Title: A weakly supervised transfer learning approach for radar sounder data segmentation

This paper appears in: IEEE Transactions on Geoscience and Remote Sensing

Date of Publication: 6 March 2023

Author(s): Miguel Hoyo Garcia; Elena Donini; Francesca Bovolo

Volume: -

Page(s): 1 – 18

DOI: 10.1109/TGRS.2023.3252939

A weakly supervised transfer learning approach for radar sounder data segmentation

Miguel Hoyo García, *Student Member, IEEE* Elena Donini, *Member, IEEE,*
and Francesca Bovolo, *Senior Member, IEEE*

***Abstract*—Airborne Radar Sounders (RSs) are active sensors that acquire subsurface data for Earth observation. RS data (radargrams) provide information on buried geology by imaging subsurface dielectric discontinuities. Recently, several automatic RS target identification techniques have been proposed, being convolutional neural network (CNN)-based methods the most promising. However, they require numerous labeled data that are hard to retrieve in the subsurface environment targeted by RS. Further, they are not designed to effectively deal with problems showing unbalanced classes like RS segmentation. We introduce newer cryosphere subsurface targets in the inland and coastal areas that can have a very low probability. To deal with the higher complexity and variability than previous works, we propose a transfer learning framework for RS data to mitigate the need for a large amount of labeled data and handle extremely unbalanced target classes. Herewith, we propose two transfer learning-based mechanisms for radargram segmentation. The first uses a lightweight architecture whose pre-training is supervised with a large labeled dataset from other domains. The second mechanism uses a deep architecture pre-trained in the RS domain, considering the pretest task of radargram reconstruction. The architectures are modified to deal with the characteristics of RS data and the radargram segmentation task. Finally, both methods are fine-tuned with a few labeled radargrams to learn radargram features useful for segmentation. We reveal experimental results on radargrams acquired in Antarctica by MCoRDS-1 and MCoRDS-3. The results demonstrate the effectiveness of transfer learning for radargram segmentation.**

***Index Terms*—Radar sounder, remote sensing, cryosphere, domain adaptation, transfer learning, deep learning.**

I. Introduction

Understanding the dynamics of the ice sheets and shelves helps in predicting the impact of climate change and the evolution of the cryosphere [1]. Analyzing the inner structure of ice sheets and shelves, one can estimate climate change indicators, for example, related to the ice shelf melting and the ice-sheet mass balance decrease [1]. The analysis of the mass balance of ice sheets and shelves requires the direct measurement of the ice up to the basal interface to extract crucial information on the subsurface geologic structures and processes [2]. These direct measurements can be provided by radar sounders (RSs). RSs are active sensors that probe the subsurface by transmitting electromagnetic (EM) waves at the nadir. The EM waves have a central frequency in the High Frequency (HF) and Very-High Frequency (VHF) ranges with a relatively wide band. The interaction of the EM waves with the dielectric interfaces in the subsurface results in backscattered echoes captured by the sensor. Radargrams are generated by coherently adding and concatenating the echoes in the along-track direction. Most of the available RS data on Earth have been acquired in the cryosphere. Radargrams image, for example, continental ice stratigraphy, basal interface, floating ice and crevasses, as well as noise-limited areas that include the thermal noise and the echo-free zone (see Fig. 1).

RSs for Earth observation are mainly mounted on an airborne platform and generally probe the subsurface of Greenland and Antarctica. Motivated by the expected increasing amount of radar sounders from planned airborne and satellite-borne missions [3], [4], [5], several authors [6], [7] designed automatic approaches for radargram analysis based on statistical methods and machine learning techniques to identify targets in the cryosphere subsurface. For example, Ilisei et al. [8] proposed an approach based on handcrafted features and a Support Vector Machine (SVM) that segments grounded ice sheet radargrams. This approach employed several input features (e.g., entropy) that are target-specific and manually designed. SVM-based methods are also used to detect and segment specific targets in cryosphere radargrams. Donini et al. [6] presented a methodology for detecting basal ice. Similarly, [7] automatically detects subglacial lakes using handcrafted features modeling the geophysical behavior of the basal interface with and without subglacial lakes. The main disadvantage of approaches based on handcrafted features is that every new target class requires a complete redesign of the input features, which is highly time-consuming. Moreover, these approaches focus on detecting inland region targets, simplifying the problem, and none analyzing more complex targets, such as floating ice and crevasses in coastal areas. This limits the potential of automatic cryosphere segmentation and systematic information extraction on the subsurface geology at a large scale in space and time [9].

Recently, advanced deep learning (DL) techniques, such as convolutional neural networks (CCNs), have become prominent in analyzing and segmenting several data types [10]. Deep CNNs automatically learn semantically meaningful features from the data during training at the cost of a large amount

This work was supported by the Italian Space Agency through the "Attività Scientifiche per JUICE fase C-D" under Contract Agenzia Spaziale Italian - Istituto Nazionale di Astrofisica (ASI-INAF) and Contract 2018-25-HH.0.

M. Hoyo García is with the Center for Digital Society, Fondazione Bruno Kessler, 38123 Trento, Italy, and also the Department of Information Engineering and Computer Science, University of Trento, 38123 Trento, Italy (e-mail: mhoyogarcia@fbk.eu).

E. Donini and F. Bovolo are with the Center for Digital Society, Fondazione Bruno Kessler, 38123 Trento, Italy (e-mails: edonini@fbk.eu, bovolo@fbk.eu).

of labeled data. Insufficient labeled training data could lead to overfitting since the amount of trainable weights in a DL architecture is usually extremely large [11]. Therefore, the larger the training dataset is, the more meaningful and general will be the learned features. If the large labeled dataset is unavailable, mitigation techniques exist to avoid overfitting and sub-optimal results [11]. For instance, the W-Net by Xia et al. [12] proposes a convolutional autoencoder (CAE) trained with a proxy task (i.e., data reconstruction) that does not require labeled data. Also, data augmentation [13] can be employed to mitigate overfitting and improve accuracy with limited training samples. However, when the training set is strongly limited, such as in the RS icy data, data augmentation is sub-optimal as it does not introduce enough data variability [10]. Alternatives include reducing the trainable weights to avoid overfitting (e.g., using a lightweight CNN [14]) and using transfer learning approaches. Transfer learning aims to reuse solidly pre-trained architectures on a source task $\mathcal{T}_S$ to adapt a network to a target task $\mathcal{T}_T$ [15]. The network is pre-trained in a domain $\mathcal{D}_S$ where a large labeled training set is available [16]. Subsequently, the network is adapted and fine-tuned with a small amount of labeled data in the target domain $\mathcal{D}_T$, exploiting weak supervision techniques [17]. Transfer learning techniques, such as domain and task adaptation, minimize the distance across domains and reuse features already learned in $\mathcal{D}_S$ for $\mathcal{D}_T$. It is assumed that $\mathcal{D}_T$ and $\mathcal{D}_S$ have different yet related distributions [18]. The literature has demonstrated the effectiveness of transfer learning in various domains (including remote sensing) and tasks. For instance, Xie et al. [16] use transfer learning and weak supervision to predict poverty classification maps with scarce labeled satellite imagery. Yao et al. [19] proposed transfer learning to segment high-resolution satellite images in a weak supervision framework. Although the promising results, a transfer learning framework has never been defined for RS data, where the absence of labeled data is one of the biggest challenges when using DL techniques. $\mathcal{D}_T$ and existing $\mathcal{D}_S$ are significantly different, and RS domain-specific solutions must be designed to adhere to RS characteristics and tasks.

When analyzing RS data with DL, it is necessary to consider several challenges. Although some are common to other domains, others are unique to RS data analysis: i) the lack of reliable and large labeled training sets to be robust against overfitting. The overfitting risk increases when training a network from scratch (i.e., with random weight initialization) [10]. ii) The limited available information because of the one-dimensional channel. iii) Datasets consist of radargrams acquired in different campaigns by different sensors, where the same subsurface target can appear with different radiometric characteristics [20]. iv) The rate signal vs. noise-limited areas in the data is notably more leveled than other remote sensing data [8], and v) there is a significant unbalance of the prior probabilities of the geological target classes that penalize the characterization of less frequent (yet relevant) classes. Several approaches have been proposed in the literature to solve some of the listed challenges. Donini et al. [21] proposed a two-step training of the network and a massive data augmentation to deal with the lack of labeled data. However, data augmentation does not avoid overfitting when there is low variability in the RS training dataset. Without variability in the training dataset, the features learned by the CNN have limited generalization capability to other RS campaigns. The method presented in [22] employs a very deep CNN that segments radargrams following a training approach similar to [21]. However, this method shows limitations in identifying classes with low prior probabilities and dealing with RS data acquired in different campaigns. Widely used computer vision techniques (including transfer learning) that successfully handle these issues ask for an understanding of the radar sounder data properties, e.g., signal and noise distributions. Nevertheless, only one work has briefly explored transfer learning techniques for radar sounder data [23] that lacks the generalization to a framework for the RS domain and does not completely exploit transfer learning. This results in a sub-optimal solution showing poor discrimination of the low prior probability classes.

