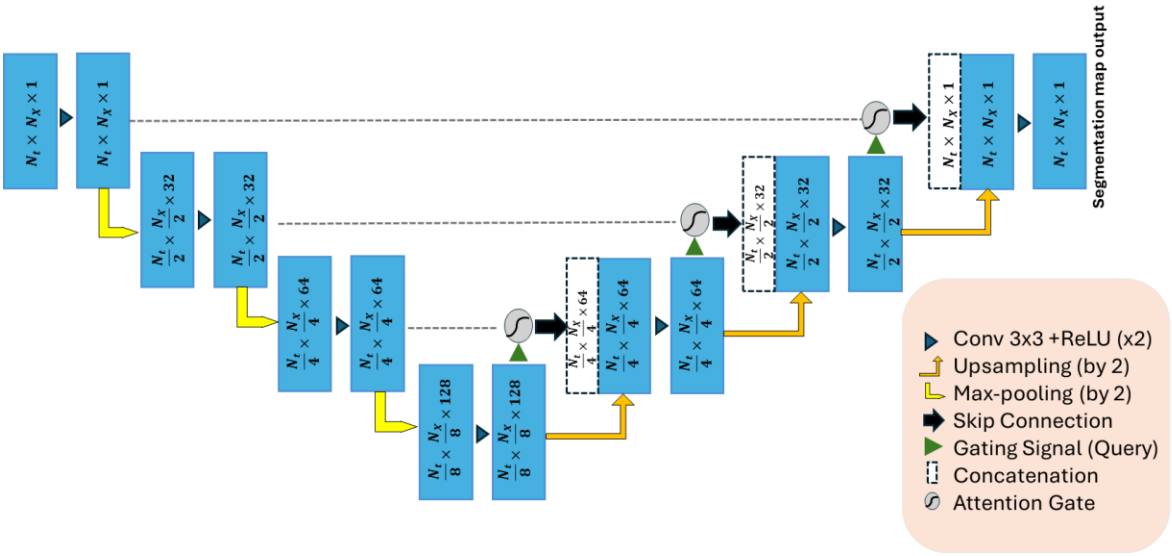


Figure 4-12: *AttentionU-Net architecture showing the insertion of attention gates into the U-Net architecture*

The attention mechanism computes an attention map that weighs the importance of each pixel or region in the feature maps. This map is used to scale the feature maps, emphasizing important areas and diminishing less important ones. They are inserted into the skip connections between the encoder and decoder paths in the U-Net architecture. Each attention gate receives two inputs: the feature map from the encoder (which is being passed

to the decoder via the skip connection) and the feature map from the corresponding decoder layer. The gate computes an attention map that modulates the encoder feature map, allowing only the most relevant features to be passed to the decoder. The attention gate uses additive attention, where the importance of each spatial location is computed based on the combination of the encoder and decoder features. The gate outputs a map that highlights areas of interest, which is then multiplied by the encoder feature map before it is passed to the decoder. By dynamically attending to relevant regions, the Attention U-Net is meant to achieve superior performance compared to traditional U-Net models and other similar segmentation methods. The attention mechanism allows the network to adaptively adjust its focus, improving segmentation accuracy and robustness across diverse patient datasets.

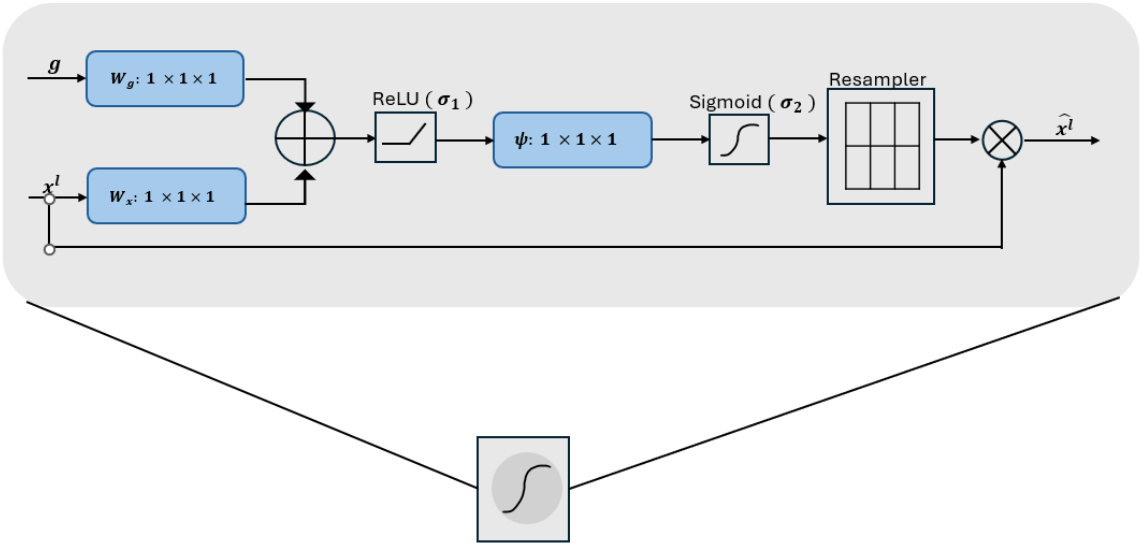


Figure 4-13: Magnified view of the additive attention operation

For our implementation, instead of using a squeeze attention module along the channel axis, attention gates were designed to filter the features by suppressing what the network

deems as irrelevant regions and highlighting salient features that are considered critical to the task at hand.

4. **DeepLab v3**: The key innovation of DeepLab lies in its employment of atrous convolution, also known as dilated convolution, which enables capturing multi-scale contextual information without significantly increasing computational complexity. By applying atrous convolution at multiple dilation rates, DeepLab effectively enlarges the receptive field of convolutional filters, allowing the model to integrate contextual information across different scales while preserving spatial resolution.

   Furthermore, DeepLab incorporates the use of atrous spatial pyramid pooling (ASPP), which further enhances its ability to capture multi-scale contextual information. The atrous convolutional layer, also known as dilated convolution, uses a dilation rate to control the spacing between kernel elements, effectively expanding the receptive field without increasing the number of parameters or reducing the spatial resolution. When the dilation rate is set to values greater than 1, the convolution operation skips over input pixels, allowing the network to capture features from a larger context, similar to downsampling. This is because, with a higher dilation rate, the convolution can aggregate information from a wider range of pixels while maintaining the same input size. As a result, the model can perceive a larger area of the input image (or feature map) without explicitly reducing its spatial dimensions, making atrous convolution a powerful alternative to downsampling, especially in tasks like this where preserving spatial resolution is crucial.

   ASPP utilizes multiple parallel atrous convolutional layers with different dilation rates to capture context at various scales. This allows DeepLab to efficiently integrate information

from different receptive fields, enabling accurate segmentation of objects at different sizes. Additionally, DeepLab employs a fully convolutional network (FCN) architecture, facilitating end-to-end training and enabling the model to produce pixel-wise segmentation masks directly from input images.

For our implementation to track echogram layers, ResNet50 was used as the backbone of the encoder taking outputs from two resolution scales and combining them as the input to the decoder. This is further upsampled (decoded) to extract features at the image scale. Given the encoder-decoder network with spatial pyramid pooling in the DeepLab architecture, it can extract contextual information from multiple scales, as well as refine them through sharp object boundaries which is crucial for the layer tracking task.

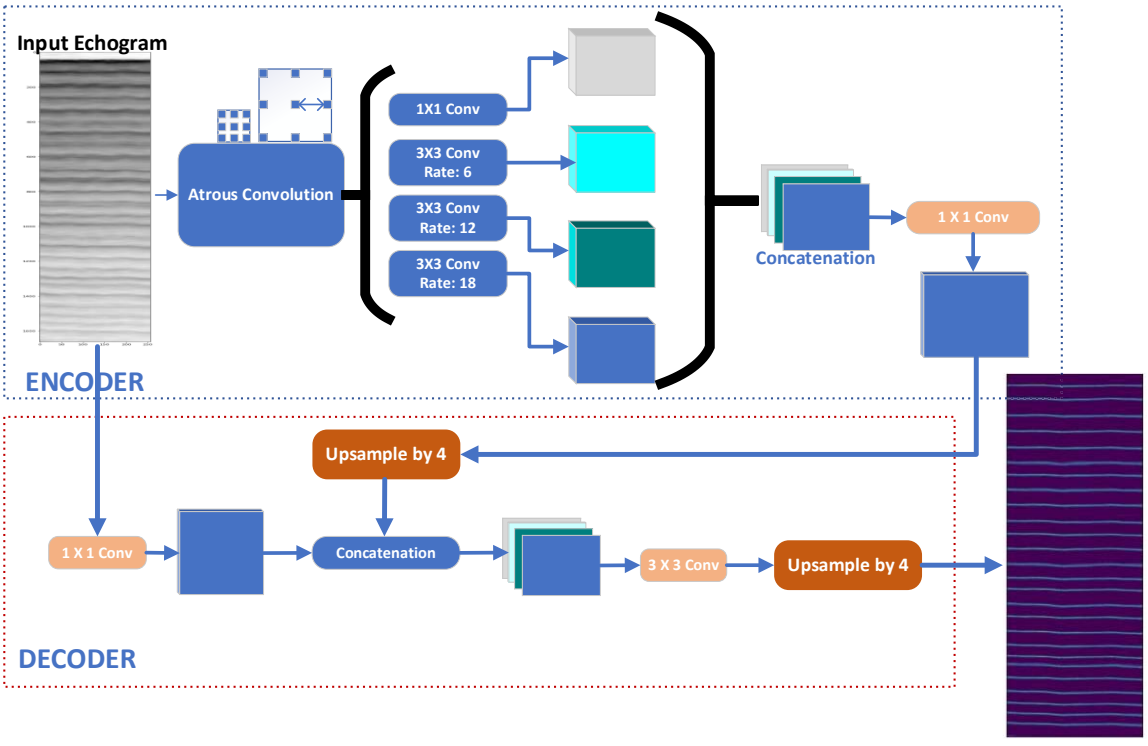


Figure 4-14: *Schematic of DeepLab architecture*

5. **Soft ensemble**: Soft ensemble is a simple routine in deep learning where the output of different models are naively combined without any further training. It is similar to using majority votes of the constituent models to decide the class of each input image pixel. To harness the strength of each of the previous four models, the soft ensemble model was created from their combination. Concretely, the ensemble is the average of the prediction map of FCN, U-Net, Attention U-Net, and DeepLabv3 models. This has the advantage of emphasizing details agreed on by most or all the models. However, it has the downside of ignoring important features identified by one or a minority of the models.

4-5-2 Training implementation details

For conciseness, we provide a summary of the dataset and implementation details for training the models below.

The models were trained on the Snow Radar dataset version 1 (detailed in Section 4-2). The echogram images and their corresponding binary labels are used for training the binary segmentation models. We employ very mild augmentation techniques such as color jitter, contrast transformation, left-right-flip, and slight elastic deformation to increase the robustness of the models. Random zooming and cropping were not used to avoid erasing the depth and spatial information in the range bin/depth and along-track axis respectively. These operations, particularly in the depth axis, make it difficult, sometimes impossible, to convert the tracked layer vectors back to depth in meters and stall the accumulation rate estimation process. To standardize the data, we enforced uniform dimensions across all echograms in the dataset by setting $N_t = 1664$ and $N_x = 256$.

For training, the Fully Convolutional Networks (FCN), U-Net, and Attention U-Net all have four stages of downsampling in their encoder paths. Each convolution block in the encoders consists of a sequence of convolution, dropout, batch normalization, and activation layers. The ReLU activation function is used in each convolution block. A similar pattern is followed in the decoder paths. In the decoder paths of U-Net and Attention U-Net, the naïve upsampling operation was replaced with the parameterized Conv2DTranspose, and the ReLU activation function was also applied within the decoder convolution blocks.

Also, unlike the conventional FCN which applies a single upscaling of the final downsampled feature map, we applied sequential stages of upsampling to generate the output binary map. However, there are no skip connections between the contraction and expansion paths. This makes it very similar to the U-Net model except that U-Net applies a double convolution block at each stage and skip connections exist between feature maps of similar spatial resolution in the encoder and decoder paths.

The Attention U-Net model is also similar to the U-Net except for the added attention gates. Our DeepLabv3 implementation uses ResNet50 architecture as the backbone with ImageNet pre-trained weights and dilation rates of [6,12,18] in the spatial pyramid pool.

All the models were trained with Binary focal loss with $\alpha = 0.25$ and $\beta = 2$ to mitigate the inherent binary class imbalance. Further details about the loss functions is delayed till section 6-2-4-2. The Adam [89] optimizer was used with an initial learning rate of 1e-3 that is decreased by a factor of 0.25 after 10 iterations at a plateau and EarlyStopping was implemented after a patience of 30 epochs at a plateau without improvement in the validation loss. Consequently, although the models were set to train for a fixed number of epochs, the terminating epoch for

each of the model differs slightly. However, we observed a mean epoch of 120 for all four models.

Table 6: *Model training hyperparameter summary*

Training hyperparameters	**Value**
Input image dimension	$N_t = 1664$ and $N_x = 256$.
Batch size	4
Initial learning rate	1e-3
Number of encoder blocks	4
Number of decoder blocks	4
Convolution filters per block	[16, 32, 64, 128]
Convolution non-linear function	ReLU
Output convolution activation function	Softmax
Number of prediction classes	1
Number of epochs	Varies
Loss function	Binary focal cross entropy
Class balancing factor (α)	0.25
Focusing parameter (β)	2

4-5-3 Hyperparameter tuning

Hyperparameter tuning is a critical step in machine learning model development. Unlike model parameters, which are learned from training data, hyperparameters are preset before training begins. These hyperparameters control aspects of the training such as learning rate, regularization strength and the architecture of the model. The selection of optimal hyperparameters is critical to improve model accuracy, reduce overfitting and overall predictive power of the model. However, finding the right combination of hyperparameters is often complex and computationally expensive. This can be alleviated by adopting some generally accepted benchmark values based on reported training schemes in literature.

To train the models a basic hyperparameter search was performed, focusing on key parameters such as batch size, convolution filter sizes, learning rate, and optimal training epochs. Due to the large dimensions of the input echogram, using a large batch size was not feasible on the available hardware. As the UNet family of models shares similar architectures, we adopted comparable hyperparameters. A batch size of 4 was ultimately chosen, as larger sizes frequently led to out-of-memory errors, especially when running multiple training sessions concurrently. Although exploratory runs with batch sizes of 8 and 16 were attempted, models like AttentionUNet often crashed due to limited GPU resources, reinforcing the decision to use a mini-batch size of 4 for the final model training.

Similarly, the size of the convolution block filters is also limited by available hardware. While larger filter sizes could theoretically improve the model's representational power, they frequently led to GPU memory exhaustion. Although some of the UNet model variants could have been trained with larger filter sizes, to ensure uniformity and consistency which is important for easy

comparison of their performance, filter sizes of 16, 32, 64 and 128 were used to train the UNet-family models.

An initial search was conducted to determine the optimal learning rate. Various constant learning rates—0.1, 0.01, 0.003, 0.001, and 0.0001—were tested over 100 epochs for the UNet model. Higher rates (0.1 and 0.01) failed to converge quickly, resulting in a large gap between training and evaluation loss. Rates of 0.003 and 0.001 performed better, with 0.001 slightly outperforming 0.003. However, constant learning rates proved inadequate, particularly in later training stages, where a slower rate is needed to navigate toward a local or global minimum.

To address this, a learning rate scheduler was implemented. The scheduler started with an initial learning rate, decreasing it gradually by a factor of 0.25 when training plateaued (i.e., when the validation loss did not decrease for 10 consecutive epochs). This adaptive learning rate was valuable, as it responded to the needs of each model's training cycle, reducing sensitivity to the initial learning rate. For all convolution-based models, an initial rate of 1e-3 was used, decreasing to 5e-6 in 0.25 increments.

Other hyperparameters, such as kernel size and network depth, were kept relatively uniform across models (except in specific cases like DeepLab). The Adam optimizer was chosen over RMSprop and SGD for its superior performance.

4-6 Model evaluation and discussion

In this section, we report the performance of the deep learning models on each section of the test set in three different stages.

4-6-1 Binary output evaluation

The immediate output of the binary segmentation models is probability heat maps derived from applying the sigmoid activation function to the logits output of the models. Visually inspecting the probability heat maps reveals the detected layers but the domain of the values in the heat maps are real numbers from 0 to 1 (sometimes containing negative values but this corresponds to noise). To convert these outputs into binary images, the probability maps are first thresholded followed by a simple non-maximum suppression algorithm. Details of our implementation of the non-maximum suppression algorithm is in 6-2-6-1. Qualitative examples of the binary output are shown in Figure 4-15 and Figure 4-16.

To assess how well the models classify "layer" and "no-layer" pixels in the echogram, we employ two evaluation metrics from the image processing domain: Optimal Dataset Scale (ODS) and Optimal Image Scale F-scores evaluation metrics. Both metrics utilize the F1-score (see section 3-2-4-2 for definitions of F1-score, recall, precision and accuracy), but they differ slightly in how they determine the optimal threshold for converting the edge probability map into a binary image. ODS considers a single threshold for the entire test set of the probability maps and finds the threshold that yields the highest average F1-score across all the images. Hence, it gives a general idea of how well the models are performing on the entire test set using a single setting. OIS, on the other hand, calculates the average of the best F1 score for each image in the test set by finding the optimal threshold for each specific image.

The SSIM metric (Structural Similarity Index Metric) is a measure of image quality that compares two images based on their structural similarity. It considers factors like luminance, contrast, and structural information, providing a more detailed assessment of image similarity

than traditional pixel-wise differences. More detailed discussion of SSIM is delayed until Section 6-2-4-4.

Table 7: *Optimal dataset scale and optimal image scale F1 scores*

	U-Net	**AttentionU-Net**	**DeepLab**	**FCN**	**Ensemble**
ODS	0.800	0.910	0.881	0.909	0.886
OIS	0.801	0.911	0.882	0.910	0.887

After binarizing, the one-pixel thick binary output for each model produced is also used to compute the recall, precision, accuracy and structural similarity index measure (SSIM) for the models. The result is summarized in Table 8.

Table 8: *Weighted average metrics for each of the models*

Model	**Recall**	**Precision**	**Accuracy**	**SSIM**	**F1**
U-Net	0.9790	0.9778	0.9790	0.9583	0.9784
AttentionU-Net	0.9779	0.9780	0.9779	0.9588	0.9780
DeepLab	0.9780	0.9781	0.9780	0.9586	0.9780
FCN	0.9779	0.9784	0.9779	0.9605	0.9782
Ensemble	0.9781	0.9783	0.9781	0.9588	0.9782

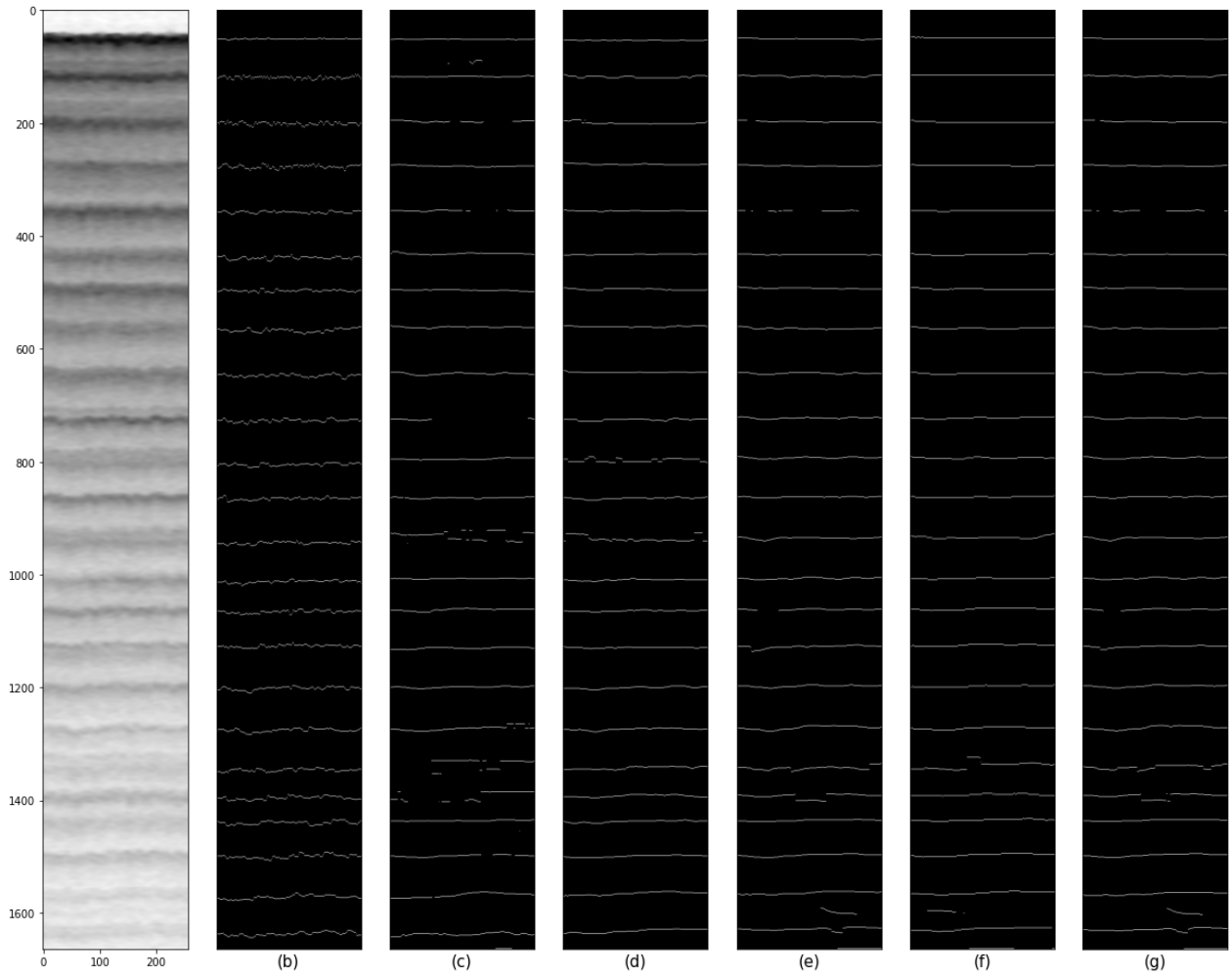


Figure 4-15: Full echogram image, (b) Ground truth annotation and models' binary outputs from (c) U-Net (d) AttentionU-Net (e) DeepLab (f) FCN (g) Soft Ensemble

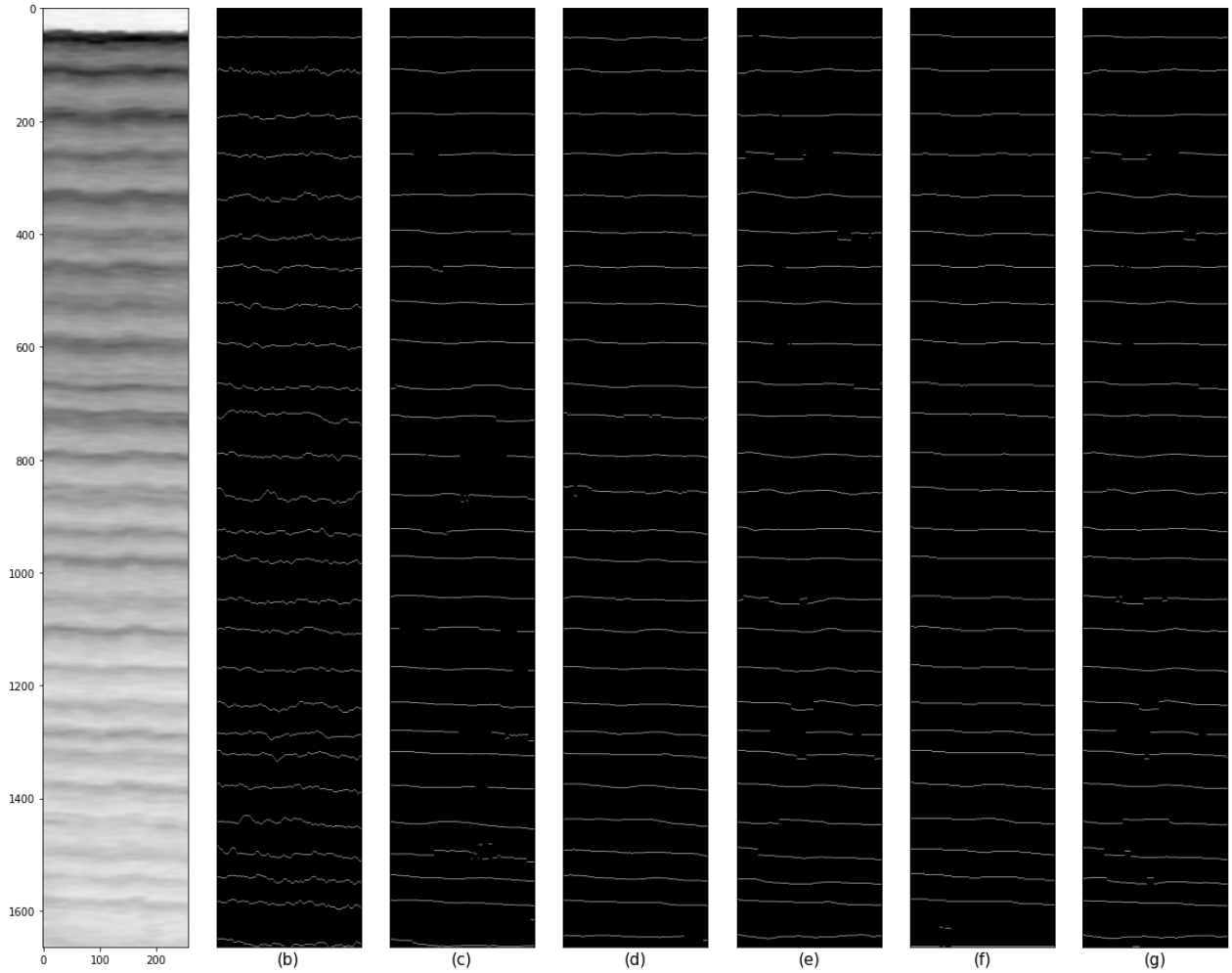


Figure 4-16: Another full echogram image, (b) Ground truth annotation and models' binary outputs from (c) U-Net (d) AttentionU-Net (e) DeepLab (f) FCN (g) Soft Ensemble

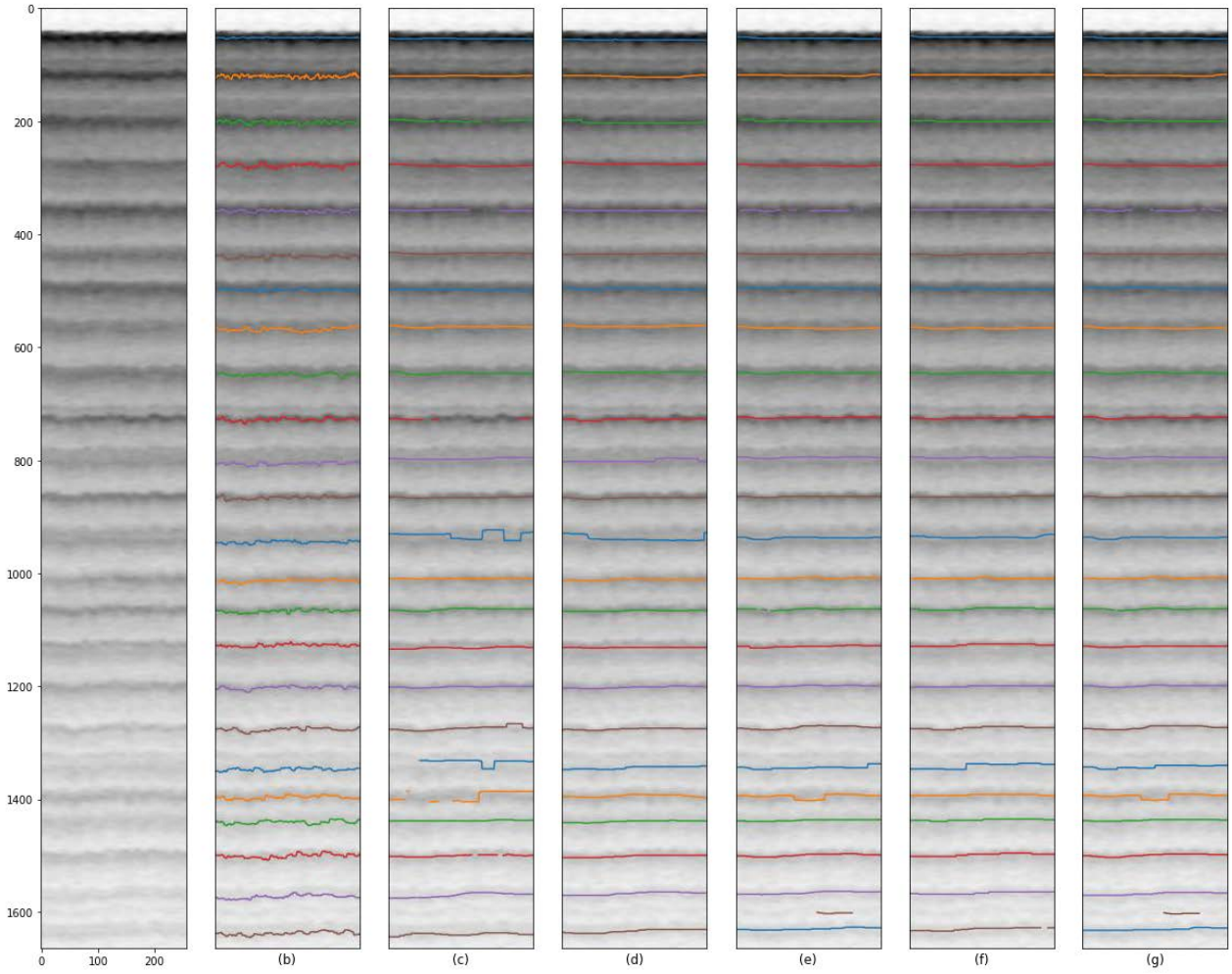


Figure 4-17: Echogram image with overlaid 1D contour tracked layer model outputs (b) Ground truth annotation (c) U-Net (d) AttentionU-Net (e) DeepLab (f) FCN (g) Soft Ensemble