This work proposes a novel framework for transfer learning in the RS domain to perform radargram weakly supervised segmentation, i.e., with a small amount of labeled data. We propose two transfer learning techniques based on deep learning that address the pronounced lack of labeled RS data to segment radargrams. The proposed techniques identify and segment radargrams into five classes: free space, inland ice layering, floating ice and crevasses, bedrock, and noise-limited regions (thermal noise and echo-free zone). This definition includes new regions and classes compared to state-of-the-art approaches that add high complexity to the RS data analysis. In particular, it considers crevasses and floating ice classes and the need to differentiate them from inland ice. However, these classes increase the variability and complexity of the data because of the inherent noise and artifacts in the RS data and being unbalanced. To handle these aspects, we design two methods that work in two operating conditions: i) supervised pre-training in a domain other than RS and ii) unsupervised pre-training with RS data.

The first approach extends the initial idea in [22] and pre-trains a lightweight CNN in a domain other than RS, where labeled data can be easily retrieved (e.g., multimedia labeled dataset) to perform a task different from radargram segmentation, such as image classification (i.e., assign a label for each image). Then, transfer learning is designed to perform domain and task adaptation and reuses the pre-trained CNN to handle RS data and perform radargram segmentation. The second approach develops the idea in [23]. It employs unsupervised learning to pre-train a very deep CAE in the RS domain using complex radargram reconstruction as a proxy task for radargram segmentation. No labeled radargrams are needed to pre-train the network weights in the RS domain. Here, transfer learning is designed to perform task adaptation and reuse the pre-trained network. Both architectures are fine-tuned using weak supervision with a small amount of labeled data of the target domain (i.e., radargrams) with a segmentation task.

We test both methods on MCoRDS-1 and MCoRDS-3 data acquired in Antarctica, imaging ice sheets and shelves. We evaluate the transfer learning effectiveness by performing several experiments varying the training set for the pre-training

and the fine-tuning in terms of size (dramatically reducing the pre-training and fine-tuning training datasets until using only about a hundred samples), variability (with or without data augmentation), and level of correlation between the data (training with campaigns similar or different than those for testing).

The paper is organized as follows: Section II formulates the problem. Section III describes the proposed framework and methods. The datasets are described in Section IV. The experimental results are in Section V. Finally, Section VI concludes the paper and presents future works.

II. Problem formulation

Let us define a radargram $\mathbf{R}$ as a 2-dimensional matrix having n_T traces (columns) in the azimuth or along-track direction, and n_S samples (rows) in the range direction:

$$\mathbf{R} = \{R(x,y)|x \in X = [1,...,n_T], y \in Y = [1,...,n_s]\}, \quad (1)$$

where R is the power of the backscattered echoes stored in the log scale, and (x, y) are the azimuth and range coordinates, respectively. The radargram $\mathbf{R}$ contains the reflected signal from the subsurface geology. It is affected by noisy, and clutter contributions (e.g., thermal noise, echo-free zone, speckle noise, multiple reflections, and off-nadir reflections) [24].

Cryosphere radargrams image the ice sheets in the continental areas and the ice shelves floating on the ocean, as shown in Fig. 1. This work defines a complex segmentation problem modeling for ice shelves and continental ice. The problem complexity relies on i) differentiating the two icy bodies since they present significant variability and complexity caused by the inherent noise and artifacts in the RS data and ii) the increasing number of target classes compared to other literature approaches that only consider inland targets (ice layers, bedrock, basal ice, noise-limited regions) [8], [21] and do not consider coastal targets (floating ice and crevasses). In cryosphere RS data, the first reflection in the range is from the surface and has the most powerful signal. This interface delimits the ice pack and the free space above the surface. Below the surface, the signal behavior varies significantly, being grounded or floating. On inland ice, the ice stratigraphy generates bright lines due to variations of the ice dielectric coefficient [25]. The deepest reflection is caused by the basal interface that can be rocky or liquid [7]. The basal interface appears as a peak in the backscattering, reflecting the rest of the incident signal. Finally, in continental ice radargrams, the background (i.e., no backscattered signal) or noise-limited regions extend above the bedrock but below the ice pack (EFZ [26]) and below the bedrock (thermal noise). Both areas have statistical properties similar to the thermal noise added by the receiver. Radargrams of coastal areas image floating ice and ice crevasses. Crevasses are vertical fractures of the ice shelf generated by the ice movements and calving [27]. Crevasses appear in radargrams as bright vertical reflections caused by the propagation of the EM signal through the fractures (see Fig. 1). In coastal radargrams, the interface between the floating ice and the ocean or the crevasses in the absence of floating ice represent the deepest reflection. Finally, in coastal radargrams, the noise-limited (background) regions are below the floating ice from the seawater (see Fig. 1).

Propitiated by the radargram characteristics, CNNs have several challenges when analyzing RS data. Radargrams capture in one single channel the backscattering properties of targets in terms of type, shape, orientation, dielectric, chemical, and mechanical characteristics of scatterers in the minimum resolution cell of the sensor. Further, subsurface targets are mostly thin and elongated, and widely extended in along-track direction (e.g., layers and bedrock) [21]. One single radargram size is substantial and significantly longer in the along-track dimension (up to two orders of magnitude more than the range dimension) due to the acquisition approach. Thus, radargrams must be divided into patches to fit a CNN architecture input. However, the commonly used small square patches in other remote sensing fields risk not containing enough context information about the subsurface targets (i.e., missing basal information, near-surface information, or both), resulting in imprecise learned features by the CNN. Further, contiguous patches have a significantly low statistical intra-radargram (i.e., same acquisition campaign) variability since they contain similar information. Patch division becomes a critical aspect due to the elongated and extended distribution of radar sounder targets (e.g., bedrock), extending from one patch to the neighboring one without interruption and thus without borders when working on a patch-based mechanism. The outcome should preserve target continuity in the along-track direction as (dis)continuity has geological relevance. Accordingly, the design of data-hungry automatic techniques for analyzing RS data and the use of deep learning patch-based architectures is a recent research field.

The described radargram target behavior leads to the background (i.e., no backscattered signal, where the thermal noise is predominant) being the most frequent feature in the radargrams, dramatically limiting the informative areas [8], [28]. Therefore, subsurface targets have a small prior probability (i.e., one order of magnitude lower than thermal noise or background prior probability). The amount of information content is further limited by radargrams being single-channel data. These characteristics make radar sounder data significantly different from the optical ones mainly used to train deep learning architectures (e.g., RGB pictures from digital cameras). Fig. 3 illustrates the difference between an RGB image used in the pre-training step and a radargram used in fine-tuning. In addition, radar sounder data are strongly affected by several unwanted signals, including clutter, and types of noise, such as speckle noise that can be approximated as multiplicative. Speckle results in salt and pepper pattern with significant radiometric intensity variability and makes target borders, contours, and edges hard to distinguish. A CNN needs a large amount of labeled data to efficiently extract features from the subsurface targets in a supervised way. RS labeled data are unavailable due to the difficulties of generating an accurate labeled RS dataset. Expert photointerpretation is the only feasible way to generate these labeled RS data datasets.