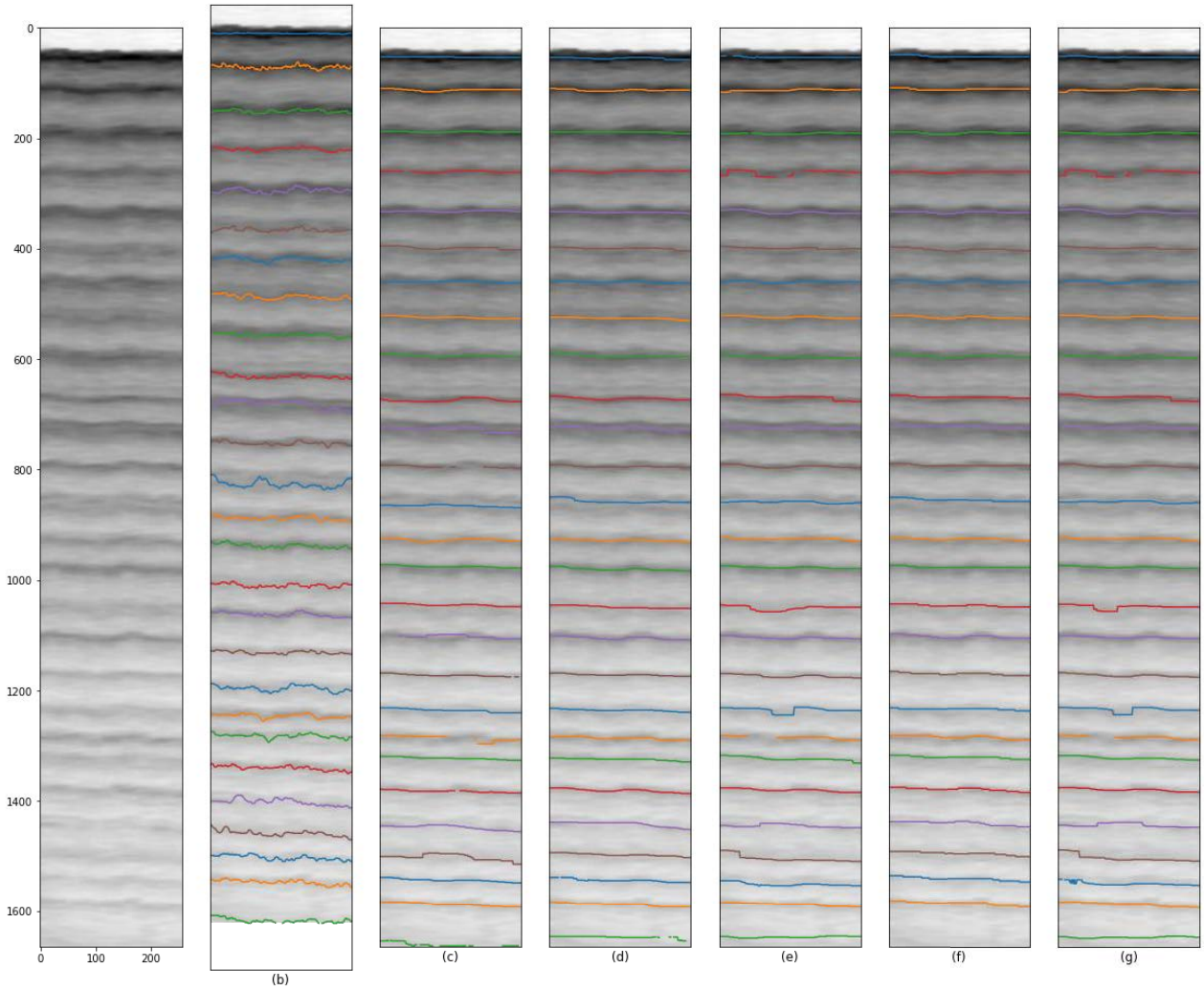


Figure 4-18: Another echogram image with overlaid 1D contour tracked layer model outputs (b) Ground truth annotation (c) U-Net (d) AttentionU-Net (e) DeepLab (f) FCN (g) Soft Ensemble

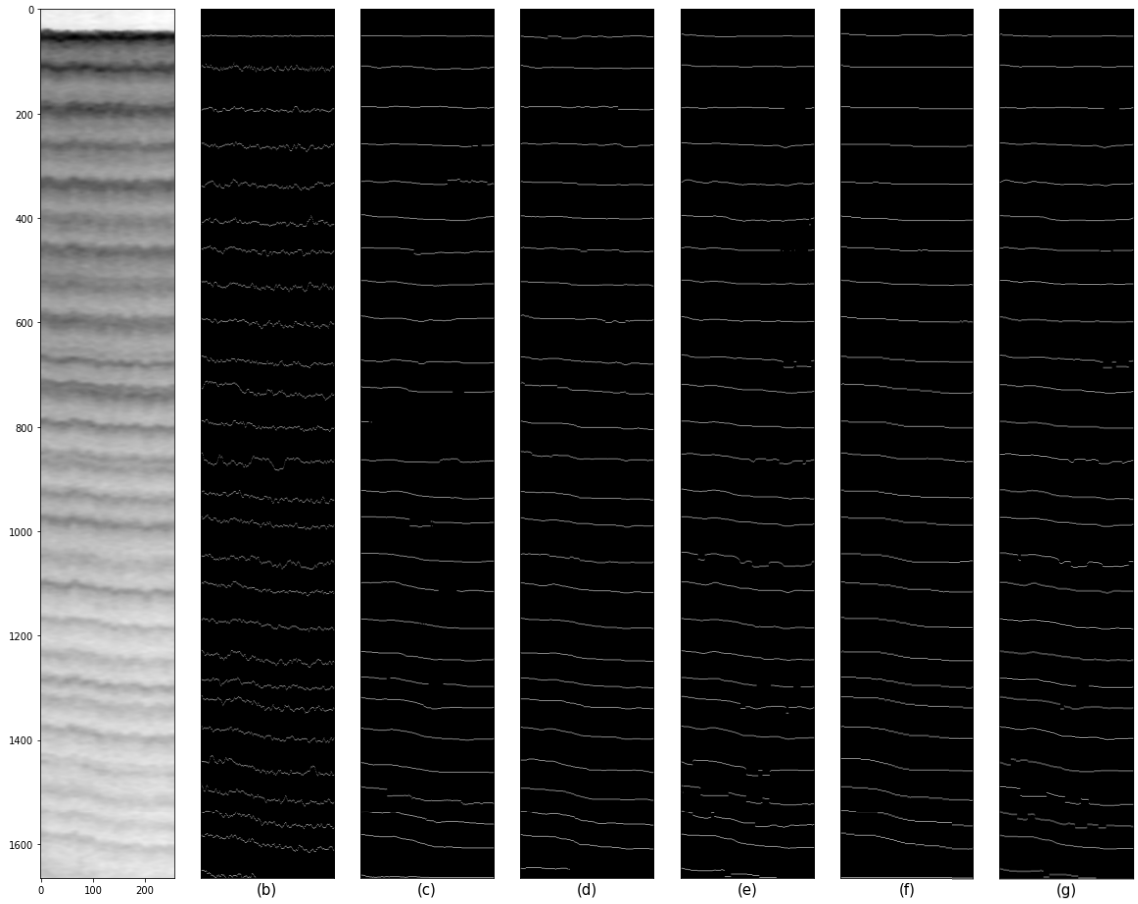


Figure 4-19: Echogram image and model binary outputs using (b) Ground truth annotation (c) U-Net (d) AttentionU-Net (e) DeepLab (f) FCN (g) Soft Ensemble

The values in Table 8 indicate near-perfect performance and are similar for all the models making it difficult to identify which model is performing consistently better than others.

However, on a closer look, we see this is because of the domination of the prominent 0 class in the task which skews the metrics. Separating the classes and computing the metric for each class reveals that the models are not performing too well on the layer (1) class pixels which we are more interested in.

Table 9: *Recall, Precision and F1 score for each binary class.*

Models	**Recall**			**Precision**			**F1**		
	Class 0	**Class 1**	**Unweighted average**	**Class 0**	**Class 1**	**Unweighted average**	**Class 0**	**Class 1**	**Unweighted average**
U-Net	0.990	0.068	0.529	0.988	0.077	0.532	0.989	0.072	0.531
AttentionU-Net	0.988	0.084	0.536	0.989	0.083	0.535	0.989	0.083	0.536
DeepLab	0.989	0.089	0.539	0.989	0.088	0.538	0.989	0.089	0.539
FCN	0.989	0.104	0.564	0.989	0.099	0.544	0.989	0.101	0.545
Ensemble	0.988	0.095	0.541	0.989	0.094	0.541	0.988	0.094	0.541

Table 9 shows the unweighted Recall, Precision and F1 scores for each class and their average. This result shows that the ensemble model and FCN are the best performing with an F1 score of about 10% on the positive class. It is important to note that a perfect score of 1, implying a perfect overlap, is not expected for these metrics. This is because the layers are multiple pixels wide and the optimal pixel within the layer's thickness is not well defined. A result that is only one pixel off from the ground truth will get a poor score even though the result is acceptably good. A consistent criterion for selecting the layer pixel (e.g. always tracking the leading edge or always tracking the middle of the layer) is required to ensure that the estimation of the accumulation between successive layers is accurate. Thus, the low recall, precision and F1 scores of the models on class 1 does not necessarily mean that they are very poor and not useful models for layer tracking. However, it does show that there may be room for improvement.

Hence, there is a need for a different metric (other than the classic computer vision pixel accuracy) to convey how well the models are performing on the layer tracking and accumulation estimation task.

4-6-2 Tracking evaluation

For a metric that better assesses the results, we employ the N-pixel accuracy earlier defined in Equation (10) to investigate how well each model tracks the layers. This calculates the accuracy of the positive class prediction in the neighborhood of the ground truth annotation. First, the model segmentation maps are thresholded and binarized to 0 (no layer pixel) and 1 (layer pixel) before each individual layer 1-D contour can be extracted. Since the width of each snow layer in the echograms is several pixels thick, the result of initial thresholding to create the binary output is similarly more than one pixel thick for each rangeline of the identified layer. This requires additional processing to thin the prediction to identify a single layer pixel for each layer's rangeline. This is similar to the non-maximum suppression post-processing algorithm in the computer vision community.

Concretely, the models' binary raster output (i.e. the matrix of 0's and 1's of dimension $N_t \times N_x$) is post-processed to create layer vectors ($N_L \times N_x$) that uniquely track and identify each of the annual snow layers in the echogram. This is done using a post-processing routine that extracts the row index (range bin) of all layer-containing pixels and clusters them into individual layers. The row indices of all pixels containing a layer (i.e. 1s in the models' binary output) are first extracted. Next, exploiting the fact that no two snow layers cross each other because of the snow accumulation between them, each annual snow layer is inherently distinct and can be separately identified. The post-processing algorithm, therefore, clusters the extracted row indices for each

snow layer to form the $N_L \times N_x$ layer vector output (N_L layers of size $1 \times N_x$). This format uniquely identifies the range bin in each layer for all the echogram's N_x rangelines and encodes it as a single row vector which is identical in structure to the ground truth layer vector (explained in Section 4-4-1).

Using the tracked 1D layer contour, the N-pixel accuracy (% of layer pixels that are less than or equal to N pixels from the ground truth) for N = 2, 5 and 10 are computed as well as the mean absolute error (MAE). Here, the MAE is the average absolute error between the prediction and the ground truth for all layer pixels as defined in Equation (12):

$$MAE = \frac{1}{N_{echo} N_L N_x} \sum_{n=1}^{n \le N_{echo}} \sum_{l=1}^{l \le N_L} \sum_{j=1}^{j \le N_x} \left| \hat{Y}_{n,l,j} - Y_{n,l,j} \right| \tag{12}$$

Table 10: *N-pixel accuracies and Mean Absolute error of each model*

	2px	**5px**	**10px**	**MAE**
U-Net	0.1125	0.5499	0.8489	6.5227
AttentionU-Net	0.1309	0.7293	0.9434	4.7331
DeepLab	0.1679	0.6523	0.9237	5.0119
FCN	0.2946	0.8015	0.9638	3.8132
Ensemble	0.1780	0.6671	0.9274	4.9081

The result in Table 10 shows that 71 to 89% of the layer predictions are more than or equal to 2 pixels from the ground truth. This error rate drops to 4 to 15% when the error margin is increased to more than or equal to 10 pixels. The FCN model shows superior performance for all metrics compared to the other models (71% error rate for ≥ 2 pixels and 4% for ≥ 10 pixels).

To tie the pixel margins and MAE estimation back to the layer tracking estimation physical problem, we can find the layer thickness error in meters instead of pixels. The Snow Radar image fast-time sampling is $\Delta_t = 0.08517$ ns and assuming a dielectric of $\epsilon_r = 2$, which is in between fresh fallen snow ($\epsilon_r \sim 1.5$) and solid ice ($\epsilon_r \sim 3.15$), gives a fast-time or row pixel height of

$$\Delta_r = \frac{c\Delta_t}{2\sqrt{\epsilon_r}} = 0.01\text{m}. \tag{13}$$

This shows that the worst performing model (U-Net) with MAE = 6.5 pixels and best performing model (FCN) with MAE = 3.8 pixels have mean errors of 5.9 cm and 3.4 cm respectively. Note that the reported MAE and N-pixel accuracies are only reported for pixels where the models and ground truth annotation simultaneously have valid predictions.

Cases where only the model or only the ground truth produced a result have been excluded. This situation typically arises from low-probability segmentation mask values, which may be due to either poor pixel quality in the echogram image or errors in the deep learning model. In both scenarios, the layer pixels fail to be converted to 1 during binarization. In certain sensitive applications and some geographical locations (e.g. wet snow zones), it is desired that consecutive predictions are available for all rangelines. The predicted layers need to be continuous with no missed layer detections that can cause gaps in the predictions. Therefore, we also investigate and report the percentage of missed pixels for each model.

In the tables below, “whole-layer pixels” refers to echograms where the entire layer is missing. “Intra-layer pixels” refers to layer pixels where only a portion of the layer is missing. The total number of layers in the test set across all images is 24,407 layers and the total number of layer

pixels is 6,184,704. There are 3165 and 21,242 layers in test groups L1 and L2 respectively. The results for both test groups are shown separately and combined. It should be noted that L2 includes echogram imagery that are more difficult to track.

Table 11: *Missing layer pixel evaluation showing "whole layer pixels", "intra-layer pixels" and the combined percentage of layer pixels missed for the models.*

	Whole layer pixels			Intra-layer pixels			Combined Percentage
	L1	L2	L1 + L2	L1	L2	L1 + L2	
U-Net	33 1.04%	1418 6.67%	1451 **7.72%**	42,228 5.23%	618,038 11.49%	660,266 **16.73%**	**24.44%**
Attention U-Net	14 0.44%	407 1.92%	421 **2.36%**	4,171 0.52%	137,555 2.56%	141,726 **3.07%**	**5.43%**
DeepLab	11 0.35%	248 1.17%	259 **1.52%**	15,215 1.89%	265,010 4.93%	280,225 **6.81%**	**8.32%**
FCN	16 0.51%	328 1.54%	1.46 **2.05%**	3,883 0.48%	133,847 2.49%	137,730 **2.97%**	**5.02%**
Ensemble	10 0.32%	232 1.09%	0.99 **1.41%**	13,176 1.63%	246,719 4.59%	259,895 **6.22%**	**7.63%**

4-6-2-1 Evaluation of layer tracking based on echogram image quality and along-track distance

As discussed in Section 4-2-1, the Snow Radar ML Dataset_version1 contains echograms from different parts of Greenland that are representative of the different snow accumulation patterns over the ice sheet. The absence of moisture and melting in a polar region such as the dry snow zone where little or no melting occurs all year long ensures that the annual snow deposition

stratigraphy is always maintained. As a result, echograms from such locations have crisp layers resulting in better deep learning algorithm results. Here, we investigate the classification and tracking performance of the models on the different categories (particularly the L1 and L2 groups) of test echogram images. L1 echogram images have the best image quality with distinct layers and high signal-to-noise ratio (SNR) between the layer pixel signal energy and the background noise. L2 echogram images are not as easy to trace as L1 images, but the snow layers in the echograms are still visible and can be identified and tracked.

The L3 images are generally of low quality and may lack discernible snow layering. However, their characterization is crucial due to their frequent occurrence, sometimes at the beginning or the end of a survey flight line. To design a deep learning model that can consistently track snow accumulation over several kilometers of ice accumulation, the model needs to be exposed during training to these "imperfect and difficult" echogram images. This allows the model to either learn to disregard them or effectively track the snow surface despite their image quality limitations. However, for most tracking performance evaluation in this work, the L3 images are omitted as the priority is on L1 and L2 images that make up the bulk of the layers that need to be tracked.

4-6-2-2 Model performance evaluation based on echogram image quality

Here, we investigate the models' ODS and OIS F1 scores and mean absolute error for each of the L1, L2 and L3 Snow zones.

Table 12: *ODS and OIS for L1, L2 and L3*

	L1		L2		L3		Combined L1 +L2 +L3	
	ODS	**OIS**	**ODS**	**OIS**	**ODS**	**OIS**	**ODS**	**OIS**
U-Net	0.916	0.917	0.789	0.790	0.214	0.214	0.800	0.801
AttentionU-Net	0.971	0.972	0.908	0.909	0.218	0.218	0.910	0.911
DeepLab	0.959	0.960	0.880	0.881	0.159	0.159	0.881	0.882
FCN	0.957	0.958	0.913	0.914	0.187	0.187	0.909	0.910
Ensemble	0.962	0.963	0.885	0.886	0.167	0.167	0.886	0.887

Table 13: *Mean absolute error (MAE) for L1, L2 and L3*

	L1	L2	L3
U-Net	4.222	6.872	70.987
AttentionU-Net	3.466	4.970	69.266
DeepLab	3.956	5.118	41.661
FCN	3.982	3.742	47.084
Ensemble	3.873	5.010	42.672

The noticeable decline in the performance of most of the models as the echogram image quality decreases confirms the hypothesis that the performance of the models is strongly linked to the quality of the echogram images. Therefore, to achieve broad generalization to echogram images from different snow zones and image quality, a robust deep learning architecture that can maintain good performance irrespective of image quality is required. Of the examined deep

learning architectures, the FCN model exhibits better robustness to the declining echogram image quality.

Table 14 : *N_pixel accuracies for each echogram image quality segment*

	L1			L2			L3		
	2px	**5px**	**10px**	**2px**	**5px**	**10px**	**2px**	**5px**	**10px**
U-Net	0.141	0.719	0.973	0.108	0.524	0.831	0.001	0.077	0.225
Attention U-Net	0.269	0.806	0.986	0.110	0.719	0.939	0.001	0.050	0.174
DeepLab	0.124	0.761	0.987	0.175	0.637	0.916	0.001	0.022	0.175
FCN	0.095	0.788	0.979	0.325	0.805	0.963	0.001	0.084	0.278
Ensemble	0.141	0.768	0.988	0.184	0.653	0.920	0.001	0.044	0.183

4-6-2-3 Model performance evaluation based on along-track length and echogram quality

We also examine the effect of the along-track length of the echograms on the performance of the models and how this performance is influenced by image quality. In Table 15, the performance of the models is reported based on the echogram along-track distance. The 10-km echograms have the lowest MAE compared to 2-km and 5-km echograms. This strongly suggests that future work should explore the use of longer echograms to more confidently find the optimal value for this hyperparameter. To further scrutinize the relationship, Table 16 shows the performance of the models as a function of the along-track distance and echogram image quality simultaneously.

Table 15: *Mean absolute error (MAE) based on along-track length*

	2 km (Count = 911)	5 km (Count = 366)	10 km (Count = 15)
U-Net	6.793	7.263	3.875

Attention U-Net	5.049	5.394	3.750
DeepLab	5.177	5.231	4.006
FCN	4.027	3.959	4.193
Ensemble	5.064	5.168	3.909

Combining the result in Table 15 and Table 16 shows that the performance of the models on 2 km and 5 km echograms are comparable. However, in L2 echograms (which are most common), the FCN architecture has low MAE for 5 km echograms.

It should be noted that the Snow Radar Dataset version 1 test set contains a very limited number of 10 km echograms. This would be increased in the next iteration of the dataset to allow for a fair comparison of the performance of models on echogram images with long along-track length.

Table 16: *Mean absolute error (MAE) based on echogram zone and along-track length*

	L1			**L2**		**L3**	
	2 km	**5 km**	**10 km**	**2 km**	**5 km**	**2 km**	**5 km**
U-Net	4.337	4.092	3.875	6.797	7.062	68.717	74.328
AttentionU-Net	3.376	3.565	3.745	5.020	4.843	53.642	91.834
DeepLab	3.956	3.931	4.006	5.145	5.052	42.629	40.381
FCN	3.928	4.020	4.193	3.791	3.620	51.727	40.377
Ensemble	3.870	3.862	3.909	5.030	4.959	42.162	43.345

4-6-3 Generalization evaluation

The overall goal of this work is to have a deep learning model that performs well, not just on the test set of the dataset but on echograms other than those in the training set. While it is not expected that the models would generalize (without extra finetuning or pre-training) to echograms that are characteristically very different such as echograms from the Multichannel Coherent Radar Depth Sounder (MCoRDS) whose center frequency is orders of magnitude different from the Snow Radar and focuses on a different section of the ice column, it is, however, expected that the models should generalize to echograms from similar radar systems and snow zones. Particularly given the large amount (approximately 50-60% of available echograms) that are yet to be tracked and several new science missions scheduled to collect more data, a model that can generalize to echograms in the wild would be valuable to automatically track layers to extract ice accumulation information.

As such, we test the trained models on echograms outside the training set. These "new" test echograms are mostly from the same campaign year (2012) but not from the ML SR _dataset_v1 dataset and include data collected in later years with similar radar hardware settings. These echograms are first pre-processed in a similar way to the preprocessing of the echograms in the dataset. However, these echograms do not have ground truth annotation, and therefore, they can only be evaluated qualitatively at this time.

Figure 4-20 and Figure 4-21 demonstrate the generalization abilities of the models. The test echogram is from the dry snow zone and contains well defined snow layers. When used to perform inference, most of the models correctly classified and tracked the individual snow layers in the echograms. The DeepLab model shows its limitations by missing some of the snow layers.

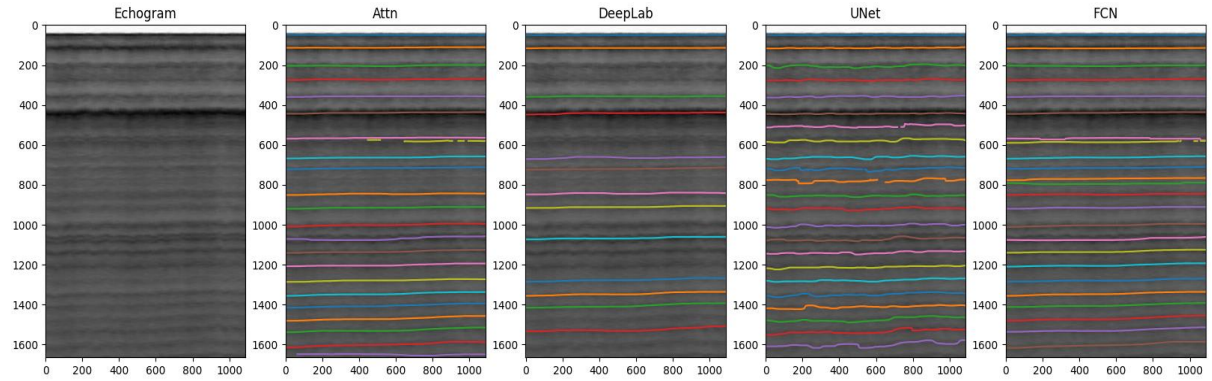


Figure 4-20: "New" echogram from 2017 data tracked with trained models

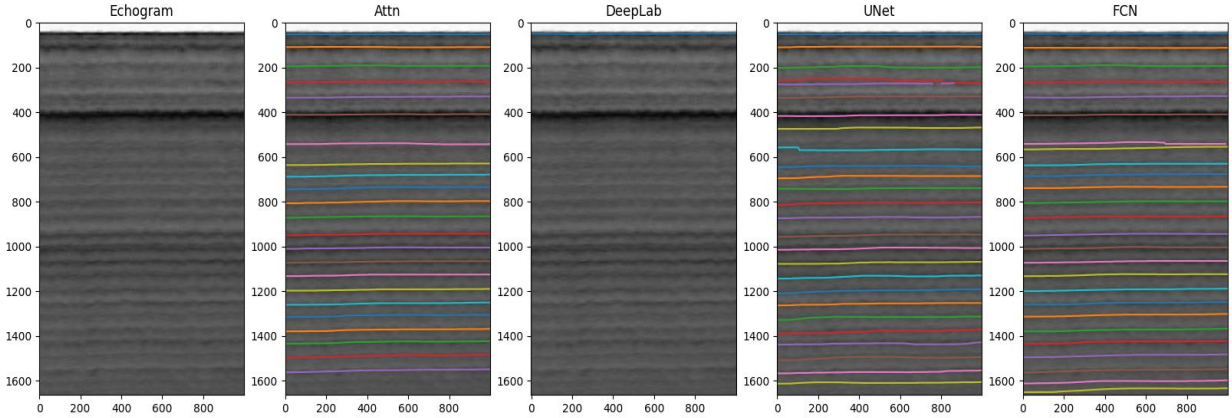


Figure 4-21: Another "new" echogram from the dry snow zone

The test was conducted on several dry snow zone echograms and the models (including the U-Net model) perform satisfactorily when the echogram image quality is good, and the snow layers can be easily identified. Specifically, these "good" echogram images have layer pixels with high SNR and clear discrimination between the layer's pixel peaks and the "no-layer" pixel background noise floor between layers.

4-6-3-1 Model limitations

Despite the performance of the models on dry snow zone echograms and their ability to generalize broadly to other echograms from different data collection years and slightly different radar hardware configurations, the models failed to reproduce similar performance on some echograms with certain features. The echogram in Figure 4-22 is an example of an echogram where all the models failed to correctly classify *all* the layer pixels and consequently failed to track the layers. Accurate tracking of the layers in post-processing is contingent on identifying most (if not all) the layer pixels correctly. For these types of echograms, where the layer stratigraphy orientation is curved or tubular, it is crucial for the models to identify all the layers pixels in the curved region to ensure accurate layer tracking.

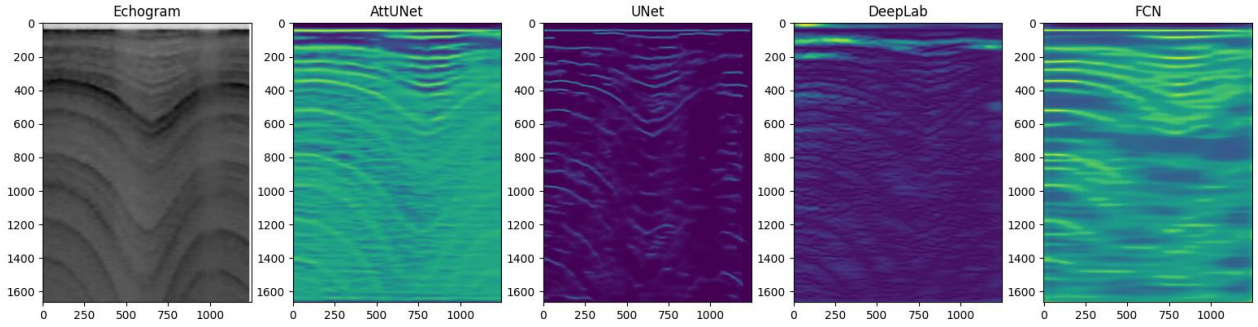


Figure 4-22: Sample echogram and activation maps where all the models fail to identify the snow layers

The failure of all the models on these types of echograms prompted a closer examination of the characteristics of the echograms and model architectures to understand why the models did well on some echograms but failed on others.

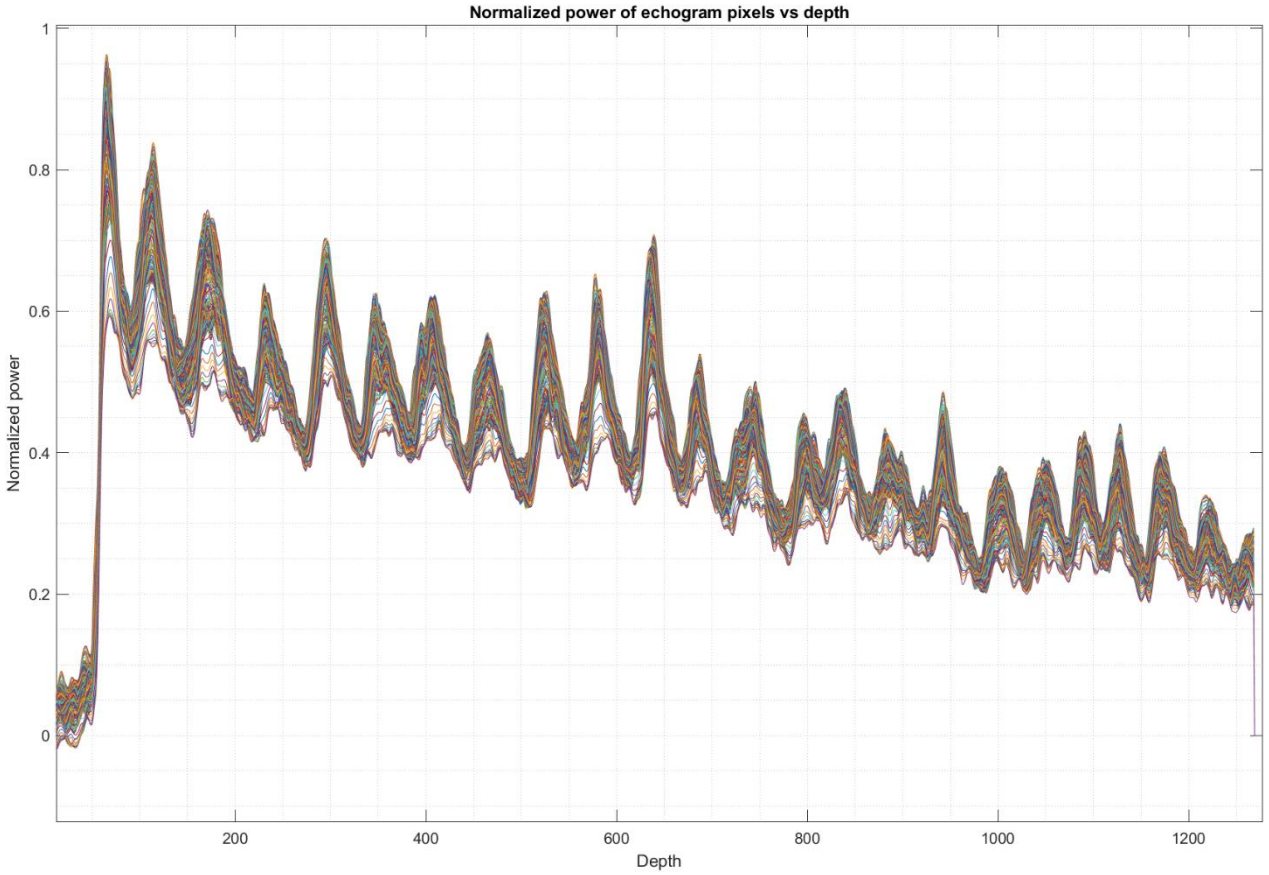


Figure 4-23: A scope plot of all the rangelines in a sample echogram from the dry snow zone

The figure above shows the plot of the normalized power as it is depicted by the echogram image but plotted as a function of depth. The A-scope (Amplitude scope) plot shows the same dataset as the echogram image but with a different visualization where the columns are plotted as vectors instead of a 2D color-coded surface map. It is created by plotting the linear normalized power of each column (rangelines) in the echogram image. The echogram image from which the A-scope was plotted is shown in Figure 4-24. As shown in the A-scope plot in Figure 4-23, when the snow zone has well-preserved layer stratigraphy, the radar received backscatter as the aircraft flies along-track shows that the layer peaks align with each other for each range line. The quasi-stationarity of the snow layers in this snow zone causes all the received backscatter for the layers to co-localize which corresponds to the “almost perfectly straight” layers seen in the echogram

image. It must be noted that this A-scope is for a processed echogram whose rangeline is formed from coherent and incoherent averaging and has undergone some fast time filtering to reduce the high frequency component in the received backscatter.