The amount of radar sounder data has slowly increased since the acquisition campaigns are mainly aircraft-mounted

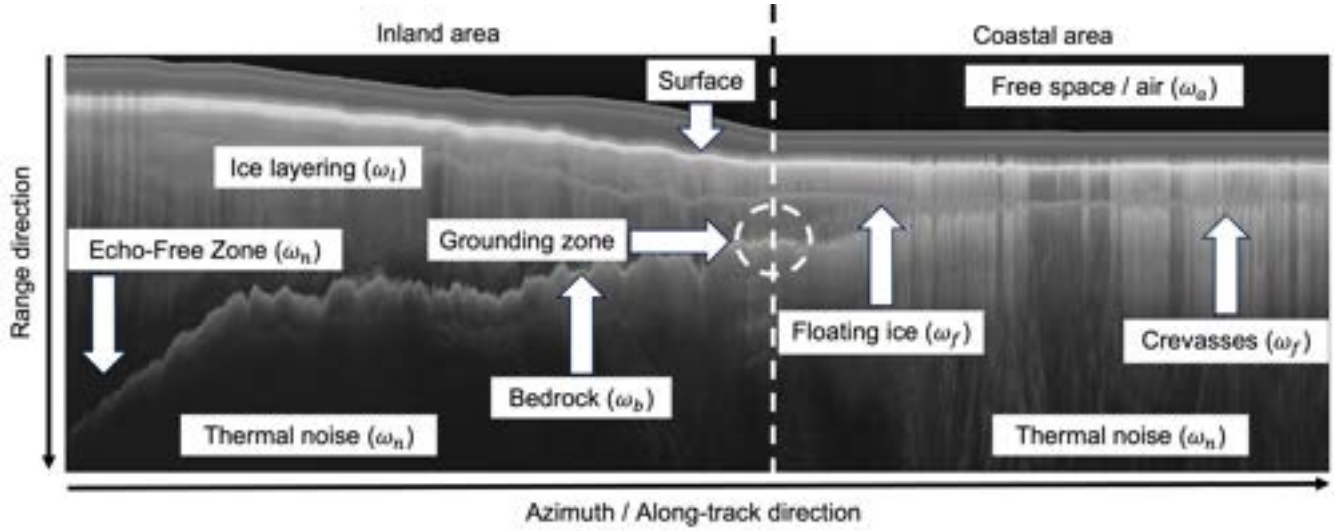


Fig. 1. Detail of a radargram acquired in Antarctica by MCoRDS-3 imaging the main subsurface targets of the problem of interest.

and thus cover small regions. Every acquisition campaign is specifically designed for the target area and features. The strategy of ad hoc campaigns leads to significant variability in terms of RS sensor architectures and configurations. This causes the number of radargrams to be relatively small compared to data in other domains and their inter-radargram (i.e., different acquisition campaigns) variability to be considerably high. Furthermore, the acquisition conditions (e.g., weather conditions, geometry, aircraft speed), radar calibration, or data post-processing strongly affect the radargram radiometric properties of different campaigns even when acquired by the same sensor [20]. For example, the noise and artifacts differently degrade the signal when differentiating floating and inland ice, depending on the acquisition campaign (e.g., acquisition conditions, radar calibration) or the sensor.

In this context, reusing existing architectures without adaptation or training from scratch results in poor performance, strong overfitting, limited generalization capability, and substantial underestimation of low prior probability (yet highly relevant, like bedrock) classes [10]. Moreover, the radiometric differences between data acquired with the same sensor evidence the necessity of testing the methods with data acquired in campaigns not used for training to reduce the intra-data correlation and demonstrate the method's generalization effectiveness.

Therefore, the objective is to segment a radargram $\mathbf{R}$ into $N = 5$ classes ($\mathcal{V}_T$): free space (ν_{fs}), inland ice layering (ν_l), floating ice and crevasses (ν_{fl}), bedrock (ν_b), and noise limited areas that include the thermal noise and EFZ (ν_n). The classes $\mathcal{V}_T = \{\nu_{T_a}, \nu_{T_s}\}$ can be divided between above surface free space ν_{T_a} and subsurface classes $\nu_{T_s} = \{\nu_{T_2}, ..., \nu_{T_N}\}$). Let $\mathcal{D}_T$ be the RS target domain, and $\mathcal{T}_T$ as the radargram segmentation target task. The source domain $\mathcal{D}_S$ and the source task $\mathcal{T}_S$ will be defined differently for each method.

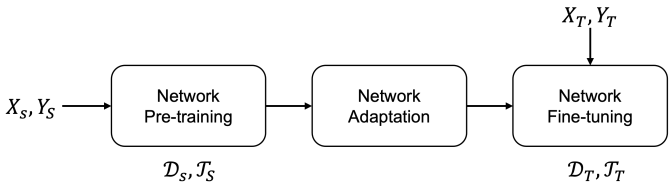


Fig. 2. Block scheme for the proposed approaches.

III. Proposed Framework and Methods

This section introduces the proposed transfer learning framework and the two automatic methods for radargram segmentation and a critical comparison between them, exposing their advantages and disadvantages.

The two proposed methods automatically segment radargrams by employing transfer learning techniques, including domain and task adaptation, that allow reusing a pre-trained CNN without using labeled RS data. Both approaches are divided into three steps, as shown in Fig. 2:

1) The network is pre-trained in the source domain ($\mathcal{D}_S$) to perform the source task ($\mathcal{T}_S$) to compensate for the lack of labeled data in the RS domain ($\mathcal{D}_T$) to perform radargram segmentation ($\mathcal{T}_T$).
2) The network is adapted to perform the target task ($\mathcal{T}_T$).
3) Finally, the network is fine-tuned in $\mathcal{D}_T$ to extract semantically meaningful features of RS data and segment radargrams in the target classes $\mathcal{V}_T$.

A. Domains definition

$\mathcal{D}_S$ is formed by a feature space $\mathcal{X}_S$ and the related probability distribution $P(X_S)$, where X_S is the set of labeled training samples $X_S = \{x_{S_1}, x_{S_2}, ..., x_{S_r}\} \in \mathcal{X}_S$. Thus, given a source domain $\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\}$, the source task $\mathcal{T}_S$ associated with $\mathcal{D}_S$ can be defined by the label space $\mathcal{V}_S$ and the predictive function $f_S(\cdot)$. The label space $\mathcal{V}_S = \{\nu_{S_1}, \nu_{S_2}, ..., \nu_{S_M}\}$ consists of the labels of the training

samples X_S. The predictive function $f_S(\cdot)$ is hidden attribute that predicts a label $y_{S_i} \in \mathcal{V}_S$ corresponding to a given sample $x_{S_i} \in X_S$. Therefore, the source task can be defined as:

$$\mathcal{T}_S = \{\mathcal{V}_S, f_S(\cdot)\}. \tag{2}$$

From a probabilistic viewpoint, $f_S(x)$ can be expressed as $P_S(y_S|x_S)$. Hence, the source domain data D_S can be defined as:

$$D_S = \{(x_{S_1}, y_{S_1}), ..., (x_{S_r}, y_{S_r})\}. \tag{3}$$

The two proposed approaches will differ in terms of the source domain $\mathcal{D}_S$ and the source task $\mathcal{T}_S$.

Let us define the target domain $\mathcal{D}_T$ similarly to $\mathcal{D}_S$: $\mathcal{D}_T = \{\mathcal{X}_T, P(X_T)\}$. $\mathcal{D}_T$ consists of the feature space $\mathcal{X}_T$ and the related probability distribution $P(X_T)$. Analogously, the set of training samples X_T is defined as $X_T = \{x_{T_1}, x_{T_2}, ..., x_{T_r}\} \in \mathcal{X}_T$. In this paper, the target domain $\mathcal{D}_T$ refers to radargrams, and each x_{T_i} is defined as patches extracted from $\mathbf{R}$. The target task $\mathcal{T}_S$, which is radargram segmentation, can be defined as:

$$\mathcal{T}_T = \{\mathcal{V}_T, f_T(\cdot)\}, \tag{4}$$

where $\mathcal{V}_T$ represents the segmentation classes and $f_T(\cdot)$ is the function that segments the radargram in the classes $\mathcal{V}_T$. Thus, $f_T(\cdot)$ can be expressed as $P(y_T|x_T), y_T \in \mathcal{V}_T$ and $Y_T = \{y_{T_1}, y_{T_2}, ..., y_{T_r}\} \in \mathcal{V}_T$, therefore the target domain data is:

$$D_T = \{(x_{T_1}, y_{T_1}), ..., (x_{T_r}, y_{T_r})\}. \tag{5}$$

B. Case I: lightweight CNN with $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S \neq \mathcal{T}_T$, and supervised pre-training

Case I uses a lightweight CNN that is pre-trained in a source domain $\mathcal{D}_S$ to perform a source task $\mathcal{T}_S$, to set the network weights and improve the performance of the network [11]. In this approach, the source domain $\mathcal{D}_S$ presents different characteristics, such as a higher number of channels and distinct statistical distribution P_S, so $\mathcal{D}_S \neq \mathcal{D}_T$ (as seen in Fig. 3 the differences between the pre-training image and the fine-tuning radargram are significant). We apply transfer learning for domain and task adaptation following the pre-training configuration. We introduce transfer learning techniques to reuse the pre-trained CNN weights and adapt the CNN architecture to analyze radargrams, $\mathcal{D}_S$, and perform the target task $\mathcal{T}_T$. We add a convolutional layer at the beginning of the architecture to perform the domain adaptation from $\mathcal{D}_S$ to $\mathcal{D}_T$. We discard the last layers of the pre-trained CNN that identify the most specific features from $\mathcal{D}_S$ [10], resulting in a reduced pre-trained CNN. The reduced CNN is included in a U-fashion architecture as an encoder, while the decoder consists of several up-convolutional layers. The up-convolutional layers identify the target classes $\mathcal{V}_T$ in the target domain $\mathcal{D}_T$ to perform the target task $\mathcal{T}_T$.