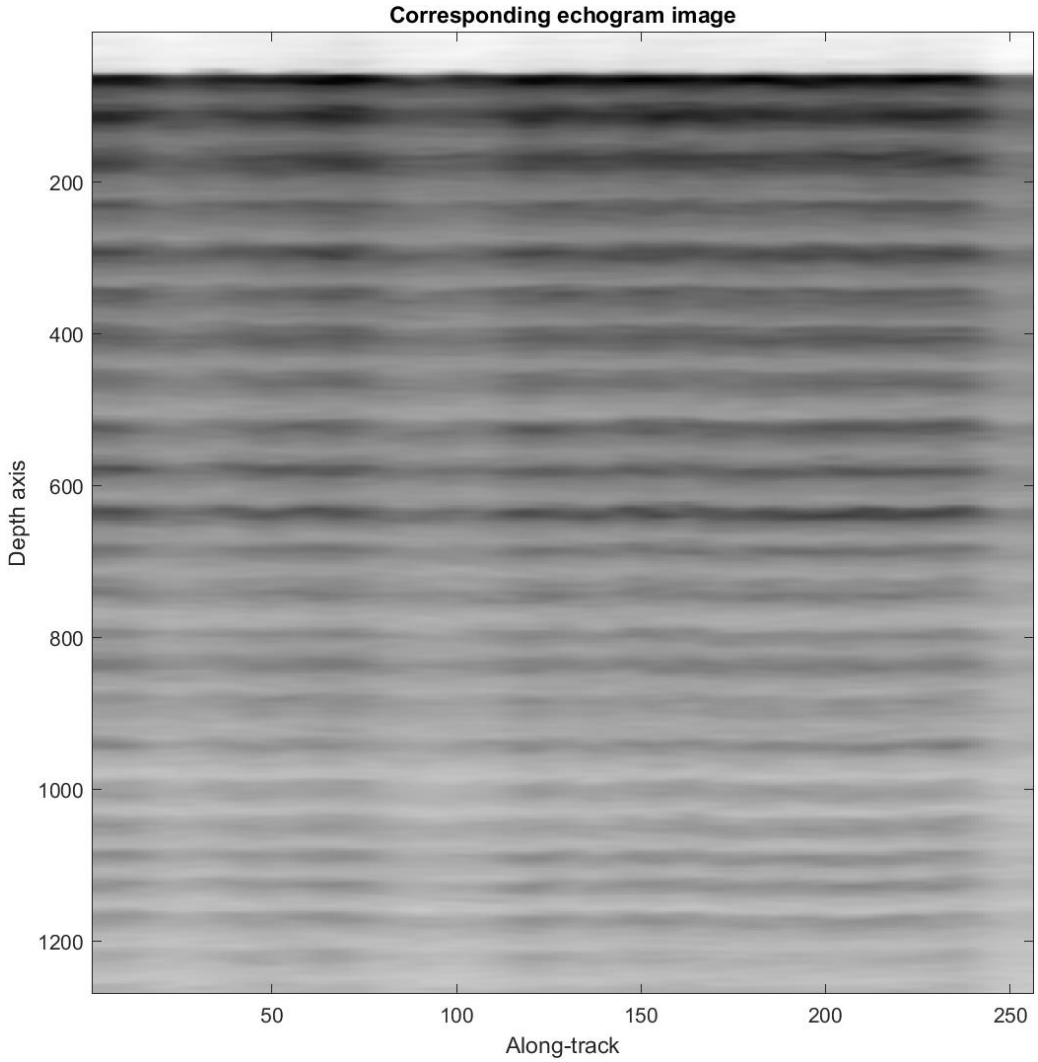


Figure 4-24: Echogram image corresponding to the A-scope plot

Figure 4-25 is the annotated A-scope plot highlighting the layers (signal peaks) with blue arrows and the interlayer background noise floor (signal troughs) with red lines. In these echograms, the peaks corresponding to the layer are easily distinguishable due to its high backscatter SNR relative to the adjacent power trough before the next layer.

As such, it is seen that when the layer pixels, which corresponds to signal peaks, are considerably higher than the intra-layer noise floor and the peaks cluster together (blue arrows) in the A-scope, the models can identify and track the layers along track. This property is similar to the decision boundaries between the probability density functions (PDF) of the two classes in a binary detection problem. In such binary detection problem, a clear separation between the PDFs of the classes implies that there is a detection statistic that can be used to distinguish the classes. However, the decision boundaries of the deep learning models are not based on the pixel values alone but includes the spatiotemporal relationship information between the neighboring layer pixels. The spatiotemporal relationship learned by the models is why layers well below the surface (such as the last blue arrow) can still be detected and tracked despite having lower pixel values than the earlier intra-layer noise floor such as the one between the surface and layer 1 (first red line).

In contrast, the echograms with curvilinear layer orientation due to rapidly varying accumulation rates are more likely to have a poor distinction between the layer (1) peaks and no layer (0) throughs due to the presence of refrozen melt water features.

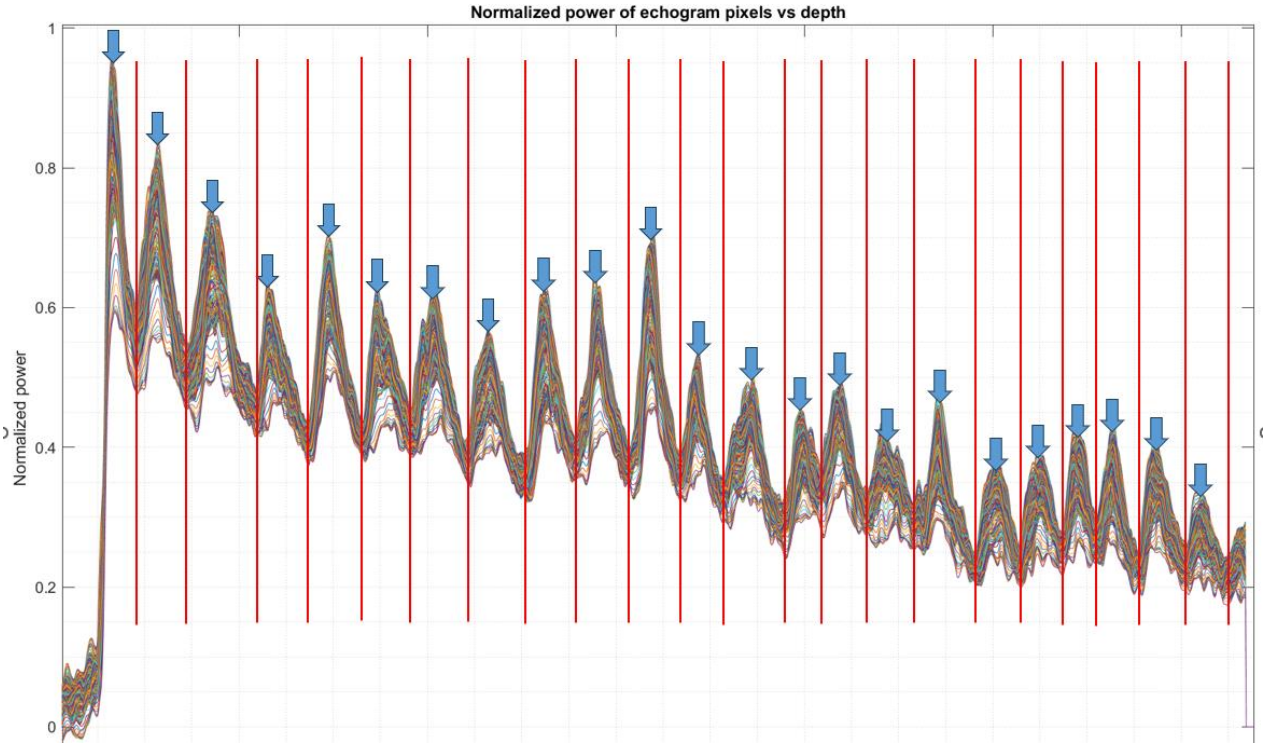


Figure 4-25: Annotated A-scopes to show each layer and linear power trough between layers

The varying spatial accumulation pattern between different geographical regions is a key factor causing the models to fail. Another issue is reduction in SNR due to the aircraft roll angle. As the aircraft maneuvers along the flight path, oftentimes, the aircraft's orientation changes to stay on course or make a turn. This causes the radar antenna to tilt, making the radar beam deviate from nadir, which lowers the signal power coming from the desired normal-incidence layer scattering and increases unwanted off-nadir backscatter. As such, these rangelines appear faded and blurred in the echograms.

Concretely, echogram layer tracking is successful when these two sequential processes are completed accurately: layer pixel classification and along-track tracking. Layer pixel classification identifies the candidate layer pixels, and this must be done accurately to ensure successful layer tracking. However, as earlier mentioned, multiple pixels around the layers' peak

are returned as the layer pixels in the model output heat map. The subsequent goal is to identify which of the candidate layer pixels for each rangeline coincide with the layer peak and achieve optimal connectivity between adjacent layer pixels to form a smooth tracked layer.

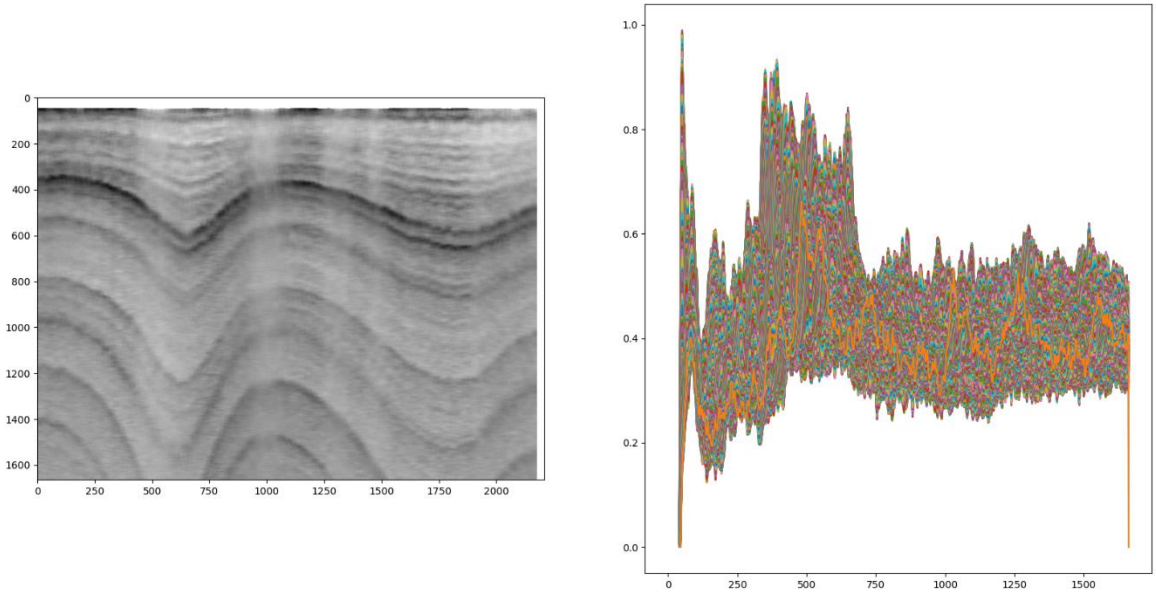


Figure 4-26: A-scope plot of all the rangelines in a sample echogram from a transition snow zone

For dry snow zones, the layer tracking problem is easier because the layers are flatter, and even when there is a reduction in SNR for some rangelines, it is easy to use information from neighboring rangelines to estimate or interpolate the faded ones. However, for the non-flat and curved orientation of the layers in the wet snow zone echograms, the relationship between neighboring lines is more complicated making the along-track tracking task difficult in these types of echograms.

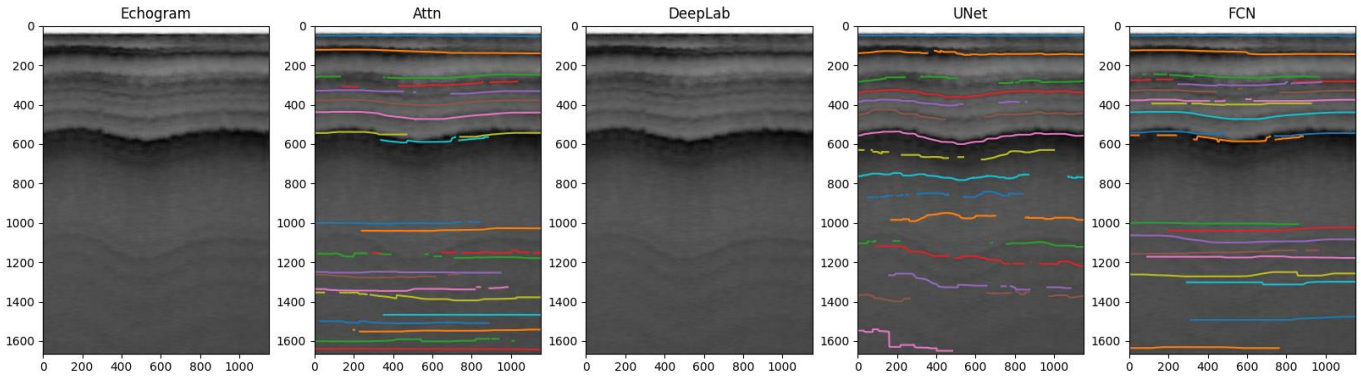


Figure 4-27: Qualitative example of the tracking performance of the models on echograms with curved snow layer orientation. (a) AttentionU-Net (b) DeepLab (c) U-Net (d) FCN

This situation is sometimes further worsened by the presence of small clusters of noisy peak pixels and artifact pixels caused either by the imperfections in the radar system or inherent in the remote data collection method. These artifact pixels are usually around the layer troughs of deeper layers (sometimes with low SNR) and mimic layer pixels. They often trick the models to classify them as valid layer pixels. These erroneous layer peaks are difficult to see on a cursory look at the echograms but closer inspection of the rangeline A-scopes reveal these peaks. In Figure 4-27, the models (AttentionU-Net, U-Net and FCN) incorrectly identified and tracked these artifact pixels around the deeper layers causing the incorrect along-track tracking.

Dealing with these data artifacts is non-trivial and can mislead the post-processing tracking if the models have not learned to disregard such pixels. As such, the models need to be given some awareness of rangeline-to-rangeline (sequential) tracking during training and learn to ignore pixels that do not cluster with others to form a uniform and smooth track along the rangelines.

It is also important to note that the mixed performance of the models (good tracking on the dry snow zone echograms but poor on wet snow zone) can also be attributed to the sparsity of wet

snow zone echograms in the training dataset. The dominance of dry snow zone echograms during training biases the model towards the most encountered echogram type.

Approaches to improve the performance and generalizability of the models, particularly on echograms from non-flat accumulation zones are investigated in the next chapter.

Chapter 5 METHODOLOGY (3)

5-1 Improving model generalizability

The quest to improve the trained models' ability to automatically track snow layers in echogram images beyond only those in the dry snow zone is critical since most of the collected data (fragmented into "segments") rarely only contain dry snow zone echograms. Some flights target the transition and wet snow zones. Therefore, deep learning models robust to the different accumulation zones would be a valuable tool.

The result obtained from the trained models revealed that the echogram snow layer tracking problem is a combination of layer pixel classification and along-track tracking problem. A model with good pixel classification performance is sufficient for dry snow zone echograms with nearly flat layers since the layer pixels are naturally aligned. However, layers with curvature require that the models learn spatial correlation in the along-track axis to not only classify individual layer pixels but also fill gaps where data quality is low due, for example, to the curvature or increased scattering that occurs more frequently outside the dry snow zone.

We approached this by developing a new model architecture adept at taking advantage of the intrinsic sequential information in the fast-time and slow-time axis of the echogram data.

5-2 Echogram vision transformer (EchoViT)

Echogram images can be viewed as a sequence of 1D rangeline data stacked in the along-track axis. As a result, the layer contour tracing problem can be framed as a sequential identification of the layer edge from one echogram column to the next. This sequential nature lends itself readily to auto-regressive and sequential deep learning model architectures.

In section 3-2-3, the sequential recurrent neural network (RNN) architecture was explored on the RowBlock problem, and it had better success than its convolutional based SkipMLP counterpart largely because it took advantage of the recurrence in the echogram data. The long short-term memory network (LSTM) was once deemed as the gold standard variant of the recurrent neural network but suffered major drawbacks of precluding parallel computation and having limited long-range dependency. It is also very susceptible to the vanishing gradient problem and very sensitive to training hyperparameters which all make the model difficult to train.

In recent years, the transformer architecture [29] with attention modules have been introduced to overcome some of the limitations of the earlier RNN architecture. It was first introduced in the Natural Language Processing (NLP) domain but has since become the de facto standard for almost all language tasks and many vision tasks given its fast computation time and support for parallel computation.

Transformers rely on a mechanism called *self-attention*, which allows the model to weigh the importance of different elements in the input sequence, irrespective of their position in the original input sequence.

5-2-1 Self-attention mechanism

As earlier mentioned, the self-attention mechanism is the cornerstone of the transformer architecture. It allows the model to weigh the importance of different elements in the input sequence when processing it. The attention mechanism takes the input data and learns the relationship between each "token" (the unit input into the transformer, an image patch for

computer vision problems) and how they are related to others and the impact that each input token has on others in the context of the entire input sequence. It mixes this information to output a context-aware and semantically rich representation of the input first before passing it on to other processing stages in the transformer architecture. The self-attention mechanism can be broken down into:

a) computation of Query, Key and Value vectors
b) calculation of attention scores
c) scaling the attention scores

a.) Computation of Query, Key and Value vectors: It does this by creating 3 copies of the input known as the query (Q), key (K) and value (V) vectors. These vectors are obtained by multiplying the embedded input tokens by the corresponding weight matrices W_Q, W_K, and W_V.

$$\begin{aligned} Q &= EW_Q \\ K &= EW_K \\ V &= EW_V \end{aligned} \tag{14}$$

where E represents the input 2D echogram data and W_Q, W_K, and W_V are the learnable weight matrices of the model. These learned matrices Q, K, V are now used as inputs to subsequent layers of the model.

b.) Attention scores calculation: The soft attention module creates a representation of the sequential echogram data that is semantically rich containing the inter-relationship between the input tokens and how they contribute to the model output. The attention score is computed by taking the dot product of the query vector of one token with the key vector of another token in the sequence of image patches resulting in $Q\ K^T$. The score between a pair indicates how much

focus one element should have on the other. In the context of echogram layer tracking, this parallel attention score forces sequential elements (rangelines or range bins) that need to be connected to form a 1D contour to learn to focus on the important pixels (those with higher attention scores) and to disregard others (those with lower scores).

c.) Scaling the scores: To prevent the dot products from growing too large in magnitude, which can push the output softmax function into a region where the gradients are small enough that gradient descent slows down too much, the scores are scaled by the square root of the dimension of the key vectors (d_k) to give $\frac{QK^T}{\sqrt{(d_k)}}$

The attention mechanism is concluded by applying the softmax operation to scaled scores to normalize them into a pseudo probability density function with the attention weight sum equaling one. However, it is important to note that certain transformations are applied to the input before the self-attention mechanism. This is represented by the input patch and embedding layer in Figure 5-1.

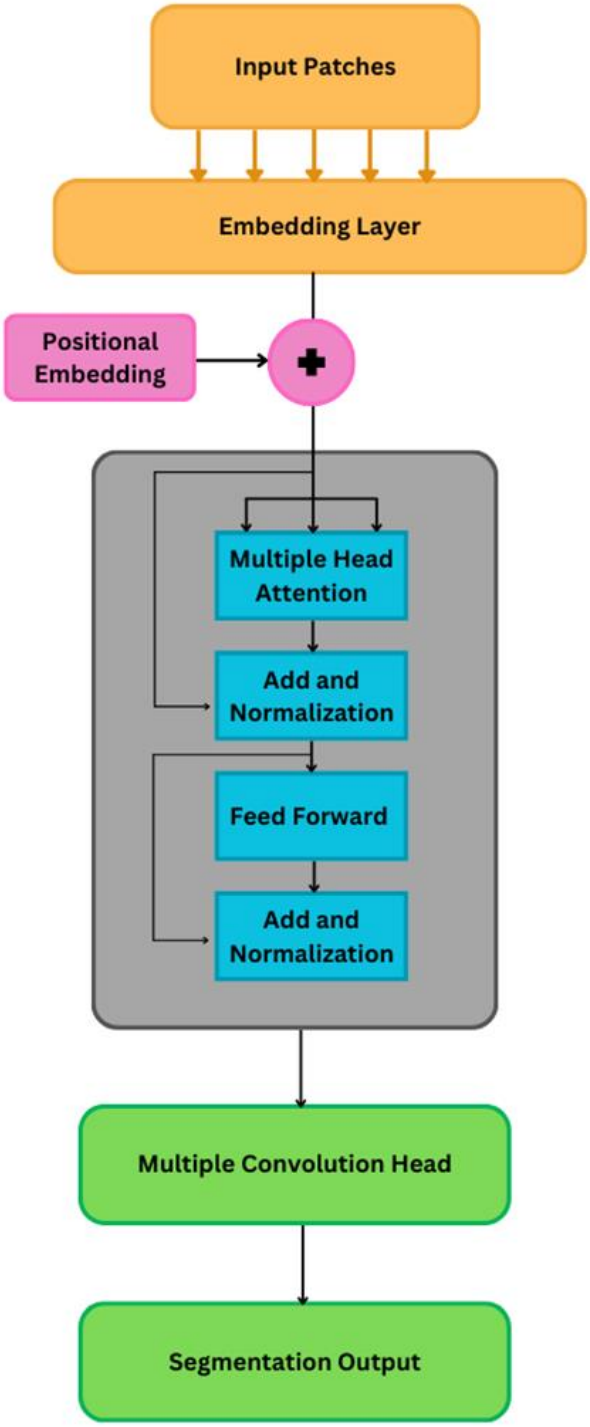


Figure 5-1: Complete transformer auto-regressive architecture

5-2-2 Embedding layer and positional encoding

Similar to NLP tokens, the input patches are first embedded into a high dimensional manifold before passing them into the encoder. In NLP, each word is represented by a high-dimensional

vector obtained from an embedding matrix. In the case of echogram images, the patch, chosen based on a patchification scheme, is passed through a convolutional base layer and the feature vectors from this serve as the layer embedding. However, unlike deterministic embeddings such as eigen-value decomposition, Fourier transform, wavelet basis, etc., this embedding space is learnt entirely from the data, therefore, it can achieve better performance by making less assumptions and learning entirely from the input data. Starting with a random initialization of the transformation basis weights, the model during the forward and backward propagation phase of training learns the appropriate basis that best explains the latent information in the input images.

Next, since the self-attention module is position and order-agnostic, the initial position of each patch in the image is provided as additional information through position embedding. The positional encodings are added elementwise to the embedding output. The initial approach used in the seminal paper [29] was to use sinusoidal functions which vary smoothly and provide unique encodings for each position. This is static and not updated during training. However, a more recent way of implementing this is to also use a trainable embedding layer that is optimized with other weights in the network during training.

The transformed input patches with added positional information are then passed as input to the earlier described self-attention module. Importantly, the power of the self-attention module is harnessed by performing multiple parallel copies (known as “heads”) of the operation. This is referred to as “multi-head self-attention” and is repeated several times to form the compound “multi-layer multi-head self-attention,” drastically improving the overall representation power of the architecture.

5-2-3 Layer normalization, skip connections and feed-forward networks

As shown in Figure 5-1, the transformer encoder (where the self-attention operation is performed) consists of the normalization layer, self-attention layer, skip connections and multilayer perceptron head. The output of the multi-head self-attention goes through the normalization layer which normalizes the inputs to each layer (as opposed to batch normalization) to mitigate potential covariate shift in the transformed input. This is critical considering the residual connections that add previous inputs which can cause the scale of the activations of the attention layer output and the skip connection to vary. Without normalization, the network could experience exploding or vanishing gradients, which can hinder the training process.

Skip or residual connections are a well-established technique that allows easier optimization of deep learning models and enables the training of deeper networks. In the transformer architecture, residual connections are inserted between the incoming transformed input to the encoder and the output of the self-attention module to ensure no information is lost due to processing. It is also inserted between the output of the attention module and input of the feed-forward network. This is done to maintain strong gradients between the input and output of the network's intermediate layer by providing an alternate path for information to be directly passed across the layers without degradation.

Finally, the feed-forward network is a dense fully connected network that applies a non-linear transformation to the inputs. It enhances the model's capacity to learn complex patterns from the input data by processing each position in the input sequence independently although using the same feed forward network. It consists of two linear transformation layers and a ReLU activation between them. While the feed-forward network in the transformer does not perform direct

mixing between different positions in the input sequence, it significantly enhances the transformer's ability to model complex patterns and relationships within each position independently. The feed-forward network introduces nonlinearity, transforms the dimensionality of the data, and, when combined with layer normalization and residual connections, improves training stability and performance.

5-2-4 Segmentation output head

A major alteration to the traditional vision transformer in the EchoViT architecture is the final prediction layer. Most vision transformer models are trained on optical images (ImageNet, ImageNet1000) as classification models. However, the echogram layer tracking problem is a tiered-image segmentation task where dense prediction is required for each pixel and subsequent tracking in along-track axis. Therefore, the final prediction layer is changed from a classification head to a fully convolutional layer that predicts the class of every pixel in the echogram.

To match the size of the output decision matrix to the input image, network hyperparameters such as the dimension of the embedding layer and feedforward layer are chosen to match the dataset's echogram input dimension.

5-2-5 EchoViT model architecture

To best leverage the vision transformer architecture for the echogram layer tracking problem, we investigated multiple patchification schemes to identify those that harness the natural rangeline-to-rangeline sequence in the echogram which will be instrumental for layer tracking along-track.

5-2-5-1 Patch methodologies

The EchoViT architecture is setup as a binary segmentation task which incorporates a pairwise self-attention mechanism to capture spatial correlations between echogram patches. EchoViT is an encoder-only transformer architecture that features a specifically designed binary segmentation output layer tailored to the input's patchifying scheme.

The input to the encoder layer $\mathbf{E}$ is the patched echogram pixels mixed with corresponding learnable positional embedding Z_{pos} as shown in Figure 5-1. Given a grayscale input echogram $\boldsymbol{G} \in \{\mathbb{R}^{N_t \times N_x} : 0 \leq \boldsymbol{G}(m,n) \leq 1\}$ where N_t is the number of fast-time bins and N_x is the number of slow-time bins.

We explored three patching schemes:

1. Fast time patch $P_{ft} \in \mathbb{R}^{N_t \times 1}$ to give patch sequence $Z_{ft} = \boldsymbol{L}[P_{ft1}, P_{ft2}, \dots, P_{fNx}] + Z_{pos\_ft}$

2. Slow time patch $P_{st} \in \mathbb{R}^{1 \times N_x}$ to give patch sequence $Z_{st} = \boldsymbol{L}[P_{st1}, P_{st2}, \dots, P_{sNt}] + Z_{pos\_st}$

3. Cropped patch $P_{cr} \in \mathbb{R}^{b_t \times b_x}$ to give patch sequence $Z_{cr} = \boldsymbol{L}[P_{cr1}, P_{cr2}, \dots, P_{crN}] + Z_{pos\_cr}$

   where b_t and b_x are the respective dimensions of the patch and $N = \frac{N_t N_x}{b_t b_x}$ is the number of cropped patches.

For each scheme, **L** is the linear embedding operation and Z_{pos_*} is the corresponding position encoding for the patch.

The positional embedding for each patchification scheme differs slightly depending on the number of patch tokens created by the scheme. For each scheme, the operation uses the integer position of each token in the input sequence (starting from 0) and maps this integer position to a continuous vector representation. This is done using an embedding matrix whose weights are learned during training that transforms the discrete positional information into a continuous vector space of the same dimension as the echogram patch embeddings. The context-rich outputs Z_{ft}, Z_{st}, Z_{cr} are now the inputs to the self-attention layer in the EchoViT model.

The patching schemes are visually illustrated in Figure 5-2 below.

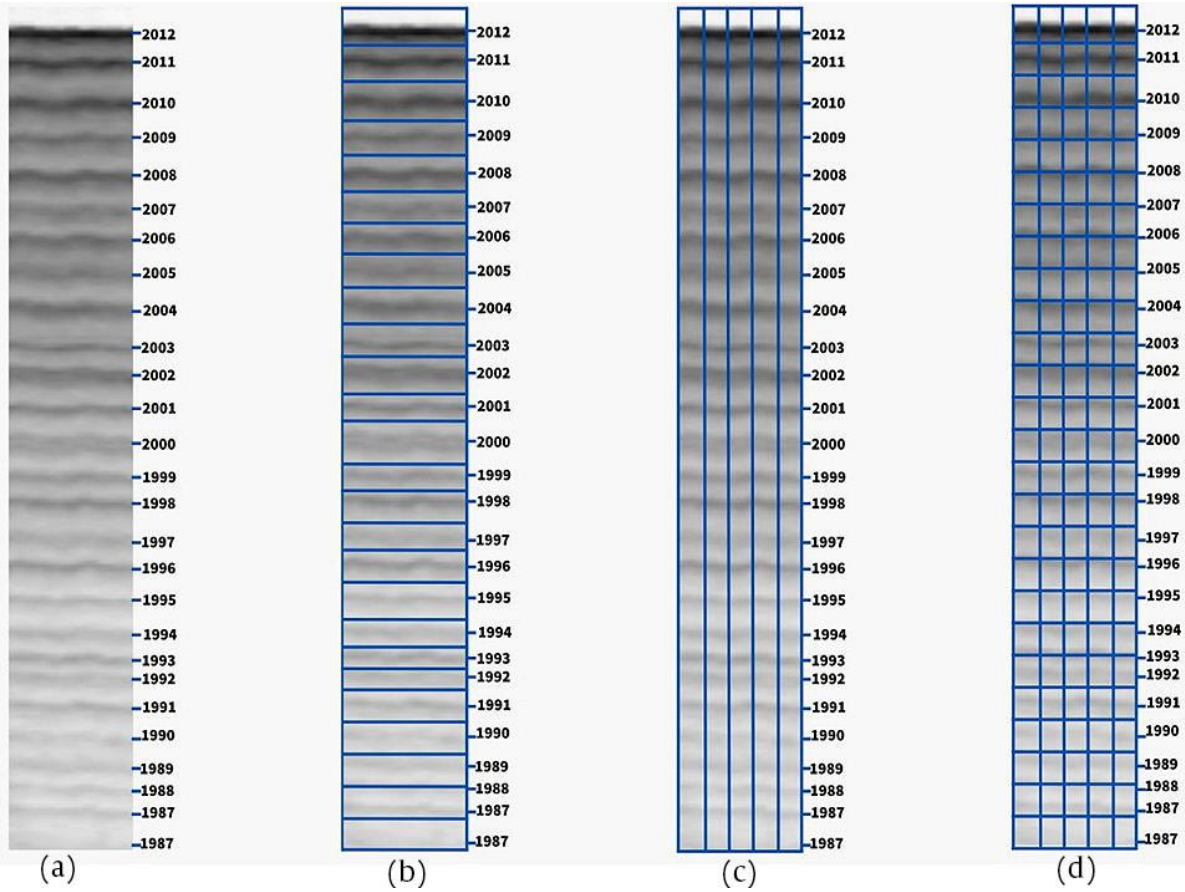


Figure 5-2: Illustration of the patching scheme (a) Echogram with annual layer annotation (b) Slow time patch (c) Fast time patch (d) Crop patch

5-2-5-2 EchoViT training experimental setup

The EchoViT model architecture is a multi-head/multi-output model with each patching scheme as an output head, but all trained together. An additional head that combined the final outputs of the three patchification schemes was added to take advantage of the combination of all the schemes.