1) Supervised pre-training in D_S: In this step, the network is trained in the source domain $\mathcal{D}_S$ to perform a source task $\mathcal{T}_S$ as can be seen in Fig. 3. Choosing the source domain $\mathcal{D}_S$ and the task $\mathcal{T}_S$ depend on i) the availability of reliable labeled data in the source domain and ii) the adaptability of the approach from working in the source domain to the target domain.

In this approach, the source domain differs from the target domain $\mathcal{D}_S \neq \mathcal{D}_T$. Therefore the source domain has its specific characteristics:

$$\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\} \tag{6}$$

and the source task differs from the target task $\mathcal{T}_S \neq \mathcal{T}_T$. The source task is defined as:

$$\mathcal{T}_S = \{\mathcal{V}_S, f_S(\cdot)\}, \tag{7}$$

where we have a different set of classes $\mathcal{V}_S \neq \mathcal{V}_T$, the source data domain consists of a set of training samples and their labels that are different from the target data domain $\mathcal{D}_S \neq \mathcal{D}_T$.

Here, we pre-train the lightweight CNN in the source domain $\mathcal{D}_S$ of visual recognition with 3-channel RGB multimedia images. Deep learning for visual recognition is currently in a very advanced stage of development. Among the available approaches, we can choose one optimized with a larger number of training data. A robust training is critical in this case since the better and more general the extracted features from D_S are, the more accurate results will be reached in the target domain $\mathcal{D}_T$ even if we have few labeled radargrams for transfer learning [11].

2) CNN adaptation and fine-tuning in $\mathcal{D}_T$ (see Fig. 3): The proposed method adapts the pre-trained lightweight CNN to the radargram characteristics ($\mathcal{D}_T$) into a new architecture with a U-fashion shape (Fig. 4). Two adaptation aspects should be considered: adaptation to the radargram characteristics ($\mathcal{D}_T$) and adaptation to the radargram segmentation task ($\mathcal{T}_T$).

Adaptation to the radargram characteristics. This step handles the source and target domain differences. $\mathcal{D}_T$ has i) one channel instead of 3 channel RGB multimedia data used for pre-training, and ii) different marginal probability with respect to the multimedia data $P(X_T) \neq P(X_S)$, which means different statistical distribution of the features [20]. Since a radargram is a one-channel image representation, we assume that a graphic representation of a radargram $\mathcal{X}_T$ is a subgroup of an RGB multimedia image $\mathcal{X}_S$. To learn and compensate for the difference between the domains, we include a 2D convolution layer at the top of the pre-trained lightweight CNN. This layer performs the domain adaptation function $f_{da}(\cdot)$ to fit the target domain into the source domain properties expected in input by the pre-trained CNN. It handles the radar sounder data-specific characteristics, such as the statistical distribution and features of the target classes, and does the 1-to-3 channel conversion as:

$$D_S = \{(f_{da}(x_{T_1}), y_{S_1}), ..., (f_{da}(x_{T_q}), y_{S_q})\}. \tag{8}$$

Adaptation to the radargram segmentation. Here, we apply transfer learning to make the pre-trained lightweight CNN segment the input radargrams into the $\mathcal{V}_T$ classes reusing the pre-trained CNN weights. The network is adapted to extract relevant features for the target task $\mathcal{T}_T$. We discard the pre-trained CNN layers that extract the most specific features to the source domain $\mathcal{D}_S$ and task $\mathcal{T}_S$. These are the last layers in the CNN architecture [29]. The modified architecture has the reduced pre-trained lightweight CNN as the encoder. Therefore we have a new reduced task $\mathcal{T}_{red} = \{\mathcal{V}_{red}, f_{red}(\cdot)\}$, a new

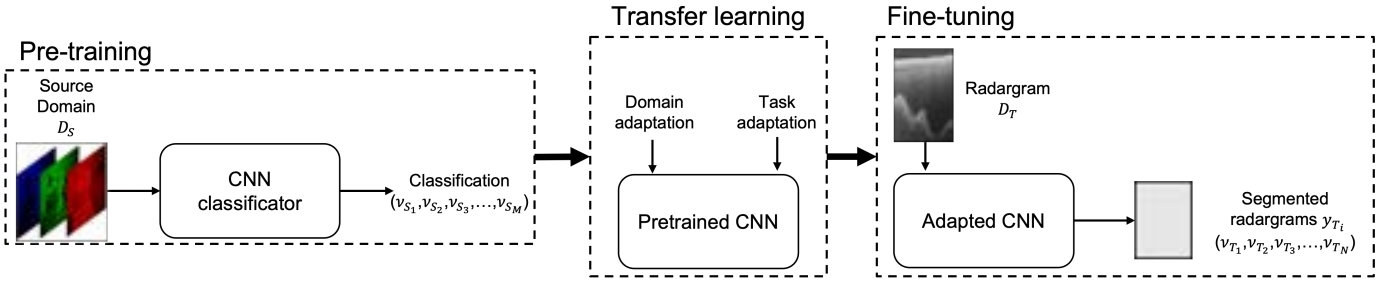


Fig. 3. Block scheme for Case I.

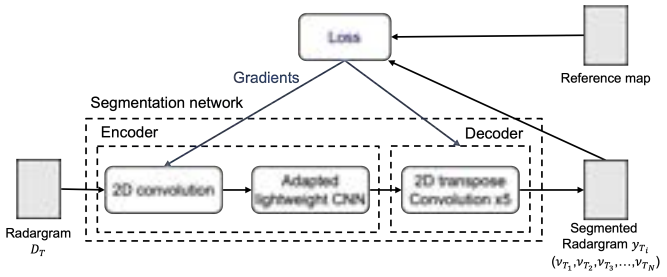


Fig. 4. Block scheme of the proposed new architecture for Case I.

label space ($\mathcal{V}_{red}$), and a new predictive function ($f_{red}(\cdot)$), which is $P_{red}(y_{red}|f_{da}(x_T)), y_{red} \in \mathcal{V}_{red}$. We introduce these changes in (3):

$$D_S = \{(f_{da}(x_{T_1}), y_{S_{red_1}}), ..., (f_{da}(x_{T_q}), y_{S_{red_q}})\}. \quad (9)$$

Due to the pre-trained lightweight CNN architecture, the new output features are significantly smaller than the input radargram. Since we aim to segment the input radargrams, we add five up-convolutional layers acting as a decoder to increase the size of the output features and learn the specific features of $\mathcal{D}_T$. Additionally, each up-convolutional layer in the decoder is shortcutted with a matching size layer in the encoder to facilitate gradient propagation. Fig. 4 shows the architecture after adaptation, where the contracting path or encoder consists of the reduced lightweight CNN, and the up-convolutional layers form the upsampling path or decoder. The decoder assigns to each pixel of the radargram one of the N classes in $\mathcal{V}_T$, acting as $f_T(\cdot)$. The target task becomes:

$$\mathcal{T}_T = \{\mathcal{V}_T, f_T(f_{S_{red}}(\cdot))\}. \quad (10)$$

Subsequently, the network layers added in this step are fine-tuned with a small amount of labeled data to set the network weights to extract semantically meaningful features from the pixels for the segment contracting path of the final architecture or encoder, which are not updated in this step. Instead, the fine-tuning loss updates the new convolutional layer weights. Only the pre-trained weights in $\mathcal{D}_S$ remain unchanged to avoid over-fitting due to the few labeled data of the target domain.

The batch normalization layers typically used in the CNN for visual recognition are counterproductive. Batch normalization adds extra noise [30] and is less effective with small batches or does not contain uncorrelated samples [31], which is the case of radar sounder data. Because of this, instead of using batch normalization layers, we use instance normalization layers, which are not affected by the small batch size or little variability of $\mathcal{D}_T$ [32].