The model was trained with the SR_ML_Datasetv1 where each echogram image has a fixed dimension N_t = 1664 and N_x = 256. We set $b_t = b_x = 4$ for the cropped patch scheme.

Consequently, the dimensions of each fast time patch $P_{ft} = 1664 \times 1$, slow time patch $P_{st} = 1 \times 256$, and cropped patch $P_{cr} = 416 \times 64$.

Table 17: *Binary EchoViT training hyperparameters*

Training hyperparameter	**Value**
Batch size	8
Starting learning rate	3e-3
Number of heads	12
Number of Transformer layers	15
Input image shape	1664 X 256
Embedding dimension	1664
MLP dense units	[512, 256]
MLP activation function	GeLU
Convolution head activation function	Softmax
Number of prediction classes	2
Training epochs	200
Learning rate schedule	Reduce LR by a factor of 0.25 on plateau after 10 epochs

The training hyperparameters are listed in Table 17. The training was performed on a Core i9 machine with an RTX A5000 GPU.

5-2-5-3 Qualitative output

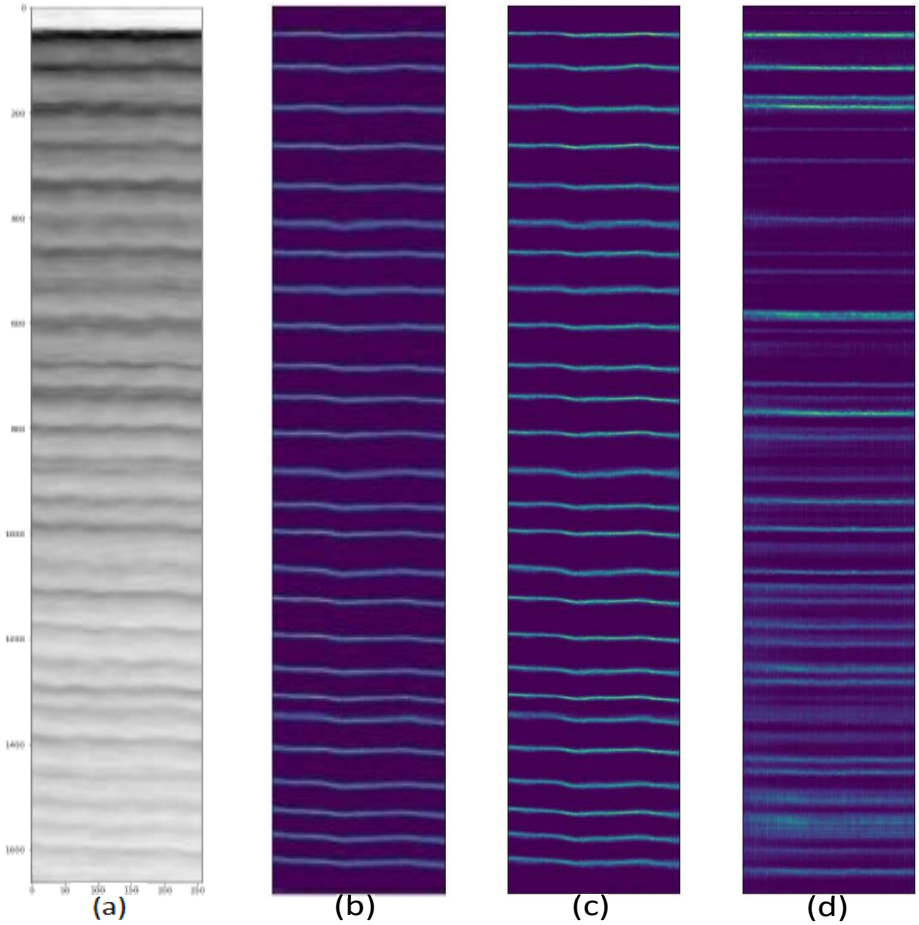


Figure 5-3: EchoViT outputs (a) Echogram image (b) Fast Time patch activation (c) Slow time patch activation (d) Cropped patch activation

5-2-5-4 EchoViT binary output evaluation

Like the outputs of the convolution-based models in Section 4-5-1, the immediate output of the binary EchoViT architectures are also probability heat maps derived from applying the sigmoid activation function to the logits output of the models. Visually inspecting the probability heat maps in Figure 5-3 reveals that the models (except the cropped patch scheme) correctly

distinguish between snow layer pixels and non-layer pixels. As before, the domain of the values in the heat maps is from 0 to 1 requiring a threshold to binarize the output.

Figure 5-3 shows the output of the models on an echogram image from the L1 test set. As seen from the performance of the convolution-based models, these echograms are relatively easy to track. Similarly, for the row and column patching schemes, the EchoViT models correctly identified the layer pixels. However, the cropped patching scheme output shows inferior performance for the same training hyperparameters that other schemes had. This shows that this patching scheme is suboptimal and confirms the hypothesis that arbitrary cropping of the echograms to form patches distorts the naturally occurring sequence formed both in the rows and columns of the echograms. To achieve good tracking performance in the along-track axis, it is best to design architectures that take advantage of this. Hence, the cropped patch scheme is excluded in the model evaluations because the output gets poorer for more challenging echograms.

Next, we evaluate the classification performance of the models to correctly distinguish “layer” and “no-layer” pixels in the echogram. The EchoViTs demonstrate good performance which is indicated by the significant separation between the foreground and the background classes as shown in the histogram plot in Figure 5-4. Due to the superior ability of the EchoViTs to correctly classify the layer pixels, the average of the ODS and OIS threshold values was used to binarize the outputs.

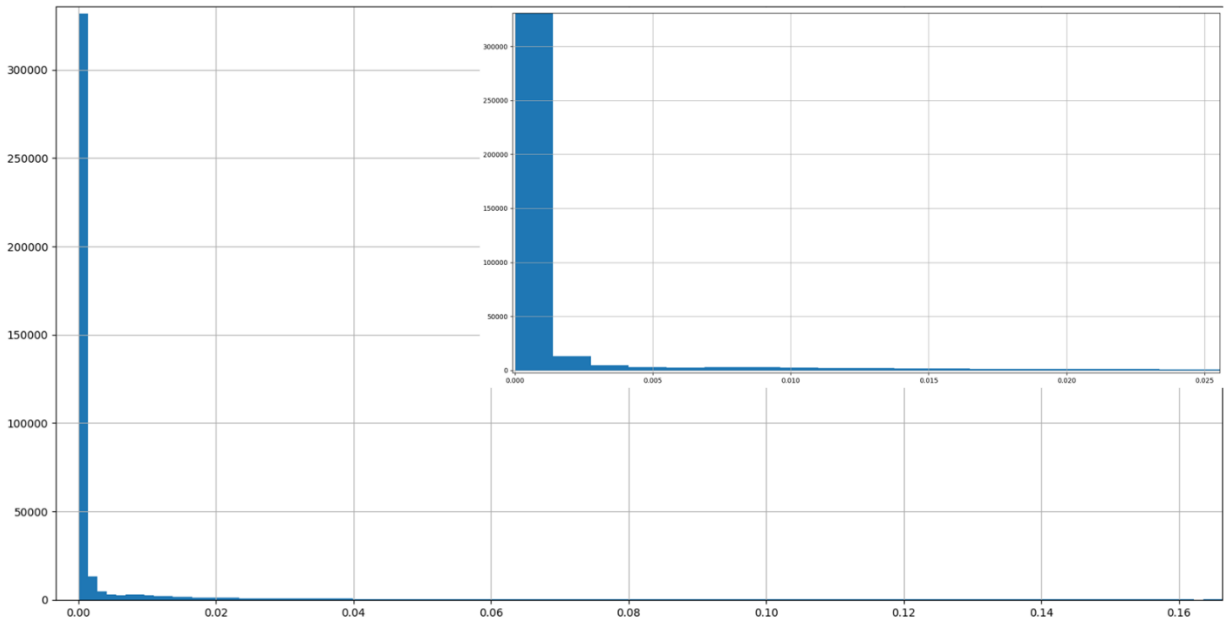


Figure 5-4: Histogram of Fast time patch output showing concentration of background pixels in first three bins. Inset is the zoomed image showing the first few bins.

Table 18 shows the ODS and OIS F1 scores for the different EchoViT architectures. The performance shows signs of improvement over convolution-based models.

Table 18: *Optimal Dataset Scale and Optimal Image Scale F1 scores for EchoViTs*

	Slow time patch	**Fast time patch**	**Combined**
ODS	0.978	0.985	0.969
OIS	0.979	0.983	0.969

Next, the recall, precision, F1 score and SSIM for the combined L1 and L2 test sets is computed and reported in the table below.

Table 19: *Weighted average metrics for the EchoViT models*

Model	Recall	Precision	Accuracy	SSIM	F1
Slow time Patch	0.9778	0.9781	0.9778	0.9585	0.9779
Fast time Patch	0.9780	0.9781	0.9780	0.9588	0.9780
Combined	0.9779	0.9779	0.9779	0.9576	0.9780

Similarly, the performance of the models on each binary class is examined. As reported in Table 20, the performance of all three versions of the EchoViT performs better on both binary classes, particularly the minority layer (1) class.

Table 20: *Recall, precision and F1 score for each of the binary class.*

	Recall			Precision			F1		
	Class 0	**Class 1**	**Unweighted average**	**Class 0**	**Class 1**	**Unweighted average**	**Class 0**	**Class 1**	**Unweighted average**
Slow time patch	0.9886	0.1195	0.5540	0.9889	0.1168	0.5528	0.9887	0.1181	0.5534
Fast time patch	0.9887	0.1199	0.5543	0.9889	0.1182	0.5536	0.9888	0.1191	0.5540
Combined	0.9887	0.1158	0.5523	0.9888	0.1150	0.5519	0.9888	0.1154	0.5521

As was the case with the convolutional-based models, these metrics tell us how well the prediction pixels coincide with the ground truth annotation, however, the thickness of the snow layers is more than one pixel thick which introduces an extra ambiguity in the model evaluation.

The N-pixel accuracies for $N = \{2,5,10\}$, MAE, and missed pixels for the models provide a better metric and are listed in the tables below for the two patchification methods.

5-2-6 EchoViT tracking evaluation

Table 21: *N-pixel accuracies and Mean Absolute error of each model*

	2px	**5px**	**10px**	**MAE**
DeepLab	0.1679	0.6523	0.9237	5.0119
FCN	0.2946	0.8015	0.9638	3.8132
Slow time patch	0.3239	0.7989	0.9605	3.7649
Fast time patch	0.3262	0.7998	0.9612	3.7286
Combined	0.3150	0.8051	0.9600	3.7604

The result in Table 21 for DeepLab, FCN and the three transformer-based models shows that the transformer architecture takes advantage of the sequential and local spatial information between adjacent rangelines to improve not just the classification performance of the model but its along track tracking performance. Compared to the convolution-based models, with FCN having the best performance with MAE = 3.8132, transformer-based models show good consistent tracking performance and a noticeable drop in their mean absolute error compared to the convolution-based models. This is further evidenced in the close to zero intra-layer pixels missed as reported

in Table 22. The intra-layer pixels are important to ensure that there is no gap in the layer's activation map which is crucial for the 1D layer contour extraction in post-processing especially for wet snow zone echograms. These values are for the combined L1 and L2 echograms excluding the L3 set.

Table 22: *Consecutive layer pixel prediction evaluation*

	Whole layer pixels	Intra-layer pixels	Combined Percentage
DeepLab	**1.52%**	**6.81%**	8.32%
FCN	**2.05%**	**2.97%**	5.02%
Slow time patch	**1.43%**	**0.25%**	**1.68%**
Fast time patch	**1.33%**	**0.22%**	**1.55%**
Combined	**2.27%**	**0.23%**	**2.50%**

5-2-6-1 EchoViT tracking performance based on echogram image quality

Table 23: *Mean absolute error (MAE) for L1, L2 and L3*

	L1	**L2**	**L3**
DeepLab	3.956	5.118	41.661
FCN	3.982	3.742	47.084
SlowTime Patch	3.880	3.748	38.042
FastTime Patch	3.858	3.709	36.935
Combined	3.499	3.799	36.837

Table 24: *N_pixel accuracies for each echogram image quality segment*

	L1			L2			L3		
	2px	**5px**	**10px**	**2px**	**5px**	**10px**	**2px**	**5px**	**10px**
SlowTime Patch	0.155	0.769	0.988	0.358	0.803	0.956	0.007	0.077	0.358
FastTime Patch	0.176	0.777	0.990	0.361	0.803	0.957	0.007	0.112	0.381
Combined	0.168	0.828	0.991	0.337	0.801	0.955	0.006	0.094	0.345

We continue the evaluation of the models in terms of the echogram image quality. The mean absolute error for L1, L2 and L3 is computed individually in Table 23. The three models show improved performance particularly on L2 and L3 echogram images with the fast time patch architecture achieving the best performance compared to the convolutional-based architectures and other transformer models. The N-pixel accuracy also shows significant improvement in all the image quality segments including the very poor quality L3 images. Particularly, the 10-pixels accuracies for both L1 and L2 images are above 95% showing that the model is doing well not just on the binary classification task but also on the along-track tracking task.

In Table 25 below, the intra-layer pixels missed in the L1 and L2 segments are enumerated separately. The L2 missed layer pixels dominates the overall combined percentage missed pixels. This again corroborates the impact of image quality on deep learning models. However, compared to the convolutional-based models, the transformer models achieved less than 0.1% intra-layer pixel missed as compared with the FCN (the best convolution-based model) with 2.97% intra-layer pixel missed. This is very significant particularly for echograms with curved layer orientation and poor image quality such as those in the wet snow zone. As discussed in Section 4-6-3-1, missed pixels in the curved regions of snow layer prediction make it difficult

and sometimes impossible to automatically track such layers in the model output. The transformer self-attention mechanism takes advantage of the inherent sequential information in rangelines and range bins to improve the along track tracking thereby reducing the frequency of missed intra-layer pixels.

Table 25: *Consecutive layer pixel prediction evaluation*

	Whole layer pixels			Intra-layer pixels			Combined Percentage
	L1	L2	L1 + L2	L1	L2	L1 + L2	
DeepLab	11 0.35%	248 1.17%	259 **1.52%**	15,215 1.89%	265,010 4.93%	280,225 **6.81%**	8.32%
FCN	16 0.51%	328 1.54%	1.46 **2.05%**	3,883 0.48%	133,847 2.49%	137,730 **2.97%**	5.02%
Slow time patch	**10** **0.32%**	**236** **1.11%**	**246** **1.43%**	**1645** **0.20%**	**2424** **0.05%**	**4,069** **0.25%**	**1.68%**
Fast time patch	**10** **0.32%**	**215** **1.01%**	**225** **1.33%**	**1447** **0.18%**	**2379** **0.04%**	**3,826** **0.22%**	**1.55%**
Combined	**19** **0.44%**	**388** **1.83%**	**407** **2.27%**	**1447** **0.18%**	**2557** **0.05%**	**4,004** **0.23%**	**2.50%**

5-2-7 Generalization evaluation

Similar to earlier developed models, the EchoViT models are tested on a wide range of unlabeled data both from the same campaign year, 2012, and other years with similar radar hardware. As shown in the quantitative metrics, the vision transformer-based models achieve both better classification results as well as along-track tracking performance. For echograms from the dry snow zone, EchoViTs have excellent tracking performance (see Figure **5-5** below) and even on echograms with noticeable along-track fading as in Figure **5-6**.

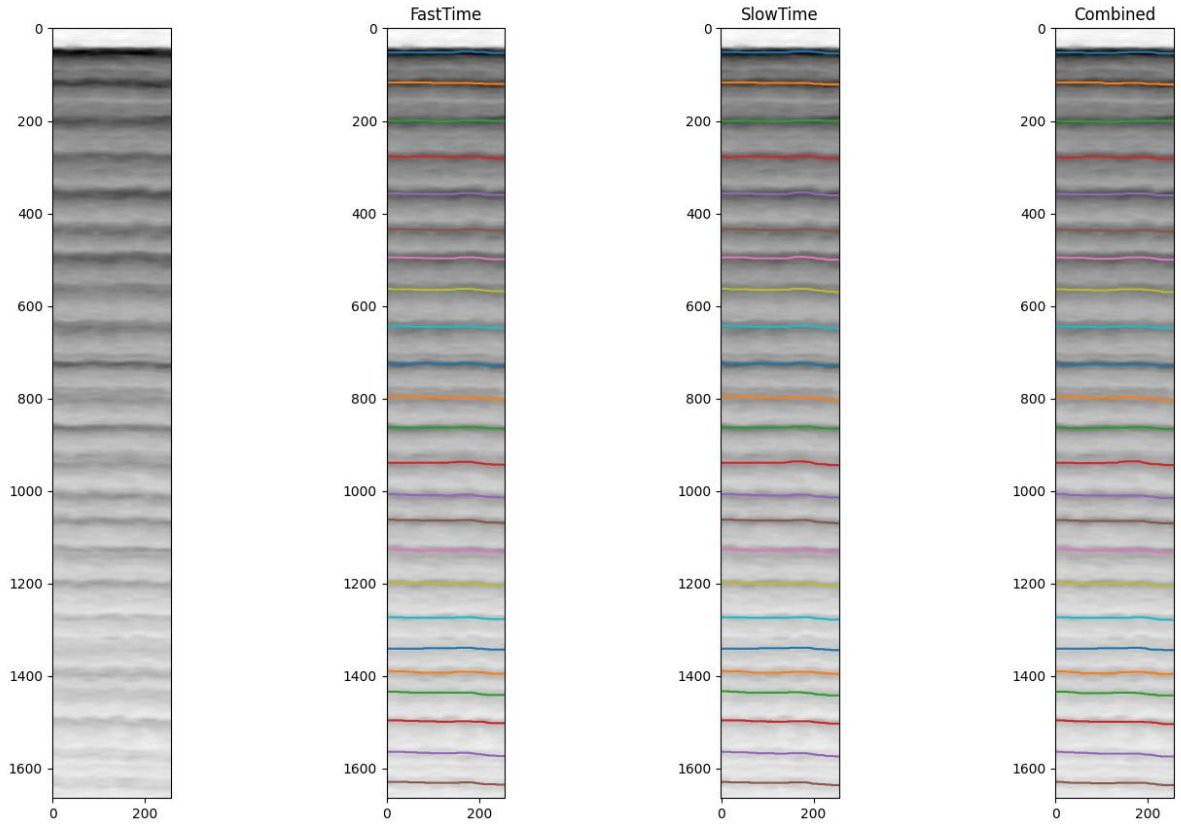


Figure 5-5: Echogram image from dry snow zone overlaid with EchoViT outputs (b) FastTime patch layers (c) SlowTime patch layers (d) Combined layers

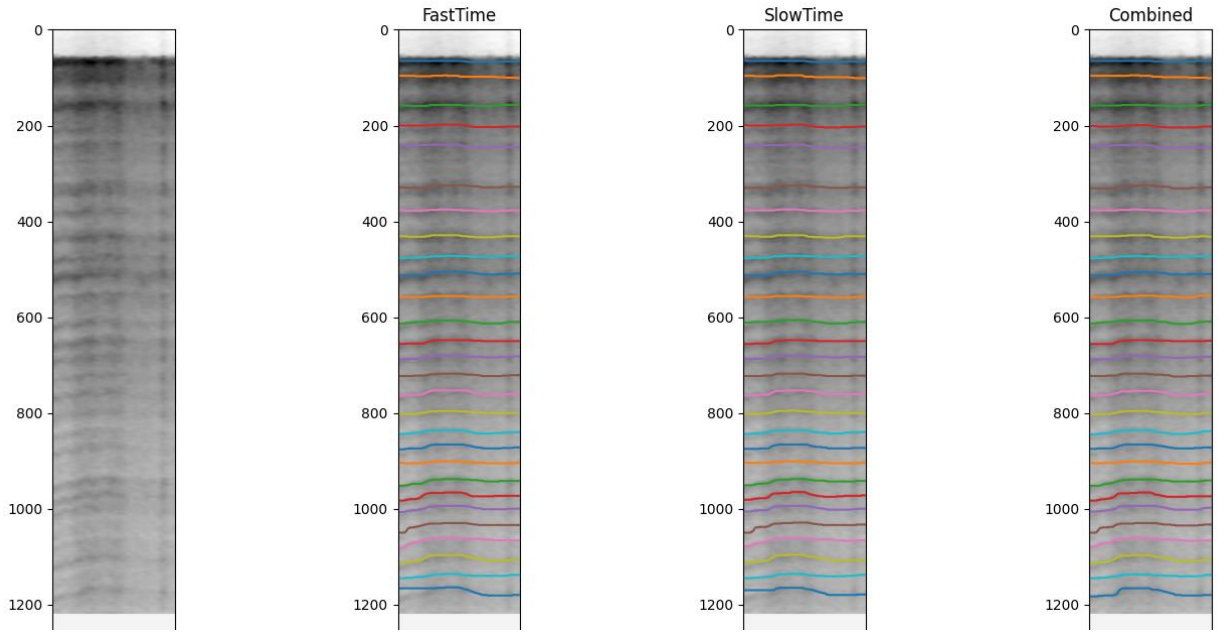


Figure 5-6: Echogram image with some along track fading overlaid with EchoViT outputs (b) FastTime patch layers (c) SlowTime patch layers (d) Combined model

The figures below (Figure **5-7** through Figure 5-10) show qualitative examples of cases where the convolutional-based echograms had a hard time tracking the layers but the transformer-based model successfully tracked the layers.

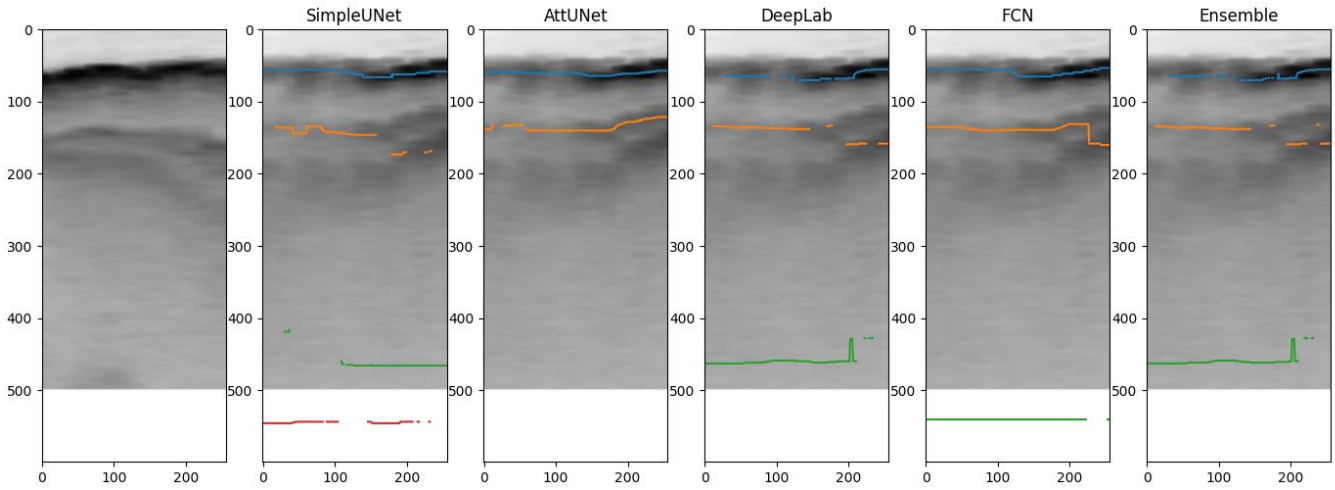


Figure 5-7: (a) Echogram image overlaid with (b) U-Net layers (c) AttentionU-Net layers (d) DeepLab layers (e) FCN layer (f) Ensemble layer

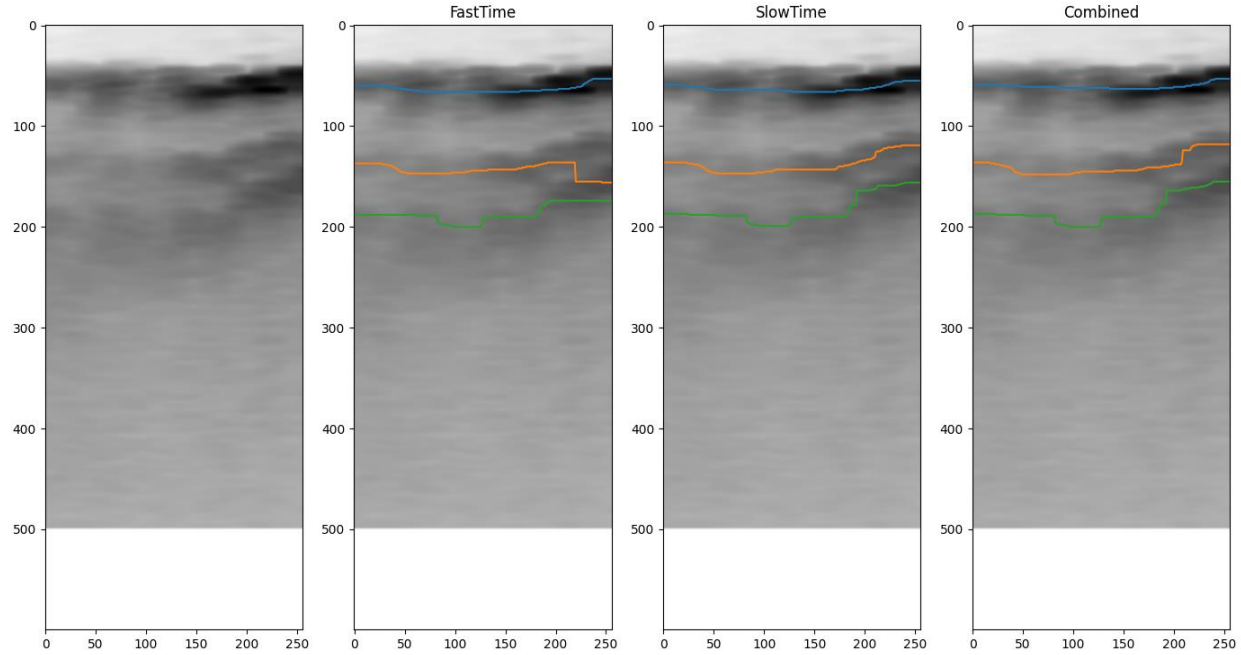


Figure 5-8: Echogram image overlaid with EchoViT outputs (b) FastTime patch layers (c) SlowTime patch layers (d) Combined layers

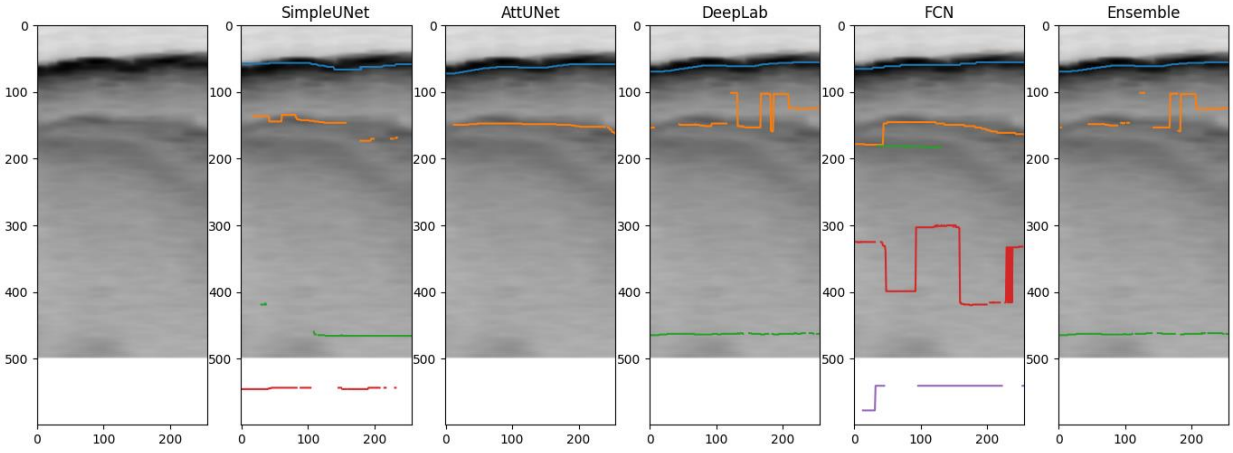


Figure 5-9: Another echogram image overlaid with convolution-based model outputs (b) U-Net layers (c) AttentionU-Net layers (d) DeepLab layers (e) FCN layers (f) Ensemble layers

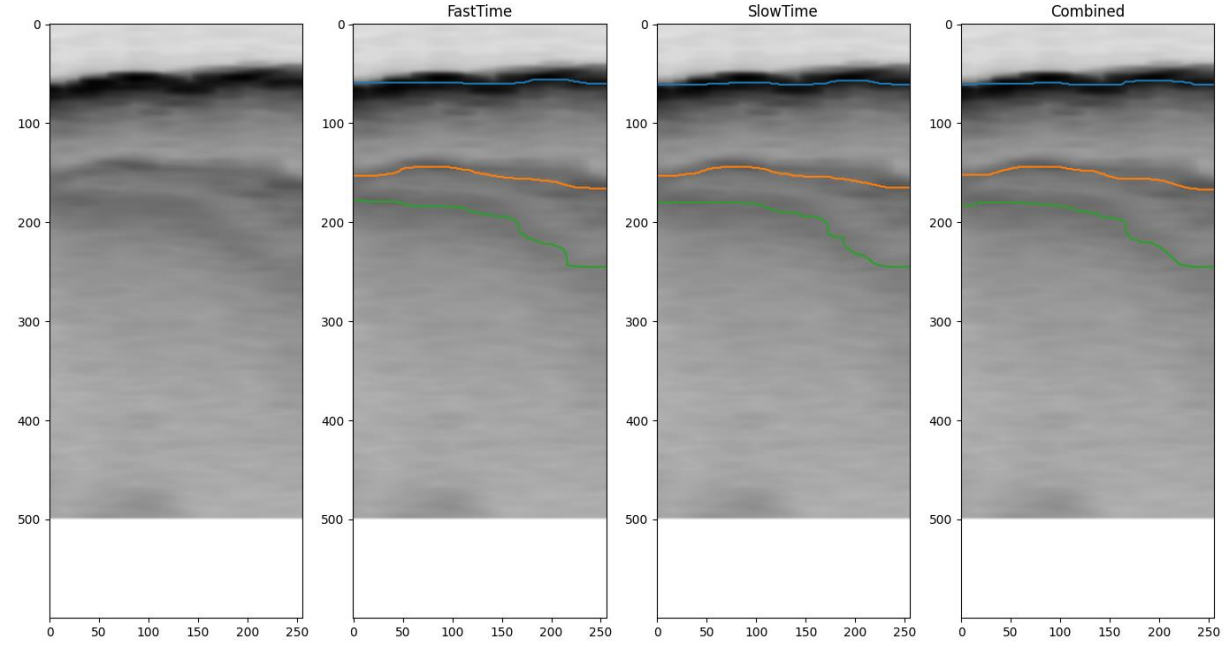


Figure 5-10: The same echogram image overlaid with EchoViT outputs (b) FastTime patch layers (c) SlowTime patch layers (d) Combined layers

5-2-8 Limitations of transformer-based models

While transformer-based models have demonstrated improved tracking performance, certain architectural subtleties introduce limitations that vary in significance depending on the specific application and required tracking precision. In some cases, these limitations may be negligible and can be overlooked; however, in others, they may necessitate exploring alternative architectures.

A key limitation of transformer-based architectures, particularly in the context of echogram layer tracking, is the requirement for a fixed input size, which stems from the predefined patch size necessary at model initialization. Transformers architecture design currently requires that the number of input tokens and the patching scheme be defined in advance, resulting in a need for consistent input dimensions. This means that the echograms fed into the model must have the same number of rangelines and range bins—an often-unrealistic scenario. In optical images, where transformers excel and achieve state-of-the-art performance, each pixel typically lacks a direct physical interpretation, allowing for arbitrary resizing or cropping without distorting the image's meaning.

While it is possible to carefully reshape the echogram images to fit the defined input shape of the transformer model by appropriately upsampling and downsampling as needed, the data interpolation is not entirely perfect since the frequency content of the image is not perfectly bandlimited leading to a loss in the high frequency content of the echogram image and introduction of subtle artifact because of the interpolation. A mild consequence of this is lower along-track resolution of the tracked echogram layers when the interpolation leads to loss of details and blurred edges in the input echogram image. Another implication of reshaping the image size to a fixed dimension is that the slopes of layers change and would no longer be

representative of the true snow accumulation layer slopes. Unless this is accounted for by training with echograms having the same ranges of distortions, this would result in layer orientations that were not encountered by the model during training. Consequently, the model's performance on such accumulation layer slopes during inference will likely be lower.

Chapter 6: METHODOLOGY (4)

6-1 Optimizing convolutional-based model for performance consistency

The improvements made by the transformer-based models indicate that further enhancements can be made to the convolutional-based model to achieve better performance. The transformer-based architecture, being an auto-regressive model, exploits the inherent spatiotemporal structure in the echogram images to outperform our previous convolutional networks. However, convolutional-based architectures are desirable for the echogram layer tracking problem because of their lack of restriction on the echogram input shape meaning that they can take any shape of input echogram and are not constrained by what the model was trained with during inference. This eliminates the need for resizing, which could otherwise degrade along-track resolution. Such flexibility, without performance loss, is critical, especially since many echogram images outside the training set differ significantly in size from the fixed dimensions used during model training.

Typically, echograms in a deep learning training set are best set to be of the same shape to facilitate the mini batch grouping for parameter optimization during training. When this is done, some architectures such as the transformer-based models, require that the test images are first resized to match the training set dimensions during inference. After obtaining the prediction, the results are subsequently reshaped back to the original dimensions of the input image. This is often not a challenge in many image domain problems but as earlier described, the pixel resolution of echogram images has a direct relationship with the physical phenomenon being measured. Downsampling and upsampling of the echogram images results in the loss of fine details of the accumulation layers particularly in the along-track axis. As such, convolutional-based models are particularly attractive because they perform well on echogram images outside

the training set as well as future echograms that will be created from upcoming scientific missions.

Consequently, a new convolutional-based model is designed to improve on the weaknesses of earlier models. Particularly, earlier convolutional-based models, despite their good performance on dry snow zone echograms with good image quality, were noticed to suffer significant performance dip on echograms with poorer image quality and less obvious delineation of the snow layers. The model's tracking performance deteriorates significantly when these two conditions coexist:

1. Weaker backscatter from the snow layer relative to the background clutter due to the presence of meltwater in the mapped region, and,
2. Curving accumulation layer coincides with signal power fading in the echogram rangelines.

The coincidence of these two effects causes gaps in the thinned layer predictions which are difficult to merge and combine to form the desired output layer 1D contours. To tackle this problem, a multi-pronged approach is adopted to improve multiple sections of the entire echogram layer tracking pipeline starting from the pre-training image processing to the training architecture, and the layer contour post-processing extraction. Notably, a new convolutional-based architecture that builds on the strength of earlier models but enforces autoregressive cohesion between adjacent layer pixels using a composite loss function is designed. Also, a larger dataset with improved echogram diversity is created and used to train the model.

6-2 Training pipeline modifications

6-2-1 Echogram image pre-processing

The performance of all the earlier trained models proves that the performance is hinged, not surprisingly, on the input echogram image quality. Hence, efforts were made to improve the training set echogram image quality as much as possible but to also increase the diversity in the training set to include more lower quality images to learn with. Of the several image pre-processing steps listed in Section 4-1, echogram filtering, averaging and detrending were noticed to considerably affect the tracking performance of the models.

6-2-1-1 Detrending

Rangeline power detrending was also noticed to have a significant effect on the model tracking performance. As explained earlier, deeper layers often have less SNR compared to shallower layers, and the lower SNR makes it difficult to distinguish the layer peaks from noise at these depths. As a result, some models failed to correctly classify and track such layers. In the new training scheme, rangeline detrending is used to make it so peak pixel intensities are more comparable regardless of the depth of the layer. To increase the training set diversity, a few training echograms were left undetrended to give the model the opportunity to learn from these kind of echogram images too. Specifically, copies of detrended and undetrended echogram images of the 18th and 28th April 2012 (segment 01) flightlines were added to the new training set. Both segments include a total of 860 undetrended images added to the training set which includes echogram data of along-track lengths of 2 km and 5 km.

6-2-1-2 Fast time filtering

The length of the fast-time filter is critical in handling noise in the rangeline data. A short filter does not fully eliminate low-amplitude noisy peaks, while longer filters (21 pixels or more) may blur weak layers along with removing noise. The ideal filter length varies across echograms and requires further experimentation for optimal performance. In the new training set, the filter length was randomly varied between 3, 5 and 7 pixels for the echograms in the training set. This further contributes to the diversity in the training set.

6-2-1-3 Deep learning-based image filtering and denoising

As established from earlier models and experiments, a deep learning model performs better when the input echogram image has good image quality with crisp delineation between the noise background and the layer pixels. However, despite the application of traditional signal processing methods such as detrending, coherent and incoherent averaging, etc., the image quality is still limited for some echograms. Worse still, the incorrect application of some processing due to wrong hyperparameter choices, such as fast time filter length, can remove vital information from the data which will negatively affect the model tracking performance. This proves that adaptive hyperparameters are needed to best condition the echogram for optimal deep learning processing.

To address this challenge, deep learning-based denoisers and filters are explored to learn and adaptively remove noise from the echogram image. This is critical for echogram image filtering and denoising where the noise characteristics are dynamic, making it difficult to choose a fitting filter type or tune an optimal filter length.

There exists a number of classical and deep learning-based filters in the literature [90], [91], [92], [93] which have been reported to surpass traditional signal processing approaches. In this work, we adopt the deep plug-and-play image restoration (DPIR) [94] learning-based image denoising algorithm.

The algorithm combines model based and learning-based paradigms, consolidating the strength of both approaches to train a variable splitting (half-quadratic splitting) CNN-based denoiser that employs both U-Net and ResNet in its denoiser architecture. Concretely, the grayscale image deep CNN denoiser termed (DRU-Net) has a U-Net backbone that consists of four image scales that each have skip connections between the downscaled and transposed convolved image pair. Starting with the initial convolutional layer with N = 64 channels, the number of channels is doubled for each successive scale with the last layer having N = 256 channels. The convolution layers' weights discard the bias weight term, and each residual block only contains one ReLU activation function – an architecture choice that has been noticed to improve deep denoiser super-resolution.

As shown in the network diagram in Figure **5-9**, the model has 4 stages of 2x2 decimation in the encoder path and corresponding stages in the decoder path. The Residual Block is made up of 4 consecutive convolutions with similar number of features and a skip connection addition between the initial input and the residual block output. This is done to increase the representation power of the network. This is then followed by sequential standard convolution blocks (SConv) in the encoder and transposed convolution blocks (TConv) in the decoder with intermediate downsampling.

Although the denoiser is not applied on the training echograms, it is included in the inference pipeline to handle the different noise levels that may exist in the test echogram, particularly other echograms outside the dataset. It adaptively removes Gaussian and non-Gaussian noise in the image. The output of the denoiser, with improved PSNR, is then passed to the model to predict the layers.

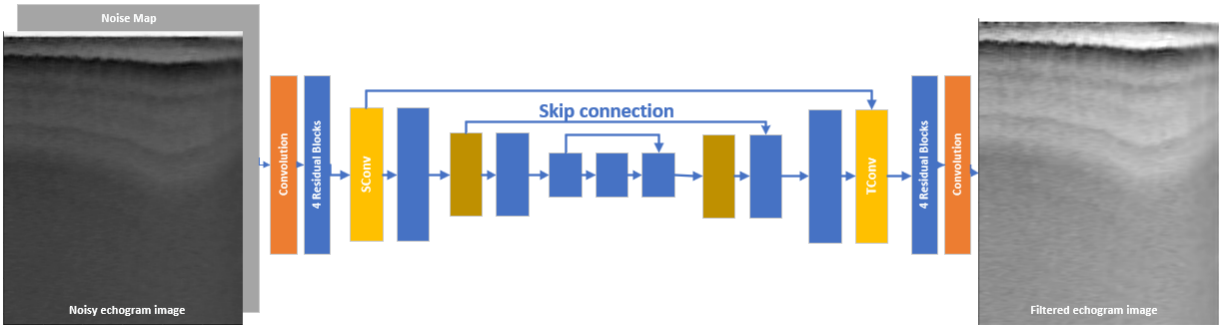


Figure 6-1: Block diagram showing overview of the DPIR denoiser architecture

6-2-1-4 Vesselness filter

Vesselness filters [95], [96], [97] are designed to enhance tubular structures in images that traditional filters have limited performance on due to the curved orientation of the feature of interest in the image. Originally designed for medical images of blood vessels, airways and other tubular anatomical structures that require specialized filters, echogram images with arcuate snow layers share similar feature orientations. Owing to the curved and river-like nature of the feature of interest in the image, the application of classical boxcar filters that operate on adjacent pixels has limited performance and can sometimes be detrimental especially in the case of a weak snow layer transversing only a few pixels out of the overlaid filter length.

These filters utilize the Hessian matrix which contains second-order partial derivatives of the image intensity to analyze the local shape and structure of the image features. As such, they can

identify pixels that contribute to elongated features like snow layers in the echogram image while ignoring other non-vessel features that correspond to noise.

The Hessian matrix $\boldsymbol{H}$ at a point (x, y) in a 2D image is defined as

$$H = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

where I_{xx}, I_{xy}, I_{yy} are the second-order partial derivatives of the image intensity I. The eigenvalues λ_1 and λ_2 of the Hessian matrix provide insight into the local curvature of the image intensity surface. These eigenvalues are critical for identifying tubular structures, with different magnitudes indicating structural characteristics and their orientation. This is important since snow layer structures typically go from left-to-right (horizontally orientated) while noise features such as image and processing artifacts, rangeline fading, often result in vertically oriented features.

In this work, we adopt the Meijering vessel filter [98] designed to enhance curvilinear structures while being robust to noise and variations in the vessel diameter or intensity discontinuities along the travel path. The filter enhances neurite-like features even in low contrast neighborhoods such as is the case with diffused layer backscatter in wet snow zone echograms. It achieves this using multi-scale analysis where the image is analyzed at multiple scales by convolving with a Gaussian derivative of different standard deviations. This creates a scale-space representation that helps in capturing structures of different pixel widths. The standard deviation σ of the Gaussian kernel is varied across the multiple scales provided which allows the filter to identify consistent tubular features across the scale-space created.

As with the DRU-Net denoiser, the Meijering filter is used only in the inference pipeline. The aim is to train the segmentation models with a variety of poor and good echogram images to

improve their robustness but perform inference on cleaned images. The Meijer filter also has the drawback of introducing vertical ridge-like artifacts, but the sequential application of the above-mentioned processing steps combined with the segmentation model can get rid of these.

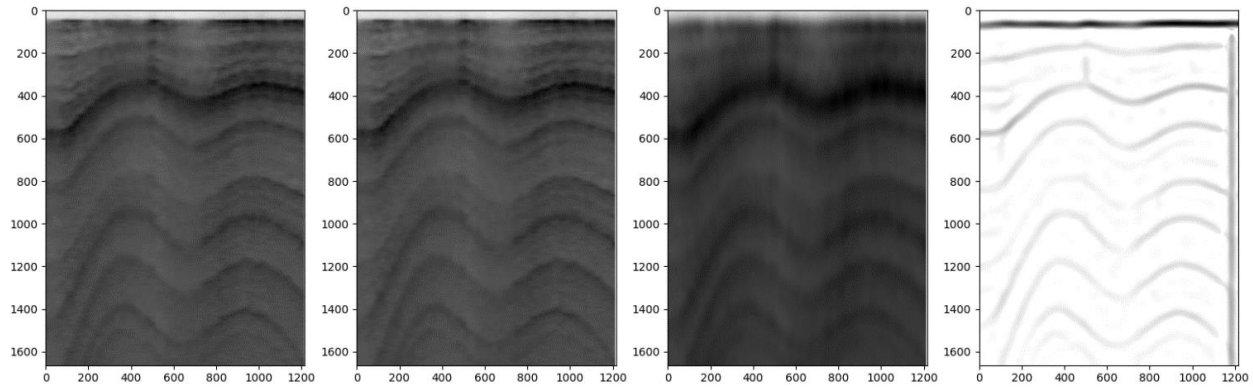


Figure 6-2: (a) Echogram with curvilinear layers (b) mild boxcar filtered echogram (N = 5) (c) aggressive boxcar filtered echogram (N=21) (d) Meijer filter output

6-2-2 Increasing echogram image diversity in the training and test set

To improve the performance of the new model, echograms with specific characteristics were added to the training set. The goal was to simultaneously improve the model's performance both on wet snow zone echograms and dry snow echograms with very deep layers e.g. echograms with more than 50 snow layers. To achieve this, more echograms with few layers and undulating orientation were manually traced and added to the training set. Particularly since the first set of models had challenges predicting wet snow zone echograms partially due to the sparsity of these kinds of echograms, increasing the population of such echograms in the training set becomes imperative. Also, simulated echograms with few layers and varying layer curvature representing echograms from the wet snow zone were added to the training and testing datasets.

Also, to improve the model performance on echograms with very deep layers such as those from the summit of Greenland which may have more than hundred (100) layers with the lower layers

being very faint, closely packed and difficult to distinguish, more of such echograms were manually tracked and added to the training set. This updated training pipeline was deployed using the active learning paradigm to create the SR_ML_Dataset version 2 with 50,000 training, validation and testing set examples.

6-2-3 Improved binary segmentation training label

In the earlier training set, the output ground truth was generated with single-pixel thick layers. However, this makes training more difficult since the layers themselves are multiple pixels thick. Assigning a single pixel as the layer is a harder training objective and increases the susceptibility of the model output to having undesired gaps in the tracked layers since it is being trained to have just one pixel for each layer. Therefore, we relax the training criteria by including the neighboring 3 pixels above and below the candidate layer pixel so that the output ground truth is generated with 7-pixel thick layers. Figure **6-3** below illustrates the difference between the previous binary ground truth and the new ground truth. By relaxing the layer thickness, the model learns to focus on the band of pixels around a layer, instead of just a single pixel, thereby reducing the possibility of a disconnected tracked layer even when some of the echogram rangelines suffer noticeable fading. It is important to note that although multiple pixels are assigned higher probability values relative to the no-layer pixels, a single pixel is eventually chosen as the layer pixel in post-processing. The selected pixel is the one with the maximum probability in the model's output probability map and the one that makes an 8-pixel connection with its neighboring pixels. This is further explained and illustrated in the post-processing set up.

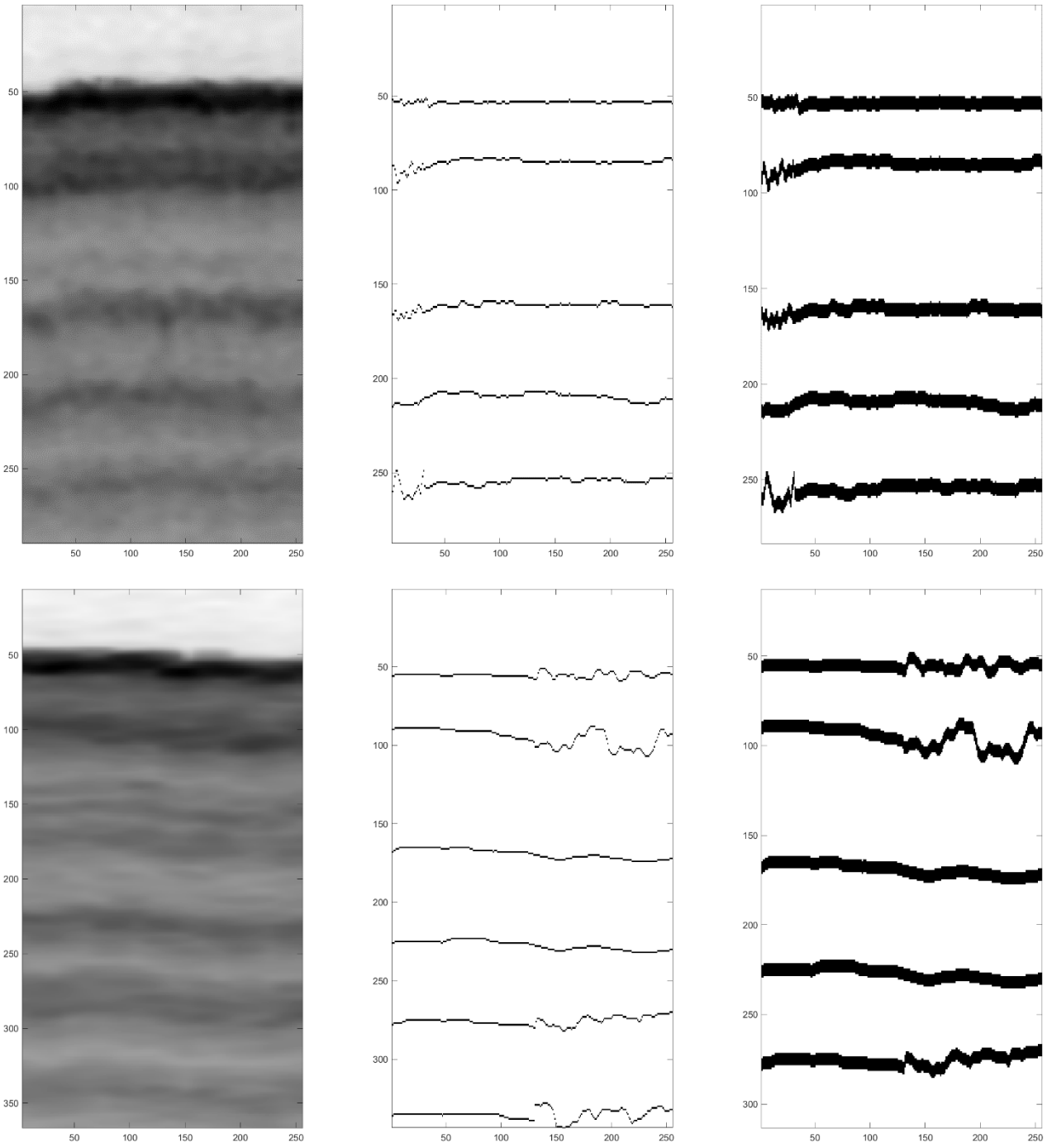


Figure 6-3: Examples of (left) zoomed echogram images with (middle) previous binary GT labels and (right) new multi-pixel binary GT labels

6-2-4 Custom hybrid model cost function

To train the new architecture, a hybrid cost function is developed to improve on the strength of earlier models while also ensuring connectivity between adjacent pixels. The cost function (also known as objective function or loss function) measures how well a deep learning model's prediction matches with the actual data. It quantifies the error between the predicted values and the true values. The selected optimizer takes the output of the cost function to minimize its value by adjusting the model's weight parameters during training. It is an important part of training a deep learning model because it dictates how well a model is trained and its success on test and out-of-sample data. An incorrectly defined cost function could either lead to the model not training well, the model not learning expected features from the data or worse, the model overfitting the training data.

6-2-4-1 Binary cross-entropy (BCE) loss

Some of the earlier trained models were trained with the most common supervised deep learning classification cost function – binary cross-entropy defined as

$$L_{BCE} = \frac{-1}{n} \sum_{i=1}^{n} [\, y_i \, log(\hat{y}_i) + (1 - y_i) \, log(1 - \hat{y}_i) \,] \tag{15}$$

where y_i is the actual label (0 or 1) and $\hat{y}_i$ is the predicted probability of the instance being in class 1.

The BCE cost function has the good attribute of producing a probabilistic interpretation of the model's output when used with a sigmoid activation layer. Its reputation on binary tasks is earned because of its non-negative loss and convexity which makes it well behaved for gradient

optimization. However, it might be too simplistic particularly for highly imbalanced classes like the binary segmentation echogram layer tracking task.

6-2-4-2 Binary focal cross entropy loss

In some problems, particularly those with highly imbalanced classes, the binary cross entropy loss can reach a plateau where it becomes insensitive to further changes in the model's parameters during training. This can make it difficult to continue optimizing the model parameters to improve performance on the minority class. The binary focal cross entropy loss [99] is designed to address the issue of class imbalance.

It introduces a modulating factor to the BCE loss giving less weight to the well-classified examples and focusing on the hard-to-classify ones.

$$L_{FL} = \frac{-1}{n} \sum_{i=1}^{n} \alpha_t (1 - p_{t,i})^{\gamma} \, log(p_{t,i}) \tag{16}$$

where p_t is the predicted probability for the true class, α_t is the balancing factor parameter that helps balance the importance of positive vs negative examples. γ is the focusing parameter that adjusts the rate at which easy examples are down weighted. When $\gamma = 0$, focal loss is equivalent to BCE loss but as γ increases, the focusing effect is intensified, putting more emphasis on the hard examples.

However, the introduction of new hyperparameters in focal loss introduces a new challenge too. Hyperparameter tuning of the focusing parameter γ which differs from one dataset to another adds a layer of complexity since its value dictates the effectiveness of the focal loss. Although further experimentation is needed, the values $\gamma = 2$ and $\alpha = 0.25$ were used for training.

6-2-4-3 Intersection over union (IoU) loss

While BCE focuses on pixel-wise accuracy and focal loss on class imbalance, there is still the need to ensure patch-level segmentation quality to avoid potential gaps in the snow layer output segmentation map. To achieve this, the IoU loss, also known as Jaccard index loss, is added to the custom composite loss function.

The IoU loss is popular in segmentation tasks for evaluating the overlap between the predicted segmentation mask and the actual segmentation label. Hence, it is useful for evaluating the accuracy of the model's predicted masks and how well the model's predicted snow layer boundaries align.

The IOU loss is derived from the IoU metric with the IoU metric defined as the ratio of the intersection of the prediction and ground truth to their union

$$IoU \text{ metric } = \frac{|A \cap B|}{|A \cup B|} \tag{17}$$

The IoU loss is computed as $c = 1 - IoU\ metric$ to approximate a convex, optimizable loss function. A small smoothing factor is sometimes added to prevent instability when the intersection or union is zero and to ensure that the loss is differentiable.

Unlike BCE and focal loss that operate solely on pixel-per-pixel basis, the IoU loss emphasizes the overall shape and boundary accuracy of the model's segmentation map which is crucial for predicting the curved boundaries of snow layers in echogram images.

6-2-4-4 Structural Similarity Index Measure (SSIM) loss

Structural Similarity Index Measure (SSIM) is a perceptual metric used to measure the similarity between two images. Unlike traditional loss functions like BCE, focal loss and IoU loss which focus on pixel-level accuracy and overlap, SSIM performs image-level evaluation by estimating the perceptual quality and structural information of the echogram images. It considers subtleties like luminance, contrast, and other structural differences.

The SSIM between two images x and y is computed as

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)\,(2\sigma_{xy} + C_2)}{(\mu_x{}^2 + \mu_y{}^2 + C_1)\,(\sigma_x{}^2 + \sigma_y{}^2 + C_2)} \quad (18)$$

where:

- μ_x and μ_y are the mean intensities of x and y;
- $\sigma_x{}^2$ and $\sigma_y{}^2$ are the variances of x and y and σ_{xy} is the covariance of x and y;
- C_1 and C_2 are small constants to stabilize the division with a weak denominator.

Like the IoU loss, SSIM loss is computed by estimating $1 - SSIM$ to ensure that maximizing the SSIM is equivalent to minimizing the SSIM loss. Including SSIM loss in the composite cost function forces the model to consider local patterns of intensities that have strong interdependencies such as the layer edges and boundaries which may be missed by BCE, Focal loss or IoU loss. SSIM is also more robust to changes in illumination and contrast in pixel values which is what happens in rangeline fading scenarios.

6-2-4-5 Custom 8-pixel connectivity loss

As earlier discussed, a major challenge noticed in previous model outputs was the gaps and disjoints that occurred in the tracking of a snow layer when the echogram rangelines suffer severe fading. To specifically address this, the custom 8-pixel connectivity loss is included in the composite loss function. The connectivity loss function encourages the predicted segmentation mask to have smooth and connected regions by penalizing discrepancies between a pixel and its neighbor in all 8 possible directions. This ensures that neighboring pixels in the mask are consistent with each other since structural continuity of the prediction regions is critical for tracking the snow layers in the echograms.

Given the model output segmentation map $\hat{y}$, where each element $y\ (i,j)$ represents the predicted value (probability or confidence score) at pixel location (i,j), the 8-pixel connectivity loss L_{conn} is defined as the sum of the absolute differences between the predicted value $\hat{y}\ (i,j)$ and its neighboring pixels in all 8 directions:

$$L_{conn}\big(\hat{y}\ (i,j)\big)\ = \frac{1}{N}\sum_{(dx,dy)\ \in\ \text{shifts}} |\hat{y}\ (\boldsymbol{i},\boldsymbol{j})\ -\ y(\boldsymbol{i}\ +dx\ ,\boldsymbol{j}\ +\ dy)| \tag{19}$$

where **shifts** = $\{\ (-1,-1),(-1,0),(-1,1),(0,-1),(0,1),(1,-1),(1,0),(1,1)\ \}$ represents the 8 neighboring pixels around each pixel (i,j). **N** = 8 is the total number of neighboring pixels.

In summary, the composite loss function used to train the model is defined below:

$$L_{comp} = L_{BCE}\ +\ L_{FL}\ +\ L_{IoU}\ +\ L_{SSIM}\ +\ L_{conn} \tag{20}$$

The composite loss which combines Binary Cross-Entropy (BCE) Loss, Focal Loss, Intersection over Union (IoU) Loss, Structural Similarity Index Measure (SSIM) Loss, and Connectivity Loss leverages their individual strengths to collectively enhance segmentation model performance.

Binary Cross-Entropy (BCE) Loss is pivotal for precise pixel-wise classification, ensuring each pixel in the segmentation mask is correctly classified based on predicted probabilities. Focal Loss supplements this by addressing class imbalance, concentrating more on challenging classes to achieve balanced predictions. Intersection over Union (IoU) Loss measures overlap between predicted and ground truth masks, crucial for accurate boundary delineation. It emphasizes the quality of segmentation outputs, ensuring predicted regions align well with ground truth masks.

Structural Similarity Index Measure (SSIM) Loss evaluates perceptual quality by considering local patterns, luminance, and contrast in images. It enhances visual similarity and structural consistency in segmentation outputs, aligning closely with human perception while the Connectivity Loss enforces 8-pixel connectivity, promoting smooth transitions and consistency between neighboring pixels. This maintains structural integrity in segmentation masks, reducing artifacts and discontinuities.

Collectively, these loss functions provide a comprehensive evaluation of segmentation quality. They address pixel-level accuracy, class balance, overlap accuracy, perceptual quality, and structural consistency. This holistic approach enhances the model's robustness, improves generalization across diverse datasets, and ultimately enhances the segmentation performance of the new convolutional model on the challenging echogram layer tracking problem.

6-2-5 The EchoRefine Model

A new convolutional-based architecture was designed to build on the success of the fully convolutional network (FCN) that achieved better performance compared to the other models tested. The FCN, although sharing a very similar architecture with U-Net and AttentionU-Net, achieved better tracking performance. This is likely due to the gradual upscaling in the decoder but significantly the absence of any connection between the encoder-decoder counterpart layers. The echogram layer tracking problem is an edge/contour detection problem which should be learned in the earlier layers of a deep neural network. Although more experimentation is needed to confirm the hypothesis, it is possible that the plain and unassuming architecture of the FCN forces it to truly understand the underlying layering process which leads to its better performance.

Since the challenge of the earlier developed convolutional models was their inability to consistently track the layers especially along the contour edges and layer boundary in the arcuate layer regions with faded rangeline power, we developed the **EchoRefine** model which introduces an extra refine module that takes the front "FCN-like" model output as its input combined with supplementary information of neighboring rangelines to further refine the layer edges and correctly delineate the layer boundaries. It must be noted, however, that the model is trained end-to-end.

The EchoRefine model has a simple yet effective architecture with 2 stages – first is a shortened FCN-like network at the front and an additional refinement network to gradually refine the coarse output of the initial network to create fine and continuous boundary predictions. The EchoRefine takes inspiration from the Deep Coarse-to-Fine family of models that takes the probability maps of the earlier model but builds an auxiliary detached model to improve the

earlier prediction. Unlike many of these models with detached component models, the EchoRefine is a consolidated model that exists as a single unit. Examples of such models include RefineNet [100] that uses long-range residual connections to preserve information during the downsampling path, SRM [101] that uses cascaded stages of intermediate segmentation maps that are refined to produce high resolution outputs. Other coarse-to-fine models include C2S [102], DHSNet [103] and DGRL [104].