Due to the extremely unbalanced appearance of classes in $\mathcal{D}_T$, we introduce weight maps to increase the relevance of the less frequent classes otherwise under-considered in training. The less frequent classes tend to be less differentiated, given their small impact on the overall error. Hence, the weight of each class $w_{\nu_{T_i}}$ considers the class priors in Y_T:

$$w_{\nu_{T_i}} = \frac{N_p}{N_{\nu_{T_i}} N}, \quad (11)$$

where $N_{\nu_{T_i}}$ indicates the pixel number of ν_{T_i} in Y_T. Finally, $w_{\nu_{T_i}}$ conditions the fine-tuning loss $L_{fine-tuning}$, used to learn the target task $\mathcal{T}_T$. $L_{fine-tuning}$ is defined considering the sparse categorical cross-entropy loss:

$$L_{fine-tuning} = -\frac{1}{N_p} \sum_{i=1}^{N_p} \sum_{n=1}^{N} B_{y_{T_i},\nu_{T_n}} \cdot log(f_T(x_T)_{i_{\nu_{T_n}}}) w_{\nu_{T_n}}, \quad (12)$$

where $B_{y_{T_i},\nu_{T_n}}$ is a binary indicator and is set to $B_{y_{T_i},\nu_{T_n}} = 1$ when $y_{T_i} = \nu_{T_n}$. $f_T(x_T)_{i_{\omega_n}}$ is the predicted probability of the class ν_{T_n} being the pixel i.

C. Case II: deep CAE with $\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S \neq \mathcal{T}_T$, and unsupervised pre-training

In Case II, the source task $\mathcal{T}_S$ is radargram reconstruction, so the deep CAE is pre-trained with unlabeled radargrams, which belong to the target domain ($\mathcal{D}_S = \mathcal{D}_T$). Therefore, the network learns specific radargram features, and the weights are not randomly initialized. The unsupervised pre-training of the deep CAE aims to update its weights by applying two loss functions, i.e., the reconstruction L_r and the above/subsurface $L_{a/s}$ losses. To perform the reconstruction task, a deep CAE extracts meaningful features for all pixels in the input data. The deep CAE consists of two U-shape architectures concatenated, acting as encoder and decoder in the deep CAE. We aim the encoder to segment radargrams, $\mathcal{T}_T$, and the decoder to reconstruct the original radargram from the segmentation map, which is the source task $\mathcal{T}_S$. As the network is pre-trained with radargrams, transfer learning in this approach consists of reusing the pre-trained deep CAE and adapting the pre-trained network to perform the target task $\mathcal{T}_T$, radargram segmentation. Task adaptation comprises removing the CAE

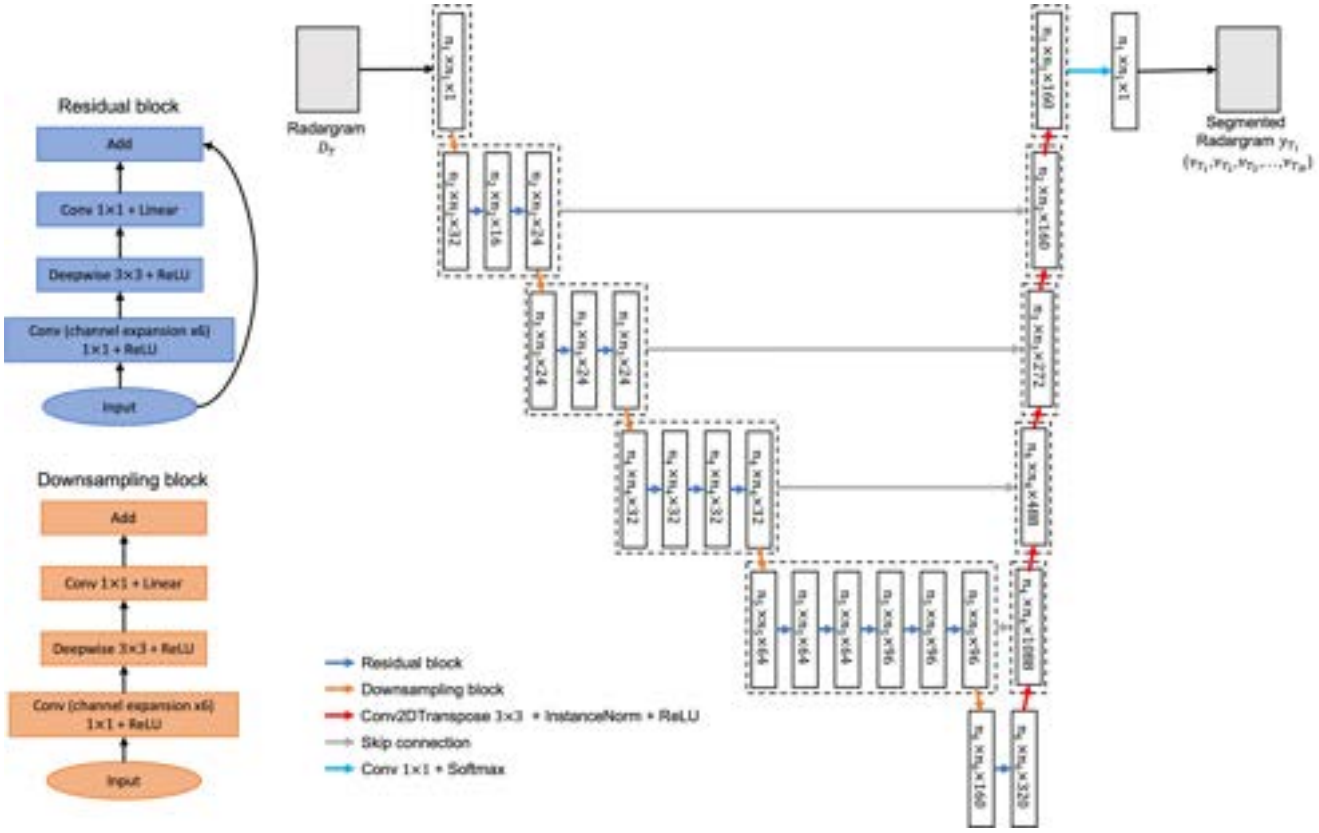


Fig. 5. Case I modified architecture.

decoder, only used to perform the source task $\mathcal{T}_S$. Similarly, the deep CAE encoder is already set up for the target task $\mathcal{T}_T$.

1) CAE unsupervised pre-training in $\mathcal{D}_T$: In the unsupervised pre-training step, the network is pre-trained in the source domain $\mathcal{D}_S$ to perform the source task $\mathcal{T}_S$. Since in this approach, the source domain consists of radargrams, $\mathcal{D}_S = \mathcal{D}_T$, we will only refer to target domain $\mathcal{D}_T$ (Fig. 6)

The source task $\mathcal{T}_S$ of this approach is radargram reconstruction:

$$\mathcal{T}_S = \{\mathcal{X}_T, f_S(\cdot)\}, \tag{13}$$

where $f_S(\cdot)$ is composed by the encoder predictive function $f_{ENC}(\cdot)$ and the decoder predictive function $f_{DEC}(\cdot)$: $f_S = f_{DEC}(f_{ENC}(\cdot))$. With the CAE that has two U-shape architectures concatenated, we aim that the output of the encoder $f_{ENC}(\cdot)$ predicts the probability of the segmentation labels $P(y_T|x_T), y_T \in \mathcal{V}_T$, and the decoder output $f_{DEC}(\cdot)$ predicts the values for the reconstructed radargram $P(x_T|x_T), x_T \in \mathcal{X}_T$. Since the output is predicted to be as similar as possible to the input data, $y_{S_i} \in \mathcal{X}_T$ and $y_{S_i} = x_{T_i}$, the unlabeled extended target data domain $D_{T_{\text{unl}}}$, including a large number of unlabeled radargrams, becomes:

$$D_{T_{\text{unl}}} = \{(x_{T_1}, x_{T_1}), ..., (x_{T_p}, x_{T_p})\}. \tag{14}$$

This allows the network to be trained with a large target domain dataset without labels and to learn the radargram-specific features in the unsupervised pre-training.