Concretely, the EchoRefine model has an FCN-like encoder-decoder front-model which contains 3 stages of sandwiched convolution, batch normalization, ReLU activation function with dropout and a 2x2 max pool layer. These convolutional layers share a similar architecture with the first 3 stages of the notable ResNet-34 [105] which facilitates feature extraction from the ResNet 34 model pretrained on ImageNet images [106]. The 3 encoder stages are followed by matching-in-size Conv2DTranspose upsampling units in the decoder to restore the feature map to the original input shape.

Like the residual refinement module in [107], the EchoRefine's refinement network uses an encoder-decoder architecture which has a single convolution layer in each of the 4-stage encoder layers with 64 filters each. An intermediate bridge stage is inserted prior to the decoder stages which uses small 3 x 3 receptive field filters to focus the refined segmentation map. The refinement network decoder performs non-overlapping progressive upsampling in equivalent 4-stages before inserting a prediction head to create the final output of the EchoRefine model.

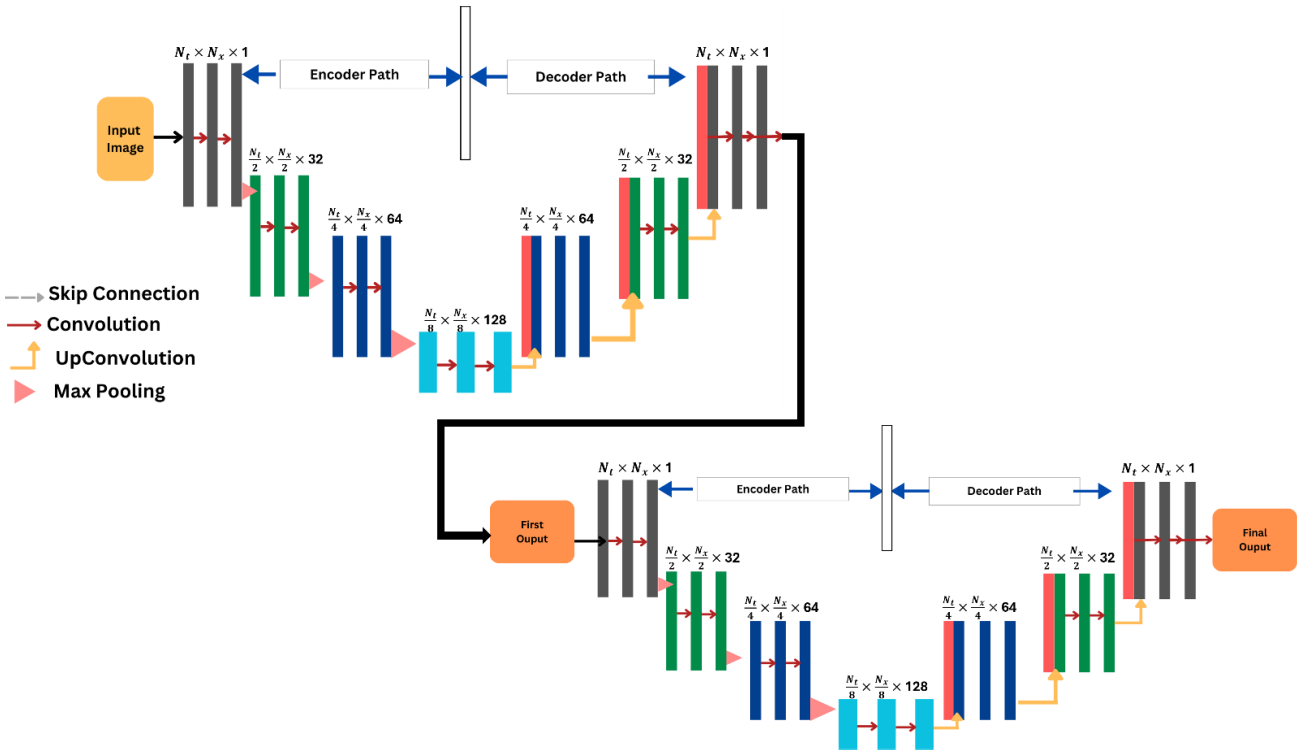


Figure 6-4: EchoRefine Model architecture diagram

6-2-6 The 1-D layer contour extraction routine

Before going on to discuss the EchoRefine tracking performance, we first discuss the steps followed to convert the segmentation model output to individual 1-D layer contours for each identified snow layer. The 1-D layer contour extraction can be summarized into the following sequential steps:

(a) Pixel thresholding

(b) Individual layer patch identification

(c) Thinned layer range bin value extraction

First, the binary segmentation model produces a probability map with values between 0 and 1 because of the sigmoid activation output on the final layer (See Figure **6-5**). Typically, the values of the non-layer pixels are low and close to zero while the layer pixels are higher. However, the exact value of the layer pixels varies depending on the model architecture and training effectiveness. The separation between the layer pixels and the non-layer pixels probabilities is a measure of the model's performance and ability to distinguish between both classes. When there is a clear separation (large value) between both classes, it is easy to find a threshold to classify each pixel into the correct class. This is often the case for most models when the echogram has good image quality like in Figure **6-5**. However, this is not always true particularly for poorer quality echogram images with deeper snow layers in echograms that are closer to each other and sometimes have poor SNR. Also critical is the ability of the model to distinguish between artifact pixels (and ignore them) but still identify layer pixels that have low values due to echogram image artifacts or rangeline power fading in the echogram image.

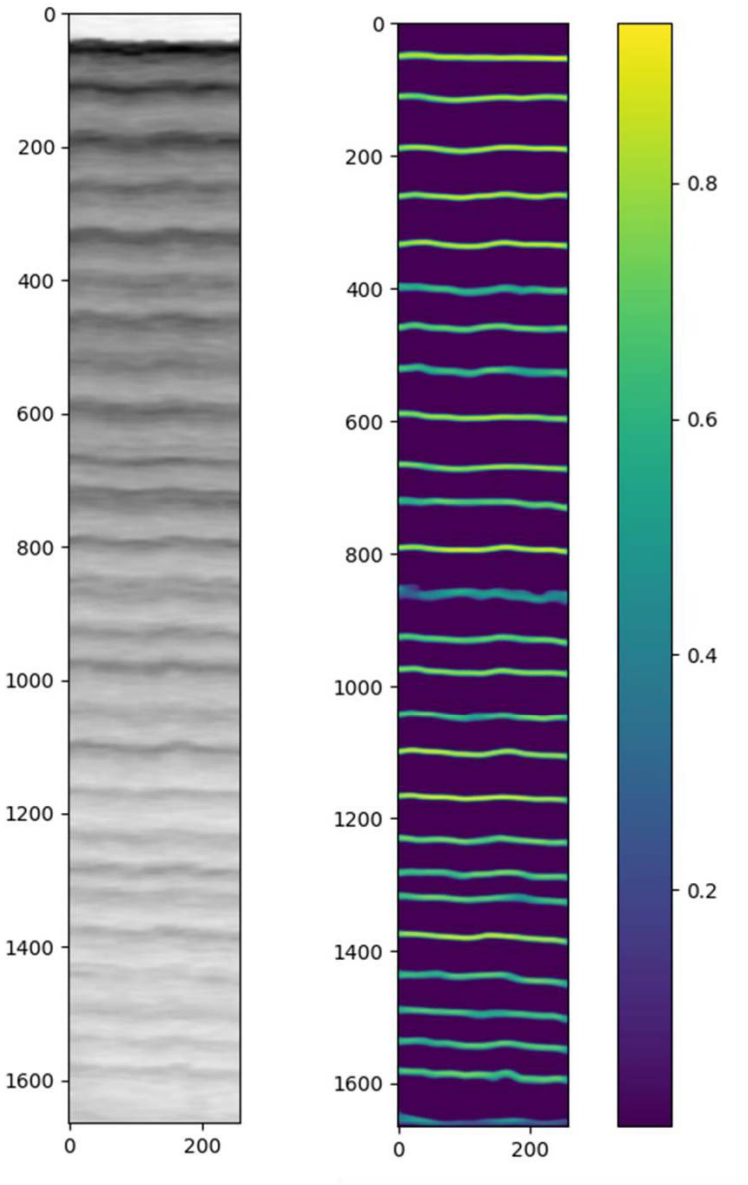


Figure 6-5: (a) Example good quality echogram image and (b) EchoRefine model output

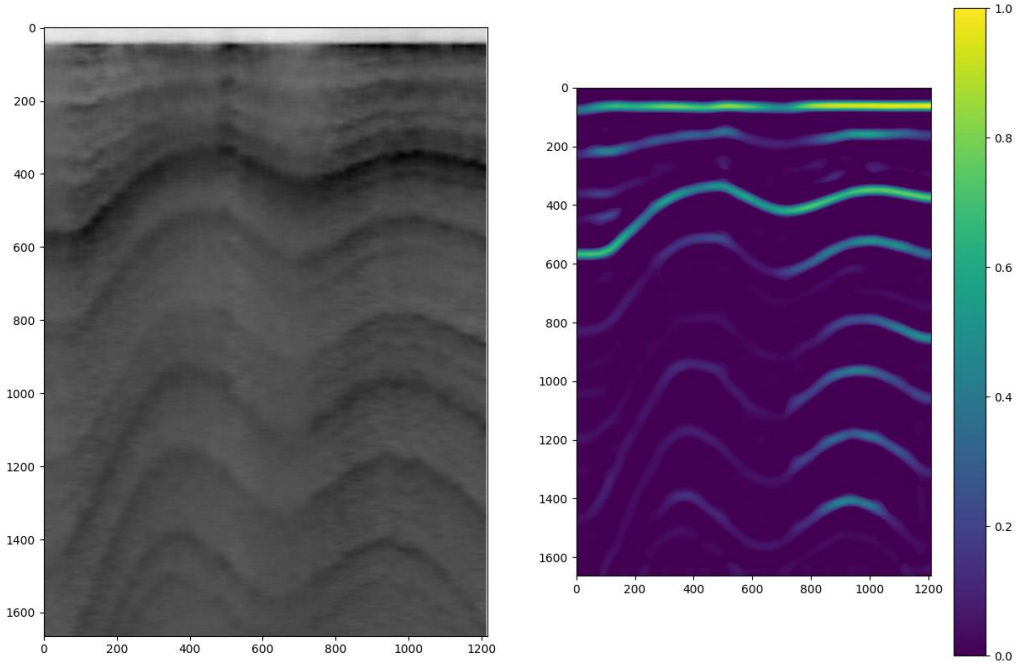


Figure 6-6: (a) Example poorer quality echogram image and (b) EchoRefine model output

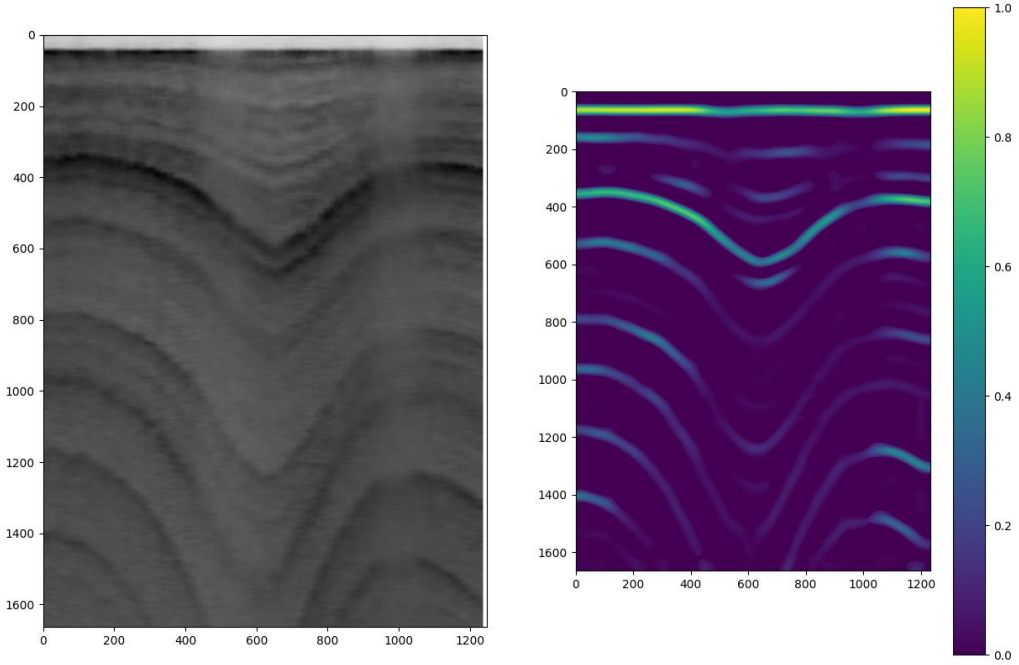


Figure 6-7: (a) Example poorer quality echogram image and (b) EchoRefine model output

6-2-6-1 Pixel thresholding

To extract the 1-D contours of each identified snow layer from the segmentation output map, the first step is to set a threshold to distinguish between layer pixels and background (no-layer) pixels. Using the Optimal Dataset Scale and Optimal Image Scale, one can get a threshold estimate for the training dataset and individual image respectively, but this computation is expensive since it performs a grid search, and the returned value does not always give optimal performance. Alternatively, we employ a simple adaptive search for each test image. This is done by estimating an optimal value that separates both classes based on the distribution of detrended echogram pixels. For echogram images with distinct layers, obtaining a good threshold value is easy as shown in Figure **6-8** below. The first histogram bin typically corresponds to most of the background class so choosing the fourth histogram bin is often sufficient to discriminate both classes. However, finding an optimal threshold value for images with poorer quality requires extra care. The current implementation uses an affine combination of the 90^{th} percentile and the Otsu threshold[108]. Concretely, all pixel values below the threshold are set to zero while those greater are set to one to give the binarized output as in Figure **6-8**(d).

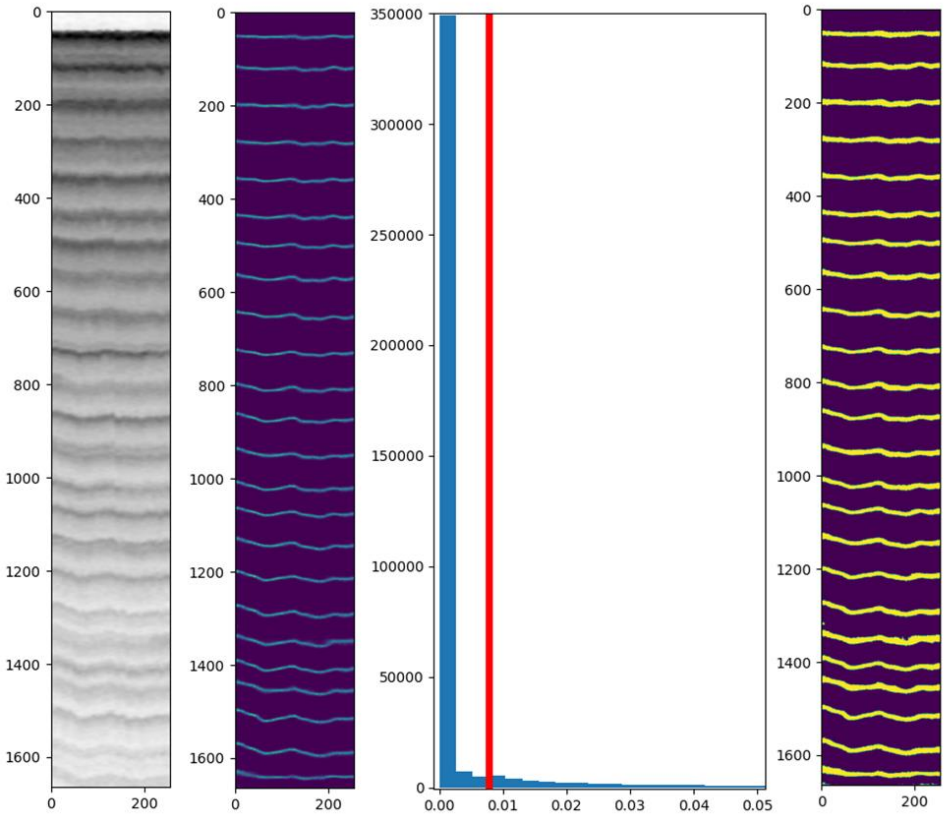


Figure 6-8: (a) Example echogram image with distinct snow layers (b) EchoViT FastTime-patch model output (c) Zoomed model output distribution with the threshold in red (d) binarized output

6-2-6-2 Individual layer patch identification

From Figure **6-8**, the output of the thresholding stage is a binarized map of the model prediction. The region (patch) for each layer can be clearly identified although it is represented by a few pixels. Also, although each layer is visually identified, it needs to be thinned to a single pixel per rangeline and extracted as a 1-D contour. The output in Figure **6-8**d is a binary map that only identifies layer pixels from non-layer pixels. To extract the 1-D layer contour, it is important to first identify all the pixels that contribute to each layer uniquely. This is done using mathematical morphological tools specifically using **connected components analysis**. The theoretical fundamentals are introduced in [109] and further developed in [110], [111]**.**

Connected components analysis (CCA) is a fundamental concept in image processing and computer vision that involves the identification of contiguous regions in a binary image. A connected component is a maximal set of foreground pixels such that there exists a path between any pair of pixels within the set. Formally, let $G = (V, E)$ be an undirected graph where each vertex $\mathbf{v} \in \mathbf{V}$ corresponds to a pixel, and each edge $\mathbf{e} \in \mathbf{E}$ represent the adjacency between the pixels. A connected component $\boldsymbol{G}$ is a subgraph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the supergraph.

Building from the foundational work of Carlo and Luigi in [112], [113] to works in [114], [115],[116], several algorithms have been proposed for the efficient computation of connected components. Classical approaches include depth-first search (DFS) and breadth-first search (BFS) which traverse the image from top left to bottom right to label each component. More advanced methods such as Union-Find offer improved performance for larger images and datasets.

Due to the chronological deposition age of each snow layer, each layer is distinct and ideally, all the pixels in the echogram image corresponding to each layer ought to be 8-pixel connected. Hence, components analysis was adopted given its relative simplicity to implement particularly for small to medium-sized echogram images.

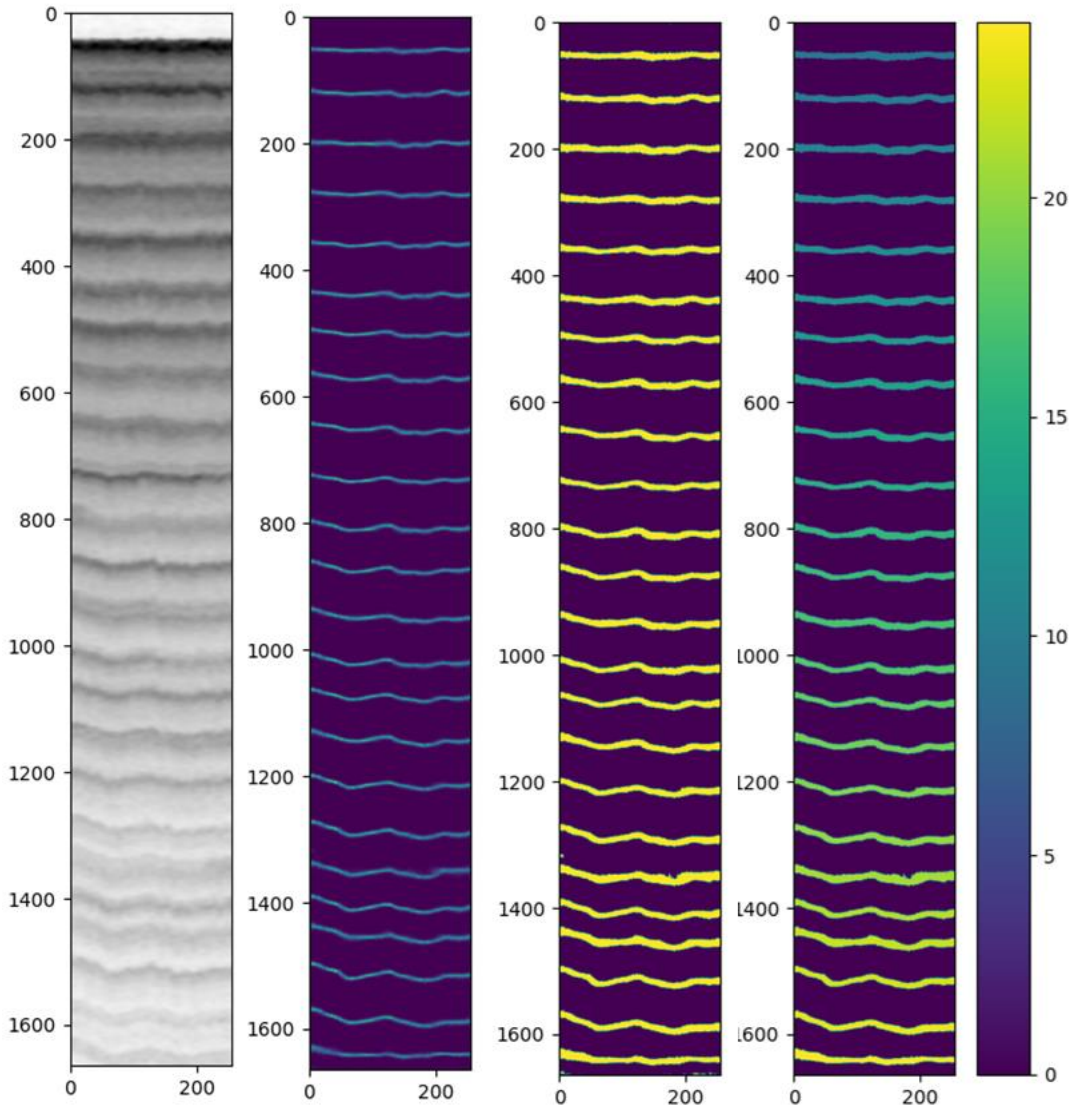


Figure 6-9: (a) Example echogram image with distinct snow layers (b) EchoViT FastTime-patch model output (c) binarized output (d) Individual layer patch identified using CCA

In Figure **6-9**, the CCA outputs a 2D map in Figure **6-9**d that is of similar dimension with the input echogram image. Also, all its 24 layers have been uniquely identified based on the segmentation map predicted by the EchoViT deep learning model in Figure **6-9**b. All the pixels in the binarized map (Figure **6-9**c) belonging to a specific layer has now been assigned the correct label of the layer in Figure **6-9**d. Now, this output can then be iterated over, one layer at a

time, to retrieve the single pixel corresponding to the layer's peak and its range bin values as a 1-D layer contour.

6-2-6-3 Thinned layer range bin 1-D layer contour extraction

Given the output of CCA with each layer uniquely identified, the multi-pixel wide layer needs to be thinned into a single pixel corresponding to the snow layer peaks. This process is often termed "skeletonization" in the computer vision community. This is achieved using dynamic programming technique to track the peaks across all the rangelines for each layer.

Concretely, a Greedy Search with a connectivity constraint is performed using both the identified layer in the CCA output and the probability map output from the deep learning model. Although the probability map from the deep learning model returns more than one pixel for each rangeline of each layer, the peak probability value along the rangelines of the layer often corresponds to the true index of the single pixel layer peak. The Greedy Search, therefore, goes through the rangelines for each layer to identify the layer peaks that best achieve 8-pixel connectivity between the layer peak pixels. As a result, the peak probability may occasionally be disregarded if it does not connect with the adjacent pixel, and instead, the nearest probability index is used. This step is repeated starting from the first identified layer by CCA to the last layer and the resulting output is a 2D matrix with each row being the 1-D range bin index which traces out the contour of each layer as shown in Figure **6-10**e.

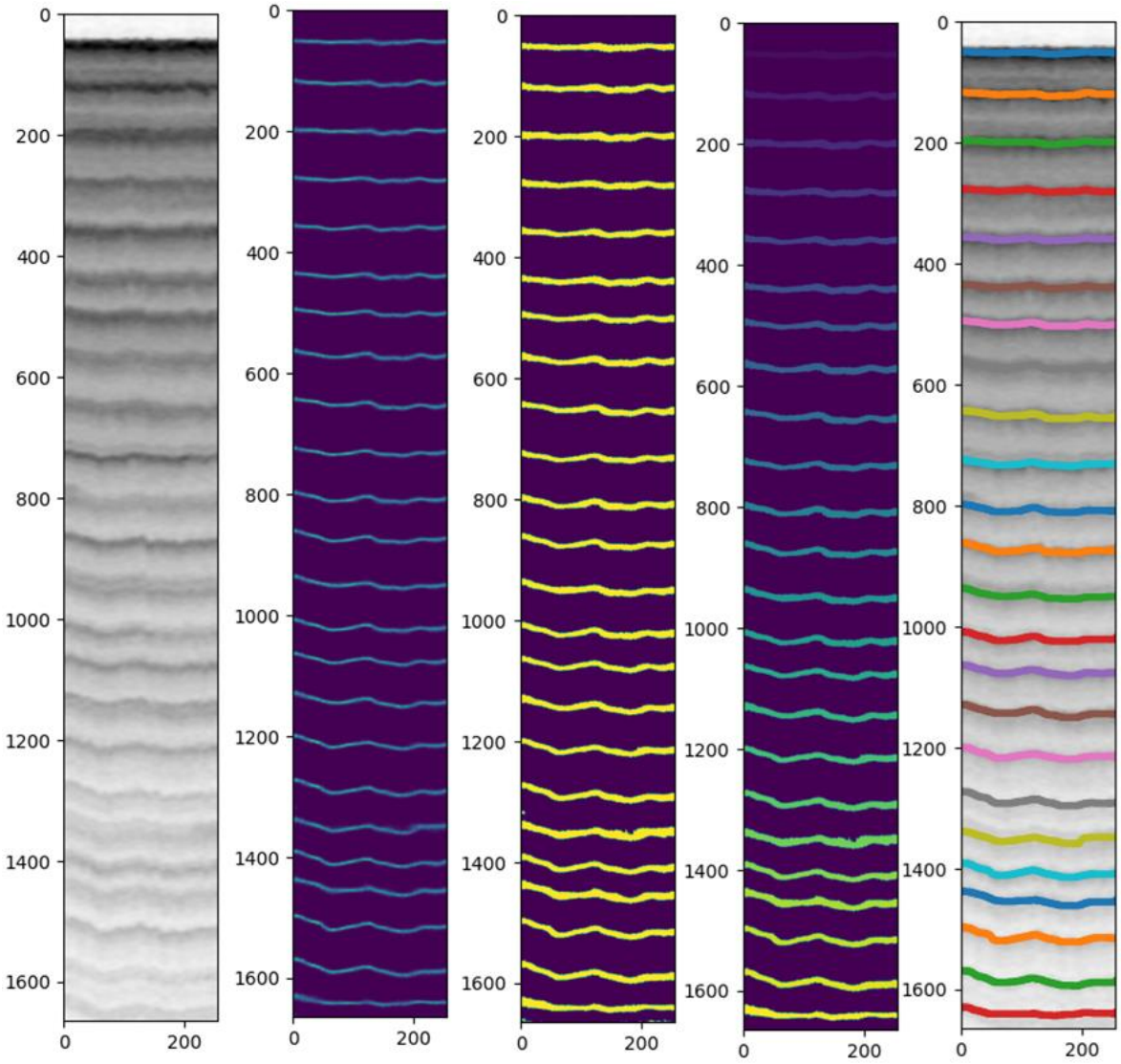


Figure 6-10: (a) Echogram image with distinct snow layers (b) FastTime-patch model output (c) binarized output (d) Individual layer patch identified using CCA (e) Extracted and plotted 1D layer contours

6-2-6-4 Addressing issues due to echogram imperfections before 1-D layer contour extraction

The description of the thresholding and CCA as described in Figure **6-9** is straightforward for the ideal case where the snow layers are distinct and there is clear separation between consecutive layers. While this is true for most echograms, it is not always the case especially for poor quality echogram images. Consequently, additional processing is required to deal with the ensuing complications.

Despite the simplicity of CCA, it suffers two potential drawbacks that can degrade the accuracy of the layer 1-D contour extraction. First, CCA requires *strict* 8-pixel connectivity between adjacent rangeline pixels in a layer without allowing as little as a single pixel gap. When there exists a gap of a pixel or more between the pixels that cluster to form a layer, CCA treats them as separate and distinct components by assigning different labels to them. As will be described, this can be difficult to resolve in some cases e.g. steeply curved contour edges. Secondly, if multiple layers are mistakenly merged due to imprecise threshold to separate them, CCA reports these layers as one. If the incorrectly merged layers are not separated, the layer contour extraction process will report them as a single 1-D contour. This often results in a confusing 1D layer contour that is a jumbled mixture of the range bins from two or more constituent layers. Both drawbacks significantly compromise the accuracy of layer tracking and completely hinder the ability to correctly estimate snow accumulation in the echogram because of the incorrect layer 1D contours estimated.

Finding the optimal threshold to separate the two classes into a binarized map is challenging in echograms with less distinct snow layer boundaries coupled with significant rangeline fading

effects. In many of these cases, it is impossible to find such a threshold without merging two or more neighboring layers as one. This leads to the challenge of optimizing for a threshold that satisfies the dual condition of ensuring strict connectivity between adjacent rangeline pixels of a layer while preventing two or more neighboring layers from merging as one.

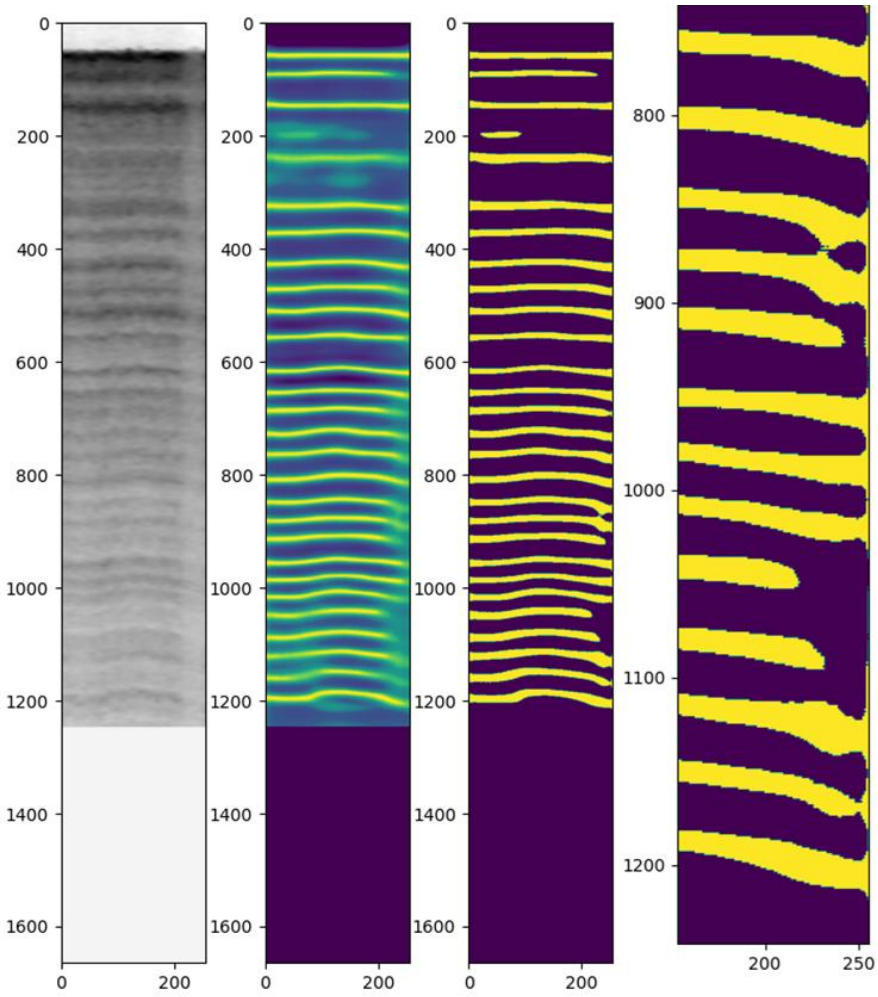


Figure 6-11: (a) Echogram image with fading effects towards the right edge (b) FCN model output (c) binarized output (d) magnified section of the binarized image to illustrate layer merging

Typically, one or both conditions are violated in snow layer tracking in echograms with image artifacts and poor image quality. While trying to find the lowest permissible threshold to avoid layer gaps in a section of the echogram (usually the deeper layers), shallower layers with a lower

noise floor merge. While the use of adaptive thresholds (such as adaptive mean or adaptive Gaussian thresholds) for each echogram pixel alleviates the “layer merging” challenge, it is not totally eradicated.