We define an additional unsupervised task driven by the encoder called above/subsurface task $\mathcal{T}_{a/s}$. This task aims to improve the radargram features, such as the target distribution, learned in the unsupervised pre-training step by learning to differentiate the regions above and below the surface in the radargram. The encoder output defines this task:

$$\mathcal{T}_{a/s} = \{\mathcal{V}_{a/s}, f_{ENC}(\cdot)\}, \tag{15}$$

Where the target labeled data are the mathematically generated above/subsurface mask $Y_{a/s} = \{y_{a/s_1}, y_{a/s_2}, ..., y_{a/s_p}\} \in \mathcal{V}_{a/s}$. The mask splits $\mathbf{R}$ into two classes ν_{T_a} and ν_{T_s} that indicate the area above the surface and the subsurface, respectively. The rough boundary between ν_{T_a} and ν_{T_s} is identified as the surface S, which is the reflection covering all the radargram in the azimuth direction with the highest power. Therefore, S is the row $[1, 2, ..., n_S]$ having the greatest power in each range line (column) $[1, 2, ..., n_T]$, i.e., $S(x) = \underset{y}{max}\{x_T(x, y)\}$ [8]. Given S, y_{a/s_i} is generated without supervision:

$$y_{a/s_i}(x, y) = \begin{cases} 0, & if\ 1 < y < x_T(x, y) = S(x) \\ 1, & elsewhere \end{cases} \tag{16}$$

So we can define the above/subsurface data domain as:

$$D_{a/s} = \{(x_{T_1}, y_{a/s_1}), ..., (x_{T_r}, y_{a/s_p})\}. \tag{17}$$

The above/subsurface task $\mathcal{T}_{a/s}$ is learned by the encoder minimizing the above/subsurface loss $L_{a/s}$. In the unsupervised pre-training, $\mathcal{T}_{a/s}$ drives the gradient calculation to optimize the encoder weights. $L_{a/s}$ is defined to estimate the difference between the radargram segmentation map predicted by the encoder $f_{ENC}(x_{T_i})$, and the above/subsurface mask y_{a/s_i}.

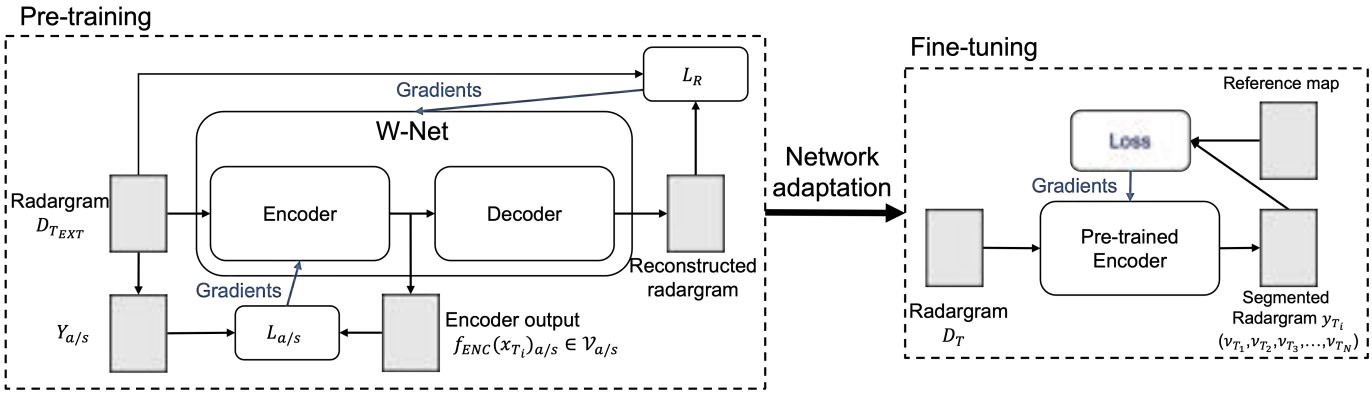


Fig. 6. Block scheme for Case II approach.

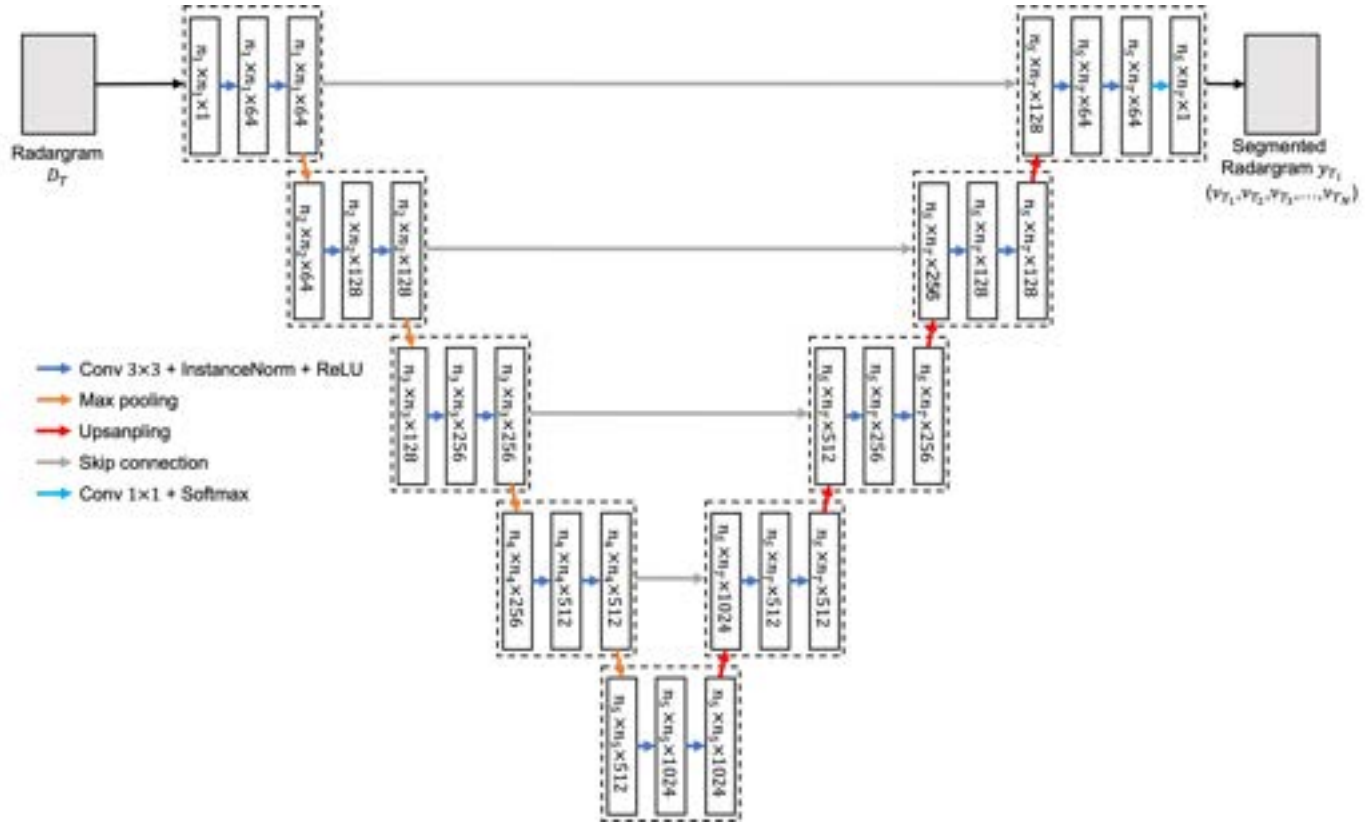


Fig. 7. Case II modified architecture.

Therefore, $L_{a/s}$ is defined as the binary cross entropy loss between $f_{ENC}(x_{T_i})$ and $f_{ENC}(x_{T_i})$. Let us assume that the subscript i indicates position (x, y) in the radargram:

$$L_{a/s} = -\frac{1}{N_p}\sum_{i=1}^{N_p} y_{a/s_i} \cdot log(f_{ENC}(x_{T_i})_{a/s}) + (1 - y_{a/s_i}) \cdot log(1 - f_{ENC}(x_{T_i})_{a/s}), \quad (18)$$

where $f_{ENC}(x_{T_i})_{a/s} \in \mathcal{V}_{a/s}$ and it is the encoder output where the prior probabilities of the subsurfaces classes ($\nu_{T_s} = \{\nu_{T_2}, ..., \nu_{T_N}\}$) predicted by the encoder are joined into a unique class ν_{T_s}. The number of pixels in x_T is defined as $N_p = n_S n_T$.

Subsequently, all the CAE weights are updated to efficiently reconstruct the input radargram x_T, which is the source task $\mathcal{T}_S$, by optimizing a $\mathcal{L}_2$ loss called reconstruction loss L_r. L_r expresses the error between the input radargram x_T and the reconstructed radargram by the autoencoder $f_S(x_T)$:

$$L_r = ||x_T - f_S(x_T)||^2. \quad (19)$$

In Case II, we also use instance normalization layers instead of batch normalization layers similar to Sec. III-B2.

2) CAE task adaptation and supervised refinement in $\mathcal{D}_T$: In this step, as the CAE is pre-trained in the target domain $\mathcal{D}_T$, we only have to adapt the network architecture to perform the radargram segmentation target task $\mathcal{T}_T$.