Separating the merged layers is achieved by computing the statistics of the connected components elements returned by CCA such as the number of columns and rows for each detected component. Merged layers have the characteristics of spanning more rows than the average of a single layer component but having gaps between consecutive rows for some of its rangelines. Once this condition is detected, a custom grid-based row-wise dynamic programming technique can be used to identify the joining pixels and separate the merged layers.

To correct the layer gap issue for dry snow zone echograms with disjointed, but relatively horizontal layers, connected component elements can be done without many complications. Since the layers are flat, despite the gaps, the separated elements (that ideally should be seen as one element) are merged based on a heuristic rule. However, this is usually complicated for closely spaced disjointed curvy layers in echogram images where the disjointed pieces are known to be difficult to order in a 2D plane. This is why rangeline fading coinciding with arcuate layer curves are often difficult to deal with. Although more sophisticated methods such as greedy beam search exist to correct the layer gap issues, a preliminary application of this algorithm indicates that the challenge persists for some images.

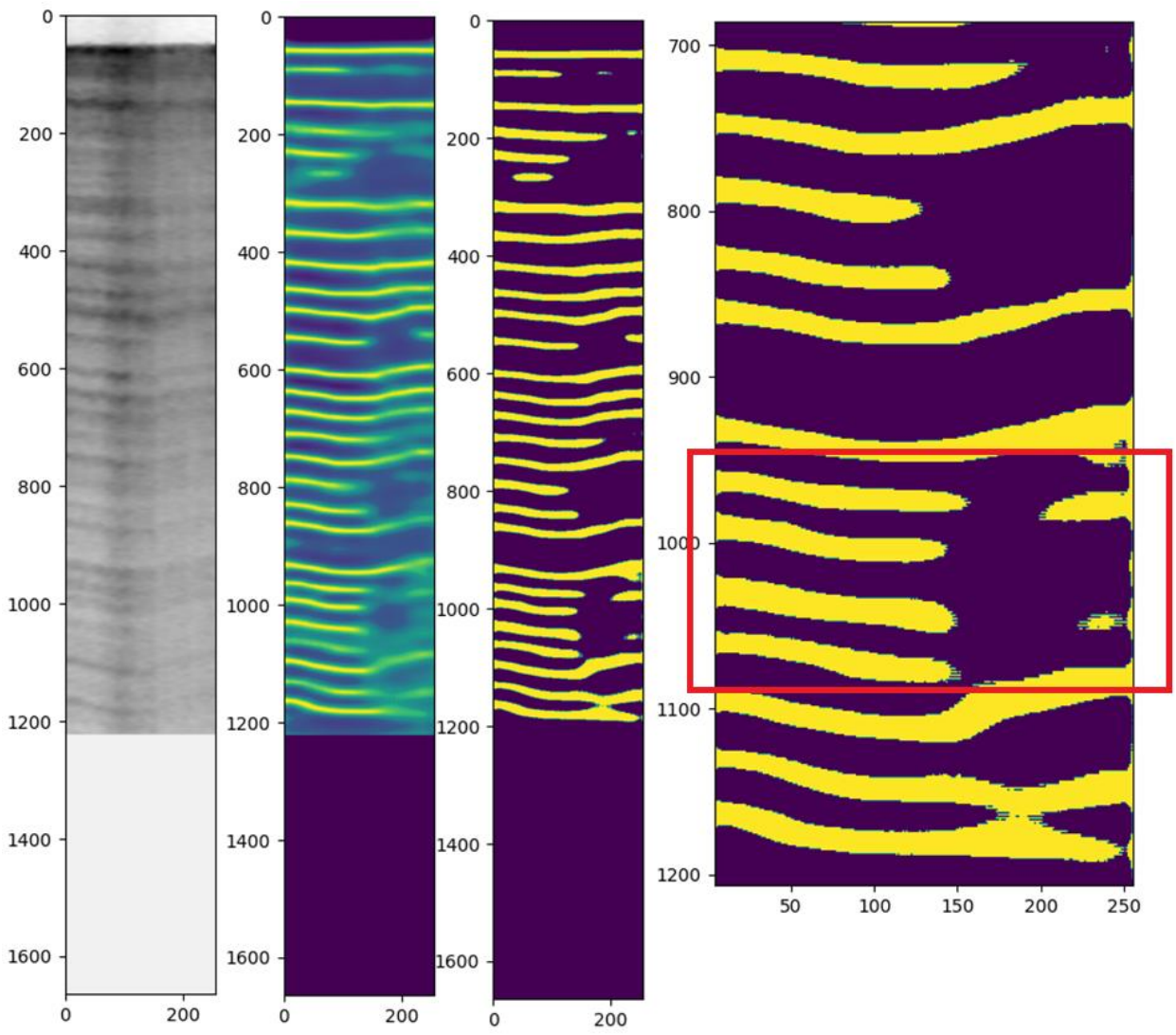


Figure 6-12: (a) Example echogram image with severe fading effects towards the right edge (b) FCN model output (c) Binarized output (d) Zoomed binarized output

As seen in the binarized echogram image above, both complications of layer gap and layer merging occurs for the deeper layers. While the layer merge issue can be resolved fairly quickly, automating the assignment of the disjointed layers as a single connected component requires further attention. This is because the length and orientation of the "broken" pieces of a layer varies widely across the dataset, a simple heuristic of correctly combining such disjointed parts remain elusive.

6-2-7 EchoRefine tracking evaluation

6-2-7-1 Qualitative model performance

Before the quantitative analysis of the EchoRefine model, we first illustrate the improvements achieved by the model over earlier models.

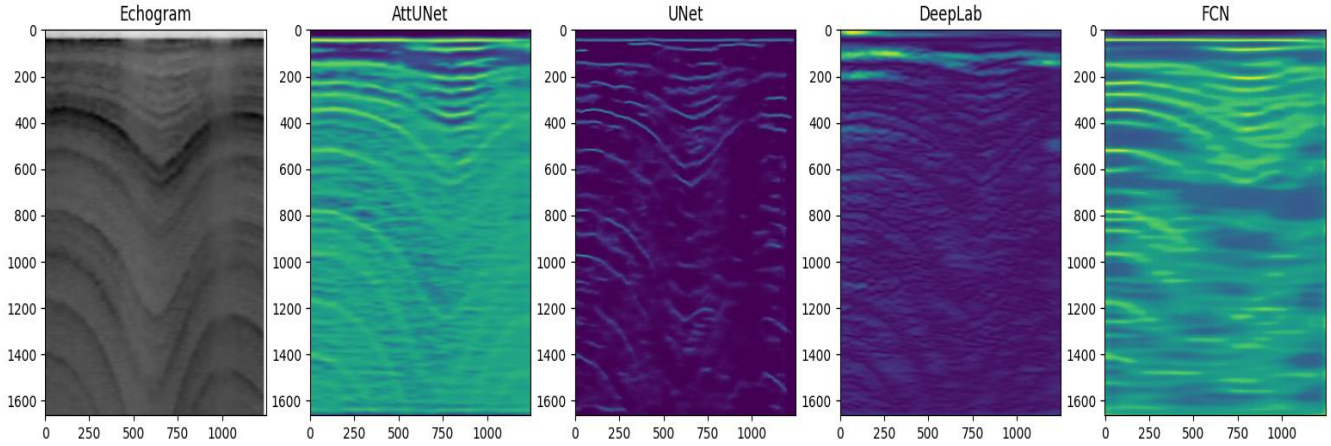


Figure 6-13: Outputs of earlier models on a challenging echogram image

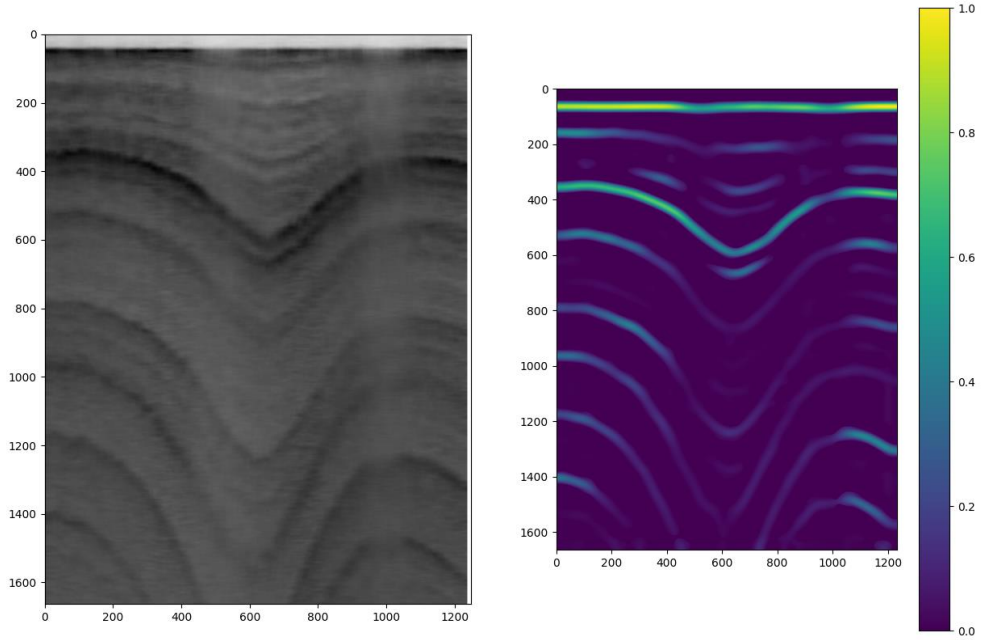


Figure 6-14: EchoRefine model output

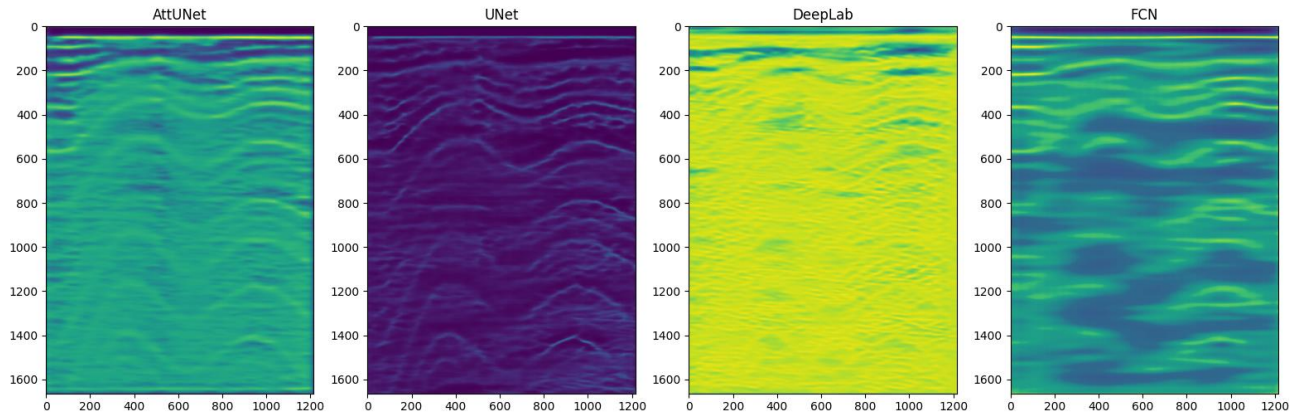


Figure 6-15: Outputs of earlier models on another challenging echogram image

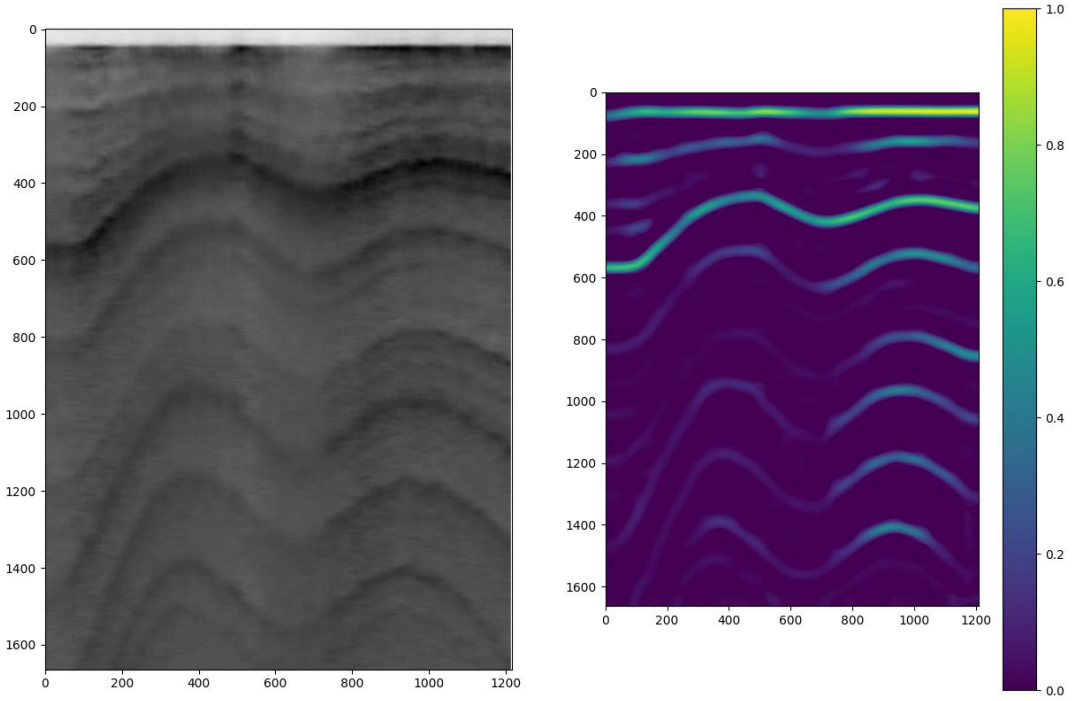


Figure 6-16: EchoRefine model output

6-2-7-2 Tracking Evaluation

Table 26: *N-pixel accuracies and mean absolute error of ViT models and EchoRefine*

	2px	5px	10px	MAE
SlowTime Patch	0.3239	0.7989	0.9605	3.7649
FastTime Patch	0.3262	0.7998	0.9612	3.7286
Combined	0.3150	0.8051	0.9600	3.7604
EchoRefine	**0.3612**	**0.8178**	**0.9947**	**3.5425**

Table 27: *Consecutive layer pixel prediction evaluation of ViT models and EchoRefine*

Metric	**Slow time patch**	**Fast time patch**	**Combined**	**EchoRefine**
Whole layer pixels	1.43	1.33	2.27	**1.26**
Intra-layer pixels	0.25	0.22	0.23	**0.15**
Combined	1.68	1.55	2.50	**1.41**

6-2-7-3 EchoRefine tracking performance based on echogram image quality

Table 28: *Mean absolute error (MAE) for L1, L2 and L3*

	L1	**L2**	**L3**
SlowTime Patch	3.880	3.748	38.042
FastTime Patch	3.858	3.709	36.935
Combined	3.499	3.799	36.837
EchoRefine	3.539	**3.546**	38.433

Table 29: *Consecutive layer pixel prediction evaluation based on image quality segments*

	Whole layer pixels			Intra-layer pixels			Combined Percentage
	L1	L2	L1 + L2	L1	L2	L1 + L2	
DeepLab	11 0.35%	248 1.17%	259 **1.52%**	15,215 1.89%	265,010 4.93%	280,225 **6.81%**	8.32%
FCN	16 0.51%	328 1.54%	1.46 **2.05%**	3,883 0.48%	133,847 2.49%	137,730 **2.97%**	5.02%
Slow time patch	10 0.32%	236 1.11%	246 1.43%	1645 0.20%	2424 0.05%	4,069 0.25%	1.68%
Fast time patch	10 0.32%	215 1.01%	225 1.33%	1447 0.18%	2379 0.04%	3,826 0.22%	1.55%
Combined	19 0.44%	388 1.83%	407 2.27%	1447 0.18%	2557 0.05%	4,004 0.23%	2.50%
EchoRefine	10 0.32%	201 0.94%	211 1.26%	1040 0.12%	1638 0.03%	2,678 0.15%	1.41%

Table 30: *N_pixel accuracies for each echogram image quality segment*

	L1			**L2**			**L3**		
	2px	**5px**	**10px**	**2px**	**5px**	**10px**	**2px**	**5px**	**10px**
SlowTime Patch	0.155	0.769	0.988	0.358	0.803	0.956	0.007	0.077	0.358
FastTime Patch	0.176	0.777	0.990	0.361	0.803	**0.957**	0.007	0.112	**0.381**
Combined	0.168	**0.828**	0.991	0.337	0.801	0.955	0.006	0.094	0.345
EchoRefine	**0.192**	0.808	**0.996**	**0.370**	0.778	**0.957**	0.010	0.088	0.341

Table 31: *MAE based on along-track distance for each echogram image quality segment*

	L1			L2		L3	
	2km	**5km**	**10km**	**2km**	**5km**	**2km**	**5km**
SlowTime Patch	3.851	3.918	3.959	3.766	3.702	**39.491**	36.024
FastTime Patch	3.846	3.851	3.946	3.761	3.577	39.744	36.664
Combined	**3.426**	3.579	3.740	3.817	3.757	40.571	**31.542**
EchoRefine	3.635	**3.382**	**3.343**	**3.549**	**3.538**	40.084	34.570

The results from Table 26 to

Table 31 record the performance of the EchoRefine model compared to the vision transformer models. This shows that EchoRefine generally outperforms other models across most metrics, especially in terms of N-pixel accuracies. It achieves the lowest MAE across all echogram image quality segments, indicating superior performance. Although its performance is comparable to some ViT models on some tasks (e.g. the combined model is competitive on the 5px accuracy and has lower L1 MAE), the EchoRefine model generally performs better than the transformer models.

Also, EchoRefine shows superior performance with the lowest percentage of missed pixels across all categories. It also has relatively consistent performance across L1 and L2 but has a high MAE for L3. Both EchoRefine and Combined ViT models exhibit some inconsistencies in the L3 segment performance across the experiments which reflects the inconsistency in the L3 echogram data. While the slow time patch and fast time patch models show consistent performance, they are generally outperformed by the EchoRefine model.

6-2-8 Generalization evaluation

The performance of the improved pipeline surpassed all previous implementations and was successfully used to track several previously unlabeled flightline with an estimate of over 10,000 5 km echogram frames. The performance of the model is consistent over dry snow zone echograms and most wet snow zone echograms. It also records strong performance on "L3-like" echograms that usually do not contain a clearly defined snow layer. Currently, the algorithm has been integrated into the Open Polar Radar Toolbox to automatically track all the layers in any given Snow Radar echogram image.

However, there are a few corner cases, particularly for very poor-quality echogram images, where the model tracking result might need manual adjustment. These corrections can be easily done using the Open Polar Radar Toolbox picker tool using any of the semi-automated algorithms.

The figures below show some examples of its performance on echogram images from different snow zones. The examples images are grouped based on how similar they are to L1, L2 and L3 test echogram images. The first image is the echogram, followed by the Refine Model activation and finally the tracked 1-D layer contours plotted over the echogram image.

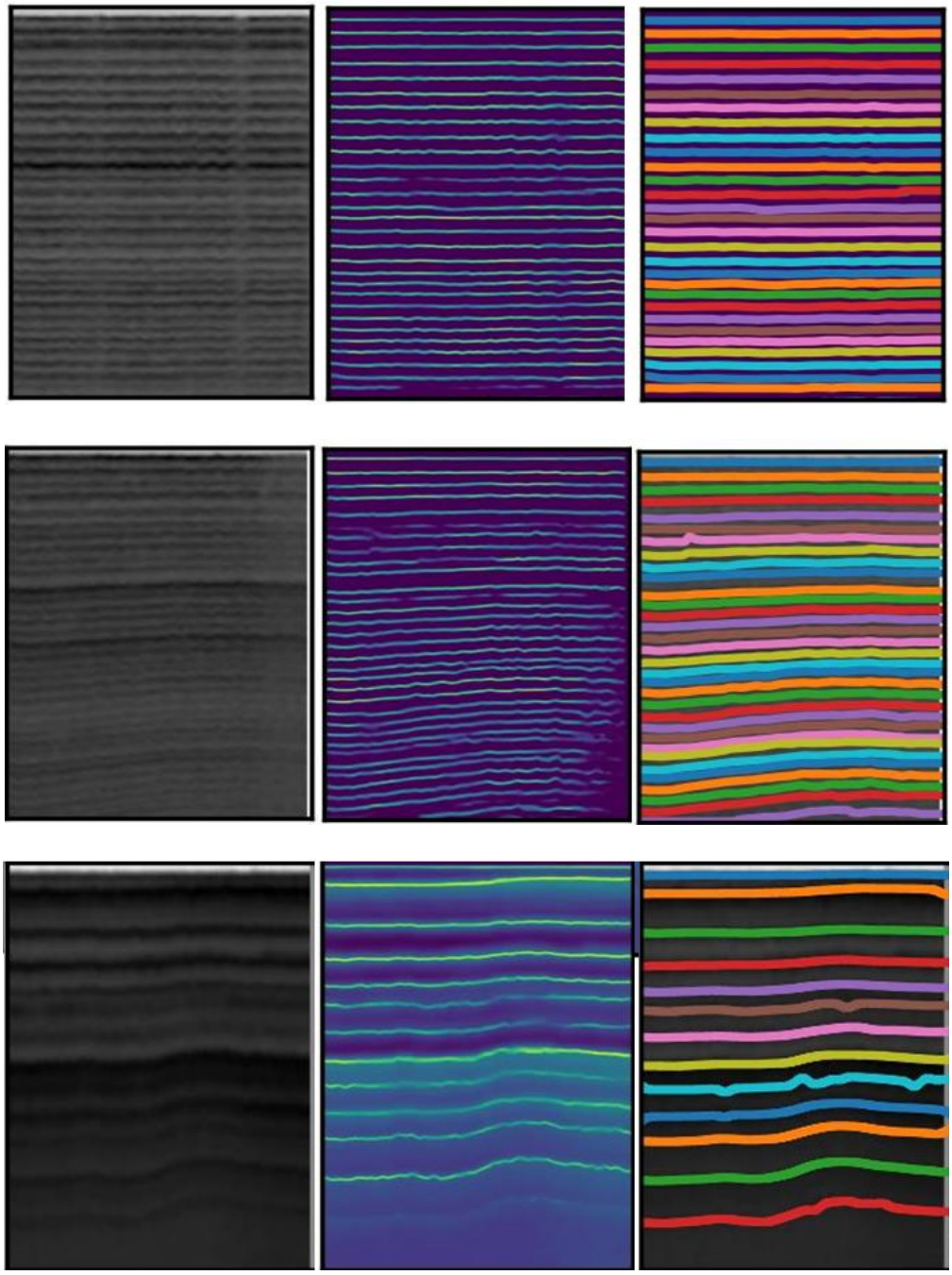


Figure 6-17: EchoRefine model output on echograms similar to L1 echograms

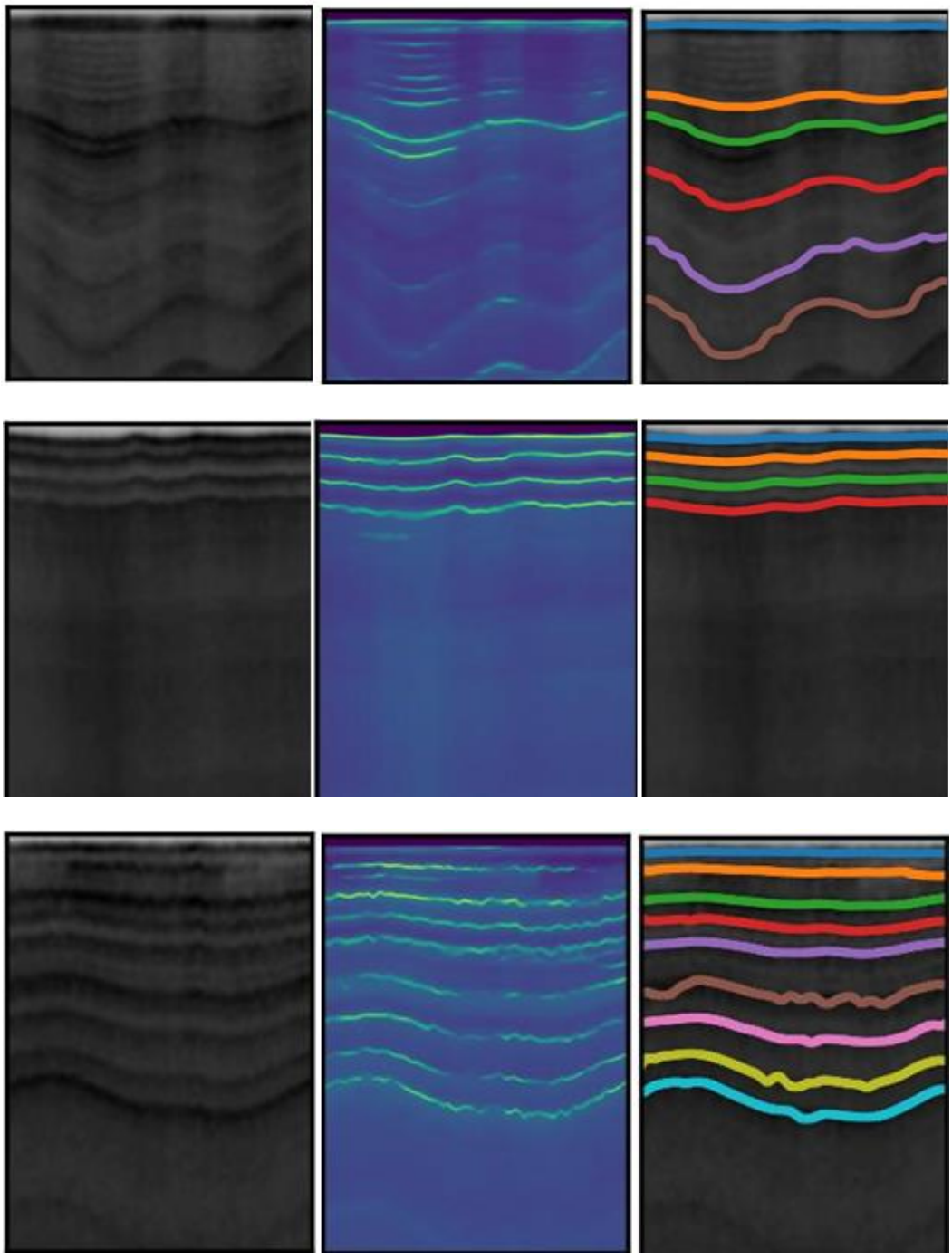


Figure 6-18: EchoRefine model output on echograms similar to L2 echograms

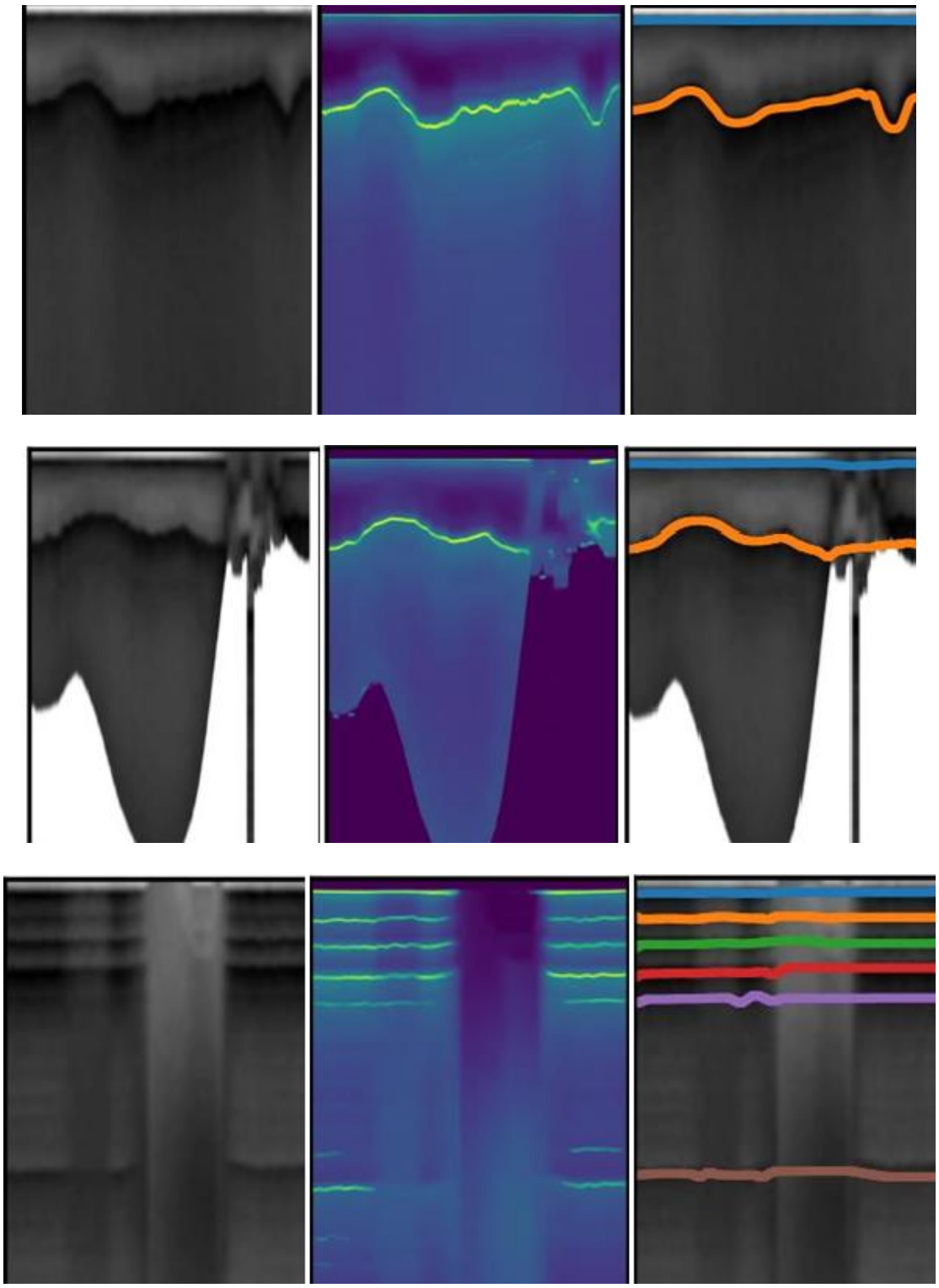


Figure 6-19: EchoRefine model output on echograms similar to L3 echograms

6-2-9 Ablation studies

In the implementation of the EchoRefine model and the updated training and inference pipeline, several changes were simultaneously integrated to address observed shortcomings in the previous models. This naturally raises the question of the relative importance of each modification and how earlier models might have performed with these updates.