Since the predictive function in charge of the radargram reconstruction was f_{DEC}, the decoder is discarded, and the encoder is employed to perform $\mathcal{T}_T$. In this approach, the pre-trained U-fashion encoder segments the radargram into the N classes in Ω (as shown in Fig. 3), and we define the target task as:

$$\mathcal{T}_T = \{\mathcal{V}_T, f_{ENC}(\cdot)\}, \tag{20}$$

where $f_{ENC}(\cdot)$ predicts the segmentation labels for the input and can be expressed as $P(y_T|x_T), y_T \in \mathcal{V}_T$.

Finally, the encoder weights are updated during fine-tuning using $L_{fine-tuning}$ (12). The encoder learns specific features from the target classes $\mathcal{V}_T$. In this way, the encoder learns the target task $\mathcal{T}_T$.

D. Critical comparison of the two methods

Both methods require pre-training due to the lack of labeled radargrams D_T. Transfer learning is applied to reuse and adapt the pre-trained networks to perform radargram segmentation. Both proposed methods are fine-tuned with a few labeled radargrams to learn the target task $\mathcal{T}_T$, radargram segmentation. The ad-hoc modifications introduced in Cases I and II make the proposed transfer learning approach specifically designed to analyze RS data. Hence, a readaptation process is necessary to use these logics in other domains.

The main differences between the two proposed approaches are i) the pre-training and ii) the network adaptation. The Case I lightweight CNN is pre-trained with ImageNet dataset D_S to perform image classification. Case I is less demanding than Case II pre-training because i) it is unnecessary to create a pre-training dataset specifically designed for this objective, which can be time-consuming, and ii) pre-trained CNN architectures are widely accessible to the scientific community. However, the CAE of Case II is pre-trained with unlabeled radargrams $D_{T_{\text{unl}}}$ to perform radargram reconstruction. $\mathcal{T}_S$ and $\mathcal{T}_T$ directly affect the architecture depth: Case II requires a deeper architecture than Case I to deal with the simultaneous reconstruction and the segmentation of the radargrams. Given the absence of labeled datasets in $\mathcal{D}_T$, Case II pre-training is more demanding than Case I. The network adaptation to perform radargram segmentation is more complex in Case I since the lightweight CNN is pre-trained in a different domain, and Case II CAE is pre-trained with radargrams. The lightweight CNN of Case I is computationally less expensive than Case II and requires time and computational resources for pre-training and fine-tuning.

IV. Dataset description

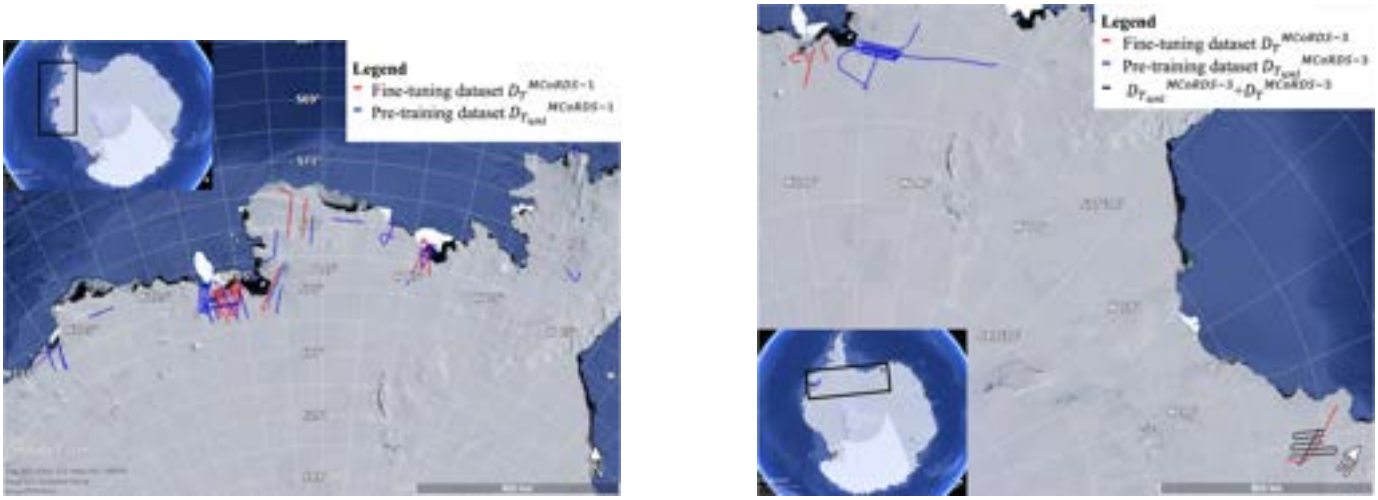
This section describes the datasets for assessing the effectiveness of the proposed methods. We consider two sets of radargrams acquired in Antarctica by MCoRDS-1 and MCoRDS-3 [33], respectively, that are distributed by the Center for Remote Sensing of Ice Sheets (CReSIS). MCoRDS-1 and MCoRDS-3 were mounted on a DC-8 aircraft, and their design parameters are presented in Table I. The data acquired by the two sensors have different properties in terms of range and azimuth resolution, and maximum penetration. The MCoRDS-3 range resolution is considerably higher than that of MCoRDS-1. This makes MCoRDS-3 radargrams more complex as they contain more details and show a larger size in the range direction than MCoRDS-1 radargrams. Fig. 8 shows the ground track of the campaigns used to generate the datasets. Since radargrams are acquired in the Antarctica coasts, they contain inland classes (free space/air, ice layering, bedrock, and noise-limited areas) and novel classes in the coastal area (e.g., floating ice and crevasses). However, the MCoRDS-1 dataset does not consider the EFZ since it is smaller than the data resolution. The MCoRDS-3 dataset is more complex than the MCoRDS-1 one. For MCoRDS-1, the prior probabilities $P(\cdot)$ are as follow: free space $P(\nu_{fs}) = 0.168$, inland ice layering $P(\nu_l) = 0.161$, floating ice and crevasses $P(\nu_{fl}) = 0.067$, bedrock $P(\nu_b) = 0.016$, and noise-limited areas $P(\nu_n) = 0.588$. For MCoRDS-3, the prior probabilities $P(\nu)$ are as follow: free space $P(\nu_{fs}) = 0.21$, inland ice layering $P(\nu_l) = 0.20$, floating ice and crevasses $P(\nu_{fl}) = 0.063$, bedrock $P(\nu_b) = 0.007$, and noise-limited areas $P(\nu_n) = 0.52$.

The radargrams are range compressed and focused in the azimuth with synthetic aperture radar (SAR) processing. Moreover, the fluctuations from the aircraft motion were corrected, and the power of each radargram was log-scaled to enhance the spatial properties and approximate as additive the noise. The radargrams were divided into one-channel patches of size 1536×64 pixels for MCoRDS-3 data and 512×64 pixels for MCoRDS-1, respectively. The range dimension is chosen to provide global contextual information to the CNN, facilitating the learning of the target spatial distribution in radargrams and maximizing the variability of the classes in the range direction. The patches are collected in datasets $D^j, j = [\text{MCoRDS-1}, \text{MCoRDS-3}]$ and normalized by scaling them in the range $[0, 1]$:

$$D^j = \frac{D^j - min(D^j)}{max(D^j) - min(D^j)}. \tag{21}$$

Finally, datasets $D^j, j = [\text{MCoRDS-1}, \text{MCoRDS-3}]$ are both divided into two sub-datasets: the unlabeled target domain dataset $D^j_{T_{\text{unl}}}$ and the labeled target domain dataset D^j_T. All the patches in D^j_T are manually labeled for the fine-tuning step. The labels are defined for each pixel in the patches by visual interpretation considering examples of radargrams manually analyzed that are available in the literature [21], [6], [8]. Note that ambiguous radargram pixels (e.g., those in the grounding area, in orange in Fig 9 and 11) are labeled as unknown and not considered in D^j_T. In both MCoRDS-1 and MCoRDS-3 datasets, the cardinality of $D^j_{T_{\text{unl}}}$ is considerably higher than that of D^j_T, i.e., $|D^j_{T_{\text{unl}}}| >> |D^j_T|$. Fig. 8 (a) and 8 (b) show the ground tracks of the campaigns in $D^j_{T_{\text{unl}}}$ in blue and those in D^j_T in red for MCoRDS-1 and MCoRDS-3 datasets, respectively. For the qualitative results, we show the segmentation maps on radargrams 025-026 in campaign 20091118_01 for MCoRDS-1 and radargrams 004-009 in campaign 20181015_01 for MCoRDS-3.