To comprehensively answer this question, an elaborate ablation study, systematically removing each component, re-training the model and observing its impact on performance is needed. Despite the intention to conduct a comprehensive ablation study, the time constraints posed by the experimental setup makes it challenging to explore all the possible experiment configurations. Currently, each run of the training requires approximately one week to complete, as such, performing the more than 32 necessary runs for a thorough ablation analysis would extend the project timeline significantly. While the importance of this study is crucial in evaluating the contribution of individual features, we have opted to prioritize the development and validation of a working solution at this stage in the research.

However, it must be noted that the new features added to the updated training and inference pipeline were strategically added to mitigate specific deficiencies identified in earlier models. For example, the initial convolution-based models had limited performance on echogram images from the wet snow zone producing blurry edge delineations in the segmentation outputs that are difficult to binarize. To improve on this, the EchoRefine implemented a follow-up refinement module to further improve the boundaries of the “FCN-like” front architecture to improve the separation between the foreground and background pixel classes. The modified composite cost function follows the same thinking: to improve the along-track tracking performance of the convolution-based models to achieve similar performance as the transformer-based models.

However, there is still the need to return to the ablation studies to gain a deeper understanding of the individual contributions of each of the features. The insights gained from the current training and inference pipeline will provide a good foundation to support further refinement and optimization in future work. It is anticipated that these subsequent studies will not only validate the design choices but also reveal additional opportunities for enhancement, thereby further solidifying the models' performance.

6-2-10 Retrospective analysis of EchoRefine model modifications

Considering the several modifications made simultaneously to the EchoRefine model training pipeline such as the updated training dataset, improved echogram image preprocessing, etc., a retrospective analysis discussing how the new features might impact the performance of earlier models is imperative.

In the following sections, we will explore each of the earlier training paradigms that were implemented and examine how they might have benefited from the enhancements introduced in the new training pipeline. It should be noted, however, that these particular experiments have not yet been conducted at the time of this writing. Nevertheless, they remain an important area for future investigation, as further experimentation could provide a deeper understanding of the impact of these modifications and offer valuable insights that could inform subsequent studies and model refinements.

6-2-10-1 Retrospective comparison with the RowBlock algorithm

The RowBlock algorithm is characteristically different from the convolution-based or transformer models because it performs a form of feature engineering on the echogram image to create the input ColumnPatches used for training. This, coupled with the difference in the input

and output of the RowBlock models, being a multi-class classification model, makes it difficult to directly compare its performance with convolution and transformer-based segmentation models. The modifications made were informed primarily from the deficiencies noticed in earlier trained convolution models, hence more suited to segmentation models.

However, the RowBlock algorithm might be able to profit from the improved echogram preprocessing. The combination of the application of the hessian based vesselness filter and the deep learning-based echogram image denoiser adaptively removes the inter-layer noise which improves the distinction between the echogram layer pixel and no-layer pixels. This is shown in the EchoRefine model performance to contribute to the improved layer tracking performance.

Incorporating this preprocessing into the RowBlock algorithm pipeline would also improve the layer pixel delineation resulting in the ColumnPatches of higher image quality. As a result, this will potentially improve the classification result of the model. Further gains could also be achieved by supplementing the higher quality input ColumnPatches with classification model architectures with increased representation power to better learn the input-output mapping in the improved training data.

6-2-10-2 Retrospective comparison with the convolution-based algorithms

The earlier trained convolution models in Chapter 4 are better suited to take advantage of all the EchoRefine training pipeline modifications introduced. While these experiments are yet to be performed, it is expected that the models would also achieve better performance. Given that the echogram image preprocessing steps introduced produced better training and testing images, the additional incorporation of increased training echogram diversity, increasing the ground truth

label width and the neighboring-pixel-aware composite cost function all justify the hypothesis that all the convolution models will produce better segmentation outputs than values reported in 4-6-2. However, it is still expected that the EchoRefine will likely surpass their improved performance. We hypothesize that it has a more robust architecture for the following reasons.

First, the additional refinement module in the EchoRefine model has the advantage of hindsight (which is not available in earlier models). Although the model is trained end-to-end, the architecture affords it the ability to further improve the imperfections of the first stage to further refine and trace out the along-track layering of the snow layers. Also, the end-to-end 2-stage architecture of the EchoRefine model increases the count of the model parameters and therefore the representation power of the model, putting it in a better position to correctly model the layer/no-layer pixel distinction and perform better on echogram images with less obvious layer pixel distinction such as wet snow zone echograms due to its auxiliary refinement module.

6-2-11 Accessing Python Code and Dataset

The code for the models and the datasets created can be found here:
https://gitlab.com/openpolarradar/opr/-/wikis/Machine-Learning-Guide

Chapter 7 CONCLUSION

This work presents our findings in developing a generalizable machine learning and deep learning computer vision algorithm for the echogram layer tracking problem. Given the enormous amount of radar sounding data that has been collected over several decades, there is a need for an automatic layer tracker to extract ice layers from the radar echograms.

Using machine learning and deep learning algorithms, infused with advanced signal processing to pre-process the images, custom models were designed to track the snow accumulation layers in echogram images from the dry snow and wet snow zones. The RowBlock algorithm took advantage of the chronology of layers in each rangeline and spatial correlation among neighboring rangelines to track the layers one at a time, reconstructing the complex layer tracking problem as a sequential multiclass classification task.

Multiple convolution-based deep learning algorithms were also developed to process the whole echogram image at once. While these excelled on echograms from the dry snow zone, their limited utilization of along-track layer correlation prompted the need for algorithms that can exploit the correlation that exists in the columns and rows of the echogram matrix. Consequently, transformer-based vision models were designed with three patchification schemes. These implementations resulted in improved along track tracking surpassing most of the earlier convolution-based methods. However, the fixed embedding layer of transformer-based models requires that echogram images are reshaped to match the fixed dimensions which could potentially lead to loss of resolution in either fast time or slow time dimensions.

Combining the lessons learned from the different architectures that were investigated, a multi-pronged approach was employed to improve several sections of the convolution-based deep learning model pipeline leading to the EchoRefine model with the 8-pixel connectivity constraint cost function. This resulted in improved along-track tracking performance and satisfactory performance even in echograms from the wet snow zone. Finally, an improved 1-D accumulation layer contour extraction algorithm was developed to efficiently return each 1-D layer contour from the segmentation model heat map output. This algorithm has been integrated into the Open Polar Radar Toolbox to automatically track many of the previously untracked echograms created from science campaigns from 2012 to 2021 which were not part of the training data.

As a result of the progress made in this work, two standard deep learning Snow Radar datasets with 11,000 and 50,000 echograms synchronized with climate fields from the Modèle Atmosphérique Régional climate model have been created and made available to the deep learning and science communities to boost further studies.

REFERENCES

[1] N. United, “Climate change ‘biggest threat modern humans have ever faced’”, [Online]. Available: https://press.un.org/en/2021/sc14445.doc.htm

[2] W. Health Organization, “We must fight one of the world’s biggest threats - climate change.”

[3] S. K. Gulev *et al.*, “Changing state of the climate system,” in *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, V. Masson-Delmotte, P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, Ö. Yelekçi, R. Yu, and B. Zhou, Eds., Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press, 2021, pp. 287–422. doi: 10.1017/9781009157896.001.

[4] S. I. Seneviratne *et al.*, “Weather and climate extreme events in a changing climate,” in *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, V. Masson-Delmotte, P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, Ö. Yelekçi, R. Yu, and B. Zhou, Eds., Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press, 2021, pp. 1513–1766. doi: 10.1017/9781009157896.001.

[5] R. Ranasinghe *et al.*, “Climate change information for regional impact and for risk assessment,” in *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, V. Masson-Delmotte, P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, Ö. Yelekçi, R. Yu, and B. Zhou, Eds., Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press, 2021, pp. 1767–1926. doi: 10.1017/9781009157896.001.

[6] National Snow and Ice Data Center, “Why Ice Sheets Matter.” [Online]. Available: https://nsidc.org/learn/parts-cryosphere/ice-sheets/why-ice-sheets-matter

[7] E. Rignot, J. Mouginot, B. Scheuchl, M. van den Broeke, M. J. van Wessem, and M. Morlighem, “Four decades of Antarctic Ice Sheet mass balance from 1979-2017,” *Proc. Natl. Acad. Sci.*, vol. 116, no. 2019, pp. 1095–1103, 2019.

[8] H. Yu, E. Rignot, H. Seroussi, and M. Morlighem, “Retreat of Thwaites Glacier, West Antarctica, over the next 100 years using various ice flow models, ice shelf melt scenarios and basal friction laws,” *The Cryosphere*, vol. 12, no. 2018, pp. 3861–76, 2018.

[9] A. Post and et al, “A complex relationship between calving glaciers and climate,” *Eos Trans. Am. Geophys. Union*, vol. 92, no. 37, pp. 305–306, 2011.

[10] G. Lewis and et al, “Regional Greenland accumulation variability from Operation IceBridge airborne accumulation radar,” *The Cryosphere*, vol. 11, no. 2, pp. 773–788, 2017.

[11] B. Medley and et al, “Constraining the recent mass balance of Pine Island and Thwaites glaciers, West Antarctica, with airborne observations of snow accumulation,” *The Cryosphere*, vol. 8, no. 4, pp. 1375–1392, 2014.

[12] B. Zalatan and M. Rahnemoonfar, “Recurrent Graph Convolutional Networks for Spatiotemporal Prediction of Snow Accumulation Using Airborne Radar,” *ArXiv Prepr. ArXiv230200817*, 2023.

[13] B. Panzer and et al, “An ultra-wideband, microwave radar for measuring snow thickness on sea ice and mapping near-surface internal layers in polar firn,” *J. Glaciol.*, vol. 59, no. 214, pp. 244–254, 2013.
[14] L. Shi and et al, “Multichannel coherent radar depth sounder for NASA operation ice bridge,” in *2010 IEEE international geoscience and remote sensing symposium*, IEEE, 2010.
[15] E. Arnold and et al, “CReSIS airborne radars and platforms for ice and snow sounding,” *Ann. Glaciol.*, vol. 61, no. 81, pp. 58–67, 2020.
[16] F. Rodriguez-Morales and et al, “Advanced multifrequency radar instrumentation for polar research,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2824–2842, 2013.
[17] J. Li and et al, “Snow stratigraphy observations from Operation IceBridge surveys in Alaska using S and C band airborne ultra-wideband FMCW (frequency-modulated continuous wave) radar,” *The Cryosphere*, vol. 17, no. 1, pp. 175–193, 2023.
[18] B. Feng and et al, “Firn stratigraphic genesis in early spring: evidence from airborne radar,” *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 9, no. 6, pp. 2429–2435, 2016.
[19] Z. Wang and et al, “Mapping age and basal conditions of ice in the Dome Fuji region, Antarctica, by combining radar internal layer stratigraphy and flow modeling,” *Cryosphere Discuss.*, pp. 1–22, 2023.
[20] C. Miege and et al, “Southeast Greenland high accumulation rates derived from firn cores and ground-penetrating radar,” *Ann. Glaciol.*, vol. 54, no. 63, pp. 322–332, 2013.
[21] A. Garcia-Garcia and et al, “A review on deep learning techniques applied to semantic segmentation,” *ArXiv Prepr. ArXiv170406857*, 2017.
[22] M. Orsic and et al, “In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
[23] R. P. K. Poudel, S. Liwicki, and R. Cipolla, “Fast-scnn: Fast semantic segmentation network,” *ArXiv Prepr. ArXiv190204502*, 2019.
[24] S. Gao and et al, “Large-scale unsupervised semantic segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 2022, pp. 141800–141818, 2022.
[25] S. Graham and et al, “One model is all you need: multi-task learning enables simultaneous histology image segmentation and classification,” *Med. Image Anal.*, vol. 83, p. 102685, 2023.
[26] H. Cao and et al, “Swin-unet: Unet-like pure transformer for medical image segmentation,” in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, 2023.
[27] D. Luo and et al, “GDN: Guided down-sampling network for real-time semantic segmentation,” *Neurocomputing*, vol. 520, pp. 205–215, 2023.
[28] J. Schlemper and et al, “Attention gated networks: Learning to leverage salient regions in medical images,” *Med. Image Anal.*, vol. 53, pp. 197–207, 2019.
[29] A. Vaswani and et al, “Attention is all you need,” in *Advances in neural information processing systems*, 2017.
[30] A. Dosovitskiy and et al, “An image is worth 16x16 words: Transformers for image recognition at scale,” *ArXiv Prepr. ArXiv201011929*, 2020.
[31] K. Kawaguchi, L. P. Kaelbling, and Y. Bengio, “Generalization in deep learning,” in *arXiv preprint arXiv:1710.05468*, 2017.
[32] B. Neyshabur and et al, “Exploring generalization in deep learning,” in *Advances in neural information processing systems*, 2017.

[33] D. Yu and L. Deng, “Deep learning and its applications to signal and information processing [exploratory dsp],” *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 145–154, 2010.
[34] P. Lang and et al, “A comprehensive survey of machine learning applied to radar signal processing,” *ArXiv Prepr. ArXiv200913702*, 2020.
[35] M. Jia and et al, “Human activity classification with radar signal processing and machine learning,” in *2020 International conference on UK-China emerging technologies (UCET)*, IEEE, 2020.
[36] Z. Geng and et al, “Deep-learning for radar: A survey,” *IEEE Access*, vol. 9, pp. 141800–141818, 2021.
[37] C. M. Gifford and et al, “Automated polar ice thickness estimation from radar imagery,” *IEEE Trans Image Process*, vol. 19, no. 9, pp. 2456–2469, 2010.
[38] A. Ferro and L. Bruzzone, “Analysis of radar sounder signals for the automatic detection and characterization of subsurface features,” *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4333–4348, 2012.
[39] A.-M. Ilisei, A. Ferro, and L. Bruzzone, “A technique for the automatic estimation of ice thickness and bedrock properties from radar sounder data acquired at Antarctica,” in *2012 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2012.
[40] B. Medley and et al, “Airborne-radar and ice-core observations of annual snow accumulation over Thwaites Glacier, West Antarctica confirm the spatiotemporal variability of global and regional atmospheric models,” *Geophys. Res. Lett.*, vol. 40, no. 14, pp. 3649–3654, 2013.
[41] 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 Trans Geosci Remote Sens*, vol. 55, no. 9, pp. 5115–5122, 2017.
[42] J. E. Mitchell, D. J. Crandall, G. C. Fox, and J. D. Paden, “A semi-automatic approach for estimating near surface internal layers from snow radar imagery,” in *International Geoscience and Remote Sensing Symposium*, 2013, pp. 4110–4113.
[43] V. de Paul Onana and et al, “A semiautomated multilayer picking algorithm for ice-sheet radar echograms applied to ground-based near-surface data,” *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 1, pp. 51–69, 2014.
[44] M. Rahnemoonfar and et al, “Automatic ice thickness estimation in radar imagery based on charged particles concept,” *2017 IEEE Int. Geosci. Remote Sens. Symp. IGARSS*, 2017.
[45] D. J. Crandall, G. C. Fox, and J. D. Paden, “Layer-finding in radar echograms using probabilistic graphical models,” in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, pp. 1530–1533.
[46] S. Lee, J. Mitchell, D. Crandall, and G. Fox, “Estimating bedrock and surface layer boundaries and confidence intervals in ice sheet radar imagery using MCMC,” in *International Conference on Image Processing*, 2014, pp. 111–115.
[47] M. Xu, D. Crandall, G. Fox, and J. Paden, “Automatic Estimation of Ice Bottom Surfaces from Radar Imagery,” in *IEEE International Conference on Image Processing*, 2017.
[48] V. L. Berger Pereira da Silva, “Probabilistic Methods for Ice Thickness Estimation Using Radar Imagery,” PhD Thesis, The University of Texas at Austin, 2019.
[49] “Automatic Enhancement and Detection of Layering in Radar Sounder Data Based on a Local Scale Hidden Markov Model and the Viterbi Algorithm | IEEE Journals & Magazine | IEEE Xplore.” Accessed: Jul. 09, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/7731235
[50] H. Kamangir, M. Rahnemoonfar, D. Dobbs, J. Paden, and G. Fox, “Deep Hybrid Wavelet Network for Ice Boundary Detection in Radra Imagery,” in *IGARSS 2018 - 2018 IEEE*

International Geoscience and Remote Sensing Symposium, Jul. 2018, pp. 3449–3452. doi: 10.1109/IGARSS.2018.8518617.

[51] M. Xu, C. Fan, J. D. Paden, G. C. Fox, and D. J. Crandall, "Multi-task Spatiotemporal Neural Networks for Structured Surface Reconstruction," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Mar. 2018, pp. 1273–1282. doi: 10.1109/WACV.2018.00144.

[52] D. Varshney, M. Rahnemoonfar, M. Yari, and J. Paden, "Deep Ice Layer Tracking and Thickness Estimation using Fully Convolutional Networks," in *2020 IEEE International Conference on Big Data (Big Data)*, Dec. 2020, pp. 3943–3952. doi: 10.1109/BigData50022.2020.9378070.

[53] Y. Cai, S. Hu, S. Lang, Y. Guo, and J. Liu, "End-to-End Classification Network for Ice Sheet Subsurface Targets in Radar Imagery," *Appl. Sci.*, vol. 10, no. 7, Art. no. 7, Jan. 2020, doi: 10.3390/app10072501.

[54] M. Yari, M. Rahnemoonfar, and J. Paden, "Multi-Scale and Temporal Transfer Learning for Automatic Tracking of Internal Ice Layers," in *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, Sep. 2020, pp. 6934–6937. doi: 10.1109/IGARSS39084.2020.9323758.

[55] Y. Wang, M. Xu, J. Paden, L. Koenig, G. Fox, and D. Crandall, "Deep Tiered Image Segmentation For Detecting Internal Ice Layers in Radar Imagery," arXiv.org. Accessed: Jul. 09, 2024. [Online]. Available: https://arxiv.org/abs/2010.03712v3

[56] "Radar Sensor Simulation with Generative Adversarial Network | IEEE Conference Publication | IEEE Xplore." Accessed: Jul. 09, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/9323676

[57] M. Rahnemoonfar, M. Yari, J. Paden, L. Koenig, and O. Ibikunle, "Deep multi-scale learning for automatic tracking of internal layers of ice in radar data," *J. Glaciol.*, vol. 67, no. 261, pp. 39–48, Feb. 2021, doi: 10.1017/jog.2020.80.

[58] M. Yari *et al.*, "Airborne Snow Radar Data Simulation With Deep Learning and Physics-Driven Methods," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 12035–12047, 2021, doi: 10.1109/JSTARS.2021.3126547.

[59] "Learning Snow Layer Thickness Through Physics Defined Labels | IEEE Conference Publication | IEEE Xplore." Accessed: Jul. 09, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/9884370

[60] R. Ghosh and F. Bovolo, "TransSounder: A Hybrid TransUNet-TransFuse Architectural Framework for Semantic Segmentation of Radar Sounder Data," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2022, doi: 10.1109/TGRS.2022.3180761.

[61] Y. Cai, F. Wan, S. Hu, and S. Lang, "Accurate prediction of ice surface and bottom boundary based on multi-scale feature fusion network," *Appl. Intell.*, vol. 52, no. 14, pp. 16370–16381, Nov. 2022, doi: 10.1007/s10489-022-03362-1.

[62] J. Li *et al.*, "Snow Radar Measurements over Alaska Mountains, Icefields and Glaciers as part of the Operation IceBridge".

[63] D. Young *et al.*, "Comprehensive multi frequency airborne mapping of the southern flank of Dome A: results of the COLDEX airborne program.," Copernicus Meetings, EGU24-12995, Mar. 2024. doi: 10.5194/egusphere-egu24-12995.

[64] X. Tang *et al.*, "Glaciological and Meteorological Conditions at the Chinese Taishan Station, East Antarctica," *Front. Earth Sci.*, vol. 8, Aug. 2020, doi: 10.3389/feart.2020.00250.

[65] J. Li and et al, "Snow stratigraphy observations from Operation IceBridge surveys in Alaska using S and C band airborne ultra-wideband FMCW (frequency-modulated continuous wave) radar," *The Cryosphere*, vol. 17, no. 1, pp. 175–193, 2023.

[66] S. Rahman, “FMCW Radar Signal Processing for Antarctic Ice Shelf Profiling and Imaging”.
[67] A. A. W. Fitzpatrick *et al.*, “Ice flow dynamics and surface meltwater flux at a land-terminating sector of the Greenland ice sheet,” *J. Glaciol.*, vol. 59, no. 216, pp. 687–696, 2013, doi: 10.3189/2013JoG12J143.
[68] S. K. Meher and G. Panda, “Deep learning in astronomy: a tutorial perspective,” *Eur. Phys. J. Spec. Top.*, vol. 230, no. 10, pp. 2285–2317, Sep. 2021, doi: 10.1140/epjs/s11734-021-00207-9.
[69] D. Baron, “Machine Learning in Astronomy: a practical overview,” Apr. 15, 2019, *arXiv*: arXiv:1904.07248. doi: 10.48550/arXiv.1904.07248.
[70] K. Choudhary *et al.*, “Recent advances and applications of deep learning methods in materials science,” *Npj Comput. Mater.*, vol. 8, no. 1, pp. 1–26, Apr. 2022, doi: 10.1038/s41524-022-00734-6.
[71] M. Ge, F. Su, Z. Zhao, and D. Su, “Deep learning analysis on microscopic imaging in materials science,” *Mater. Today Nano*, vol. 11, p. 100087, Aug. 2020, doi: 10.1016/j.mtnano.2020.100087.
[72] W. S. Alharbi and M. Rashid, “A review of deep learning applications in human genomics using next-generation sequencing data,” *Hum. Genomics*, vol. 16, no. 1, p. 26, Jul. 2022, doi: 10.1186/s40246-022-00396-x.
[73] T. Yue *et al.*, “Deep Learning for Genomics: A Concise Overview,” Oct. 04, 2023, *arXiv*: arXiv:1802.00810. doi: 10.48550/arXiv.1802.00810.
[74] C. Baur, S. Albarqouni, and N. Navab, “Semi-supervised deep learning for fully convolutional networks,” in *Medical Image Computing and Computer Assisted Intervention− MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III*, Springer International Publishing, 2017.
[75] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
[76] X. Wang, F. Yu, Z.-Y. Dou, T. Darrell, and J. E. Gonzalez, “SkipNet: Learning Dynamic Routing in Convolutional Networks,” Jul. 25, 2018, *arXiv*: arXiv:1711.09485. doi: 10.48550/arXiv.1711.09485.
[77] C. B. Vennerød, A. Kjærran, and E. S. Bugge, “Long Short-term Memory RNN,” May 14, 2021, *arXiv*: arXiv:2105.06756. doi: 10.48550/arXiv.2105.06756.
[78] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks,” Dec. 14, 2014, *arXiv*: arXiv:1409.3215. doi: 10.48550/arXiv.1409.3215.
[79] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
[80] K. CReSIS, *Open Polar Radar Toolbox*. [Online]. Available: https://gitlab.com/openpolarradar/opr/
[81] “The viterbi algorithm | IEEE Journals & Magazine | IEEE Xplore.” Accessed: May 15, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1450960
[82] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” May 18, 2015, *arXiv*: arXiv:1505.04597. doi: 10.48550/arXiv.1505.04597.
[83] M. A. Richards, *Fundamentals of radar signal processing*, Third edition. New York: McGraw Hill, 2022.
[84] M. M. Herron and C. C. Langway, “Firn Densification: An Empirical Model,” *J. Glaciol.*, vol. 25, no. 93, pp. 373–385, Jan. 1980, doi: 10.3189/S0022143000015239.
[85] X. Fettweis *et al.*, “GrSMBMIP: intercomparison of the modelled 1980–2012 surface mass balance over the Greenland Ice Sheet,” *The Cryosphere*, vol. 14, no. 11, pp. 3935–3958, Nov. 2020, doi: 10.5194/tc-14-3935-2020.

[86] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” Mar. 08, 2015, *arXiv*: arXiv:1411.4038. doi: 10.48550/arXiv.1411.4038.
[87] O. Oktay *et al.*, “Attention U-Net: Learning Where to Look for the Pancreas,” May 20, 2018, *arXiv*: arXiv:1804.03999. doi: 10.48550/arXiv.1804.03999.
[88] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking Atrous Convolution for Semantic Image Segmentation,” Dec. 05, 2017, *arXiv*: arXiv:1706.05587. doi: 10.48550/arXiv.1706.05587.
[89] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” Jan. 29, 2017, *arXiv*: arXiv:1412.6980. doi: 10.48550/arXiv.1412.6980.
[90] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering,” *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007, doi: 10.1109/TIP.2007.901238.
[91] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising,” *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017, doi: 10.1109/TIP.2017.2662206.
[92] S. Gu, L. Zhang, W. Zuo, and X. Feng, “Weighted Nuclear Norm Minimization with Application to Image Denoising,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA: IEEE, Jun. 2014, pp. 2862–2869. doi: 10.1109/CVPR.2014.366.
[93] K. Zhang, W. Zuo, and L. Zhang, “FFDNet: Toward a Fast and Flexible Solution for CNN based Image Denoising,” *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, Sep. 2018, doi: 10.1109/TIP.2018.2839891.
[94] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, “Plug-and-Play Image Restoration with Deep Denoiser Prior,” Jul. 12, 2021, *arXiv*: arXiv:2008.13751. Accessed: Jul. 02, 2024. [Online]. Available: http://arxiv.org/abs/2008.13751
[95] Y. Sato *et al.*, “Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images,” *Med. Image Anal.*, vol. 2, no. 2, pp. 143–168, Jun. 1998, doi: 10.1016/S1361-8415(98)80009-1.
[96] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, “Multiscale vessel enhancement filtering,” in *Medical Image Computing and Computer-Assisted Intervention — MICCAI’98*, W. M. Wells, A. Colchester, and S. Delp, Eds., Berlin, Heidelberg: Springer, 1998, pp. 130–137. doi: 10.1007/BFb0056195.
[97] C.-C. Ng, M. H. Yap, N. Costen, and B. Li, “Automatic Wrinkle Detection Using Hybrid Hessian Filter,” in *Computer Vision -- ACCV 2014*, D. Cremers, I. Reid, H. Saito, and M.-H. Yang, Eds., Cham: Springer International Publishing, 2015, pp. 609–622. doi: 10.1007/978-3-319-16811-1_40.
[98] E. Meijering, M. Jacob, J.-C. f. Sarria, P. Steiner, H. Hirling, and M. Unser, “Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images,” *Cytometry A*, vol. 58A, no. 2, pp. 167–176, 2004, doi: 10.1002/cyto.a.20022.
[99] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal Loss for Dense Object Detection”.
[100] G. Lin, A. Milan, C. Shen, and I. Reid, “RefineNet: Multi-path Refinement Networks for High-Resolution Semantic Segmentation,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, Jul. 2017, pp. 5168–5177. doi: 10.1109/CVPR.2017.549.
[101] T. Wang, A. Borji, L. Zhang, P. Zhang, and H. Lu, “A Stagewise Refinement Model for Detecting Salient Objects in Images,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice: IEEE, Oct. 2017, pp. 4039–4048. doi: 10.1109/ICCV.2017.433.

[102] X. Li, F. Yang, H. Cheng, W. Liu, and D. Shen, “Contour Knowledge Transfer for Salient Object Detection,” in *Computer Vision – ECCV 2018*, vol. 11219, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., in Lecture Notes in Computer Science, vol. 11219. , Cham: Springer International Publishing, 2018, pp. 370–385. doi: 10.1007/978-3-030-01267-0_22.
[103] N. Liu and J. Han, “DHSNet: Deep Hierarchical Saliency Network for Salient Object Detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 678–686. doi: 10.1109/CVPR.2016.80.
[104] T. Wang *et al.*, “Detect Globally, Refine Locally: A Novel Approach to Saliency Detection,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 3127–3135. doi: 10.1109/CVPR.2018.00330.
[105] 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)*, Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.
[106] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
[107] X. Qin *et al.*, “Boundary-Aware Segmentation Network for Mobile and Web Applications,” May 11, 2021, *arXiv*: arXiv:2101.04704. doi: 10.48550/arXiv.2101.04704.
[108] N. Otsu, “A Tlreshold Selection Method from Gray-Level Histograms”.
[109] R. M. Haralick, S. R. Sternberg, and X. Zhuang, “Image Analysis Using Mathematical Morphology,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 4, pp. 532–550, Jul. 1987, doi: 10.1109/TPAMI.1987.4767941.
[110] J. Serra and P. Soille, *Mathematical Morphology and Its Applications to Image Processing*. Springer Science & Business Media, 2012.
[111] P. Maragos, R. W. Schafer, and M. A. Butt, *Mathematical Morphology and Its Applications to Image and Signal Processing*. Springer Science & Business Media, 2012.
[112] C. Arcelli, L. P. Cordella, and S. Levialdi, “From Local Maxima to Connected Skeletons,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-3, no. 2, pp. 134–143, Mar. 1981, doi: 10.1109/TPAMI.1981.4767071.
[113] D. P. McKay and S. C. Shapiro, “Using Active Connection Graphs for Reasoning with Recursive Rules.,” in *IJCAI*, Citeseer, 1981, pp. 368–375.
[114] L. Di Stefano and A. Bulgarelli, “A simple and efficient connected components labeling algorithm,” in *Proceedings 10th International Conference on Image Analysis and Processing*, Sep. 1999, pp. 322–327. doi: 10.1109/ICIAP.1999.797615.
[115] R. M. Haralick, “Model-based morphology: Simple and complex shapes,” *Vis. Form Anal. Recognit.*, pp. 275–285, 1992.
[116] L. He, Y. Chao, K. Suzuki, and K. Wu, “Fast connected-component labeling,” *Pattern Recognit.*, vol. 42, no. 9, pp. 1977–1987, Sep. 2009, doi: 10.1016/j.patcog.2008.10.013.