Pre-training datasets. For Case I, we choose as the source domain dataset (D_S) a multimedia dataset easily accessible:



(a) (b)

Fig. 8. Ground tracks of the MCoRDS-1 (a) and MCoRDS-3 (b) radargrams. The acquisition ground tracks for the radargrams in the unlabeled pre-training dataset $D^j_{T_{\text{unl}}}$ are shown in blue, for the fine-tuning labeled dataset in red D^j_T, and those used in both $D^j_{T_{\text{unl}}}$ and D^j_T are in black.

TABLE I
PARAMETERS OF MCORDS-1 AND MCORDS-3 [33] AND RELATED PROPERTIES OF THE RADARGRAMS.

Parameter	**MCORDS-1**	**MCORDS-3**
Central frequency (f_c)	193.5 MHz	190 MHz
Bandwidth (BW)	9.5 MHz	50 MHz
Transmitted power (P_{tr})	550 W	6000 W (1000W/channel)
Aircraft altitude (h)	7000 m	1500 m
Range resolution in Ice (R_r)	13.6 m	4.3 m
Along-track resolution (R_a)	25 m	27.5 m

TABLE II
DETAILS OF THE UNLABELED TARGET DOMAIN DATASET $D^j_{T_{\text{UNL}}}$.

Parameter	$D^{\text{MCoRDS-1}}_{T_{\text{EXT}}}$	$D^{\text{MCoRDS-3}}_{T_{\text{EXT}}}$
Number of campaigns	17	6
Number of radargrams	122	114
Number of traces (n_T)	254912	220672
Patch size ($h \times w \times c$)	512×64×1	1536×64×1
Number of patches (P)	3983	3448

ImageNet [34]. The dataset consists of 14 million RGB images of size $224 \times 224 \times 3$ divided into 1000 classes. Image classification is the source task ($\mathcal{T}_S$). For Case II, the pre-training is done with the extended unlabeled target domain datasets $D^j_{T_{\text{unl}}}, j = [\text{MCoRDS-1}, \text{MCoRDS-3}]$ (details of the data are in Table II).

Fine-tuning datasets. For each sensor, we define a labeled dataset $D^j_T, j = [\text{MCoRDS-1}, \text{MCoRDS-3}]$ (details in Table III) for the fine-tuning of both Cases I and II.

V. EXPERIMENTAL RESULTS

This section describes the architecture and experimental setup, the baseline to compare the proposed method, the evaluation metrics, and the segmentation results.

TABLE III
FINE-TUNING DATASET D_T.

Parameter	$D^{\text{MCoRDS-1}}_T$	$D^{\text{MCoRDS-3}}_T$
Number of campaigns	17	6
Number of radargrams	39	79
Number of traces (n_T)	122112	100736
Patch size ($h \times w \times c$)	512×64×1	1536×64×1
Number of patches (P)	1908	1574
Number classes (N)	5	5

A. Architecture setups

1) Case I setup: Here, we use the MobileNet V2 as a pre-trained lightweight CNN, which performs well in many segmentation and classification tasks [14] but any other lightweight CNN can be used. We choose this network as it has a low number of parameters and is optimal to avoid overfitting. The MobileNet V2 employs depthwise separable convolutions, a low computational cost convolution with a small trade-off in accuracy reduction [35]. In addition, the network takes advantage of inverted residual blocks, a variation of the traditional residual blocks that insert shortcuts between the bottlenecks to avoid transmitting non-linear transformations. This feature could be critical when working with radargrams $\mathcal{D}_T$ since nonlinearities could dramatically affect the data by changing radargram properties. Finally, as Mobilenet V2 is developed in the computer vision domain, available solutions are pre-trained with 3-channel RGB images to perform image classification task [14]. The MobileNet V2 has been pre-trained with ImageNet [34] and the categorical cross-entropy loss, following [14]. This pre-training aims to extract semantically meaningful features from the multimedia data that later can be extended to the radar data.

In the fine-tuning, we used transfer learning to adapt the network by i) adding one convolutional layer at the beginning of the architecture to perform domain adaptation, ii) removing

the last three layers of the CNN architecture, which are two 2D convolutional layers and one average pool layer, and iii) adding five up-convolutional layers at the top of the reduced CNN architecture following the steps described in Sec. III-B2. The proposed architecture is shown in Fig. 5, while the parameter setup is described in Table IV. We choose a small learning rate l_r to limit over-fitting due to the few labeled radargrams used for fine-tuning and the reduced number of layers to train. In fine-tuning, we only train the up-convolutional layers and the domain adaptation convolutional layer, the layers added following Sec. III-B2. The network is fine-tuned for 200 epochs, and we set a batch size of 16 and 4 for the MCoRDS-1 and MCoRDS-3 datasets, respectively.

2) Case II setup: As deep CAE, we consider the W-Net [12], an autoencoder consisting of two concatenated U-Nets [13] but any other W-Net-based architecture can be used. The W-Net is pre-trained for 20 epochs to perform the radargram reconstruction task with $D_{T_{\text{EXT}}}$ by optimizing the reconstruction loss L_r and the above/subsurface loss and $L_{a/s}$ as shown in Table IV. Note that the high number of network trainable weights makes this process computationally extremely expensive. We apply transfer learning to adapt and use the pre-trained W-Net to perform radargram segmentation by discarding the second U-Net that acts as the decoder (see Sec. III-C2 and Fig. 7). Case II CNN is fine-tuned for 200 epochs with $\mathcal{D}_T$ by optimizing $l_{fine-tuning}$ to perform the radargram segmentation task. Table IV shows the fine-tuning parameter setup. During fine-tuning, we decreased the learning rate l_r to 0.0001 since we only aim to slightly modify the pre-trained weights to learn the class-specific features. The CNN for fine-tuning has considerably less trainable weights than the pre-trained W-Net and more weights than the Case I CNN.

B. Evaluation metrics

To evaluate the performance of the proposed methods, we consider three metrics that compare the predicted segmentation map $f_T(y_{T_i})$ with the reference map y_{T_i}: the sensitivity, the specificity, and the accuracy. The sensitivity is the probability that a pixel is correctly classified as the class ν_i that the pixel belongs to. It is computed by dividing the number of true positive classified pixels TP_{ν_i} by the number of pixels belonging to class ν_i: TP_{ν_i} and FN_{ν_i}, which are the pixels not classified as class ν_i that actually belong to it:

$$Sensitivity = \frac{TP_{\nu_i}}{TP_{\nu_i} + FN_{\nu_i}}. \tag{22}$$

The specificity indicates the proportion of pixels not classified as ν_i that do not belong to ν_i. The specificity is calculated by dividing the true negative pixels of ν_i (TN_{ν_i}) by the total number of pixels not belonging to class ν_i: TN_{ν_i} and the false positive pixels of ν_i (FP_{ν_i}), which are the pixels classified as class ν_i that actually do not belong to it:

$$Specificity = \frac{TN_{\nu_i}}{TN_{\nu_i} + FP_{\nu_i}}. \tag{23}$$

The overall accuracy (OA) that is calculated by dividing the correctly classified pixels by the network C_{ν_i} by the total number of pixels N_p:

$$OA = \frac{C_{\nu_i}}{N_p} \cdot 100\%. \tag{24}$$

C. Experimental setups

To prove the effectiveness of the proposed methods, we evaluate and compare the performance of the approaches under different conditions. We set up experiments by changing the training strategy to assess the impact of transfer learning and fine-tuning dataset size, the intra-data correlation, and the data variability. To assess the impact of the size of the training set, we perform several experiments by varying the fine-tuning training set size. Both fine-tuning datasets D_T^j are divided into validation set (20% of D_T^j), test set (20% of D_T^j), and training set D_{Tr}^j (60% of D_T^j). In the experiments, we varied the fine-tuning dataset for training to take about 100%, 84%, 67%, 17%, and 8% of D_{Tr}^j (60%, 50%, 40%, 10%, 5% of D_T^j, respectively). To evaluate the effect of the data intra-correlation, we propose two configurations for fine-tuning. Similarly to [21], in Configuration I, the training and test sets consist of patches extracted from the same campaigns. In Configuration II, the patches of the training set are extracted from campaigns different than the test set to reduce the intra-data correlation. Configuration II is novel compared to literature and closer to an operative scenario. Regarding variability, we train the network with and without data augmentation to estimate its impact when using a small labeled dataset for fine-tuning. As data augmentation techniques, we chose transformations that do not affect the integrity of the data and maintain a realistic geologic appearance. We apply random horizontal flips and crops, and we use random elastic deformation [36], [13], demonstrated to be effective in radar sounder data [21].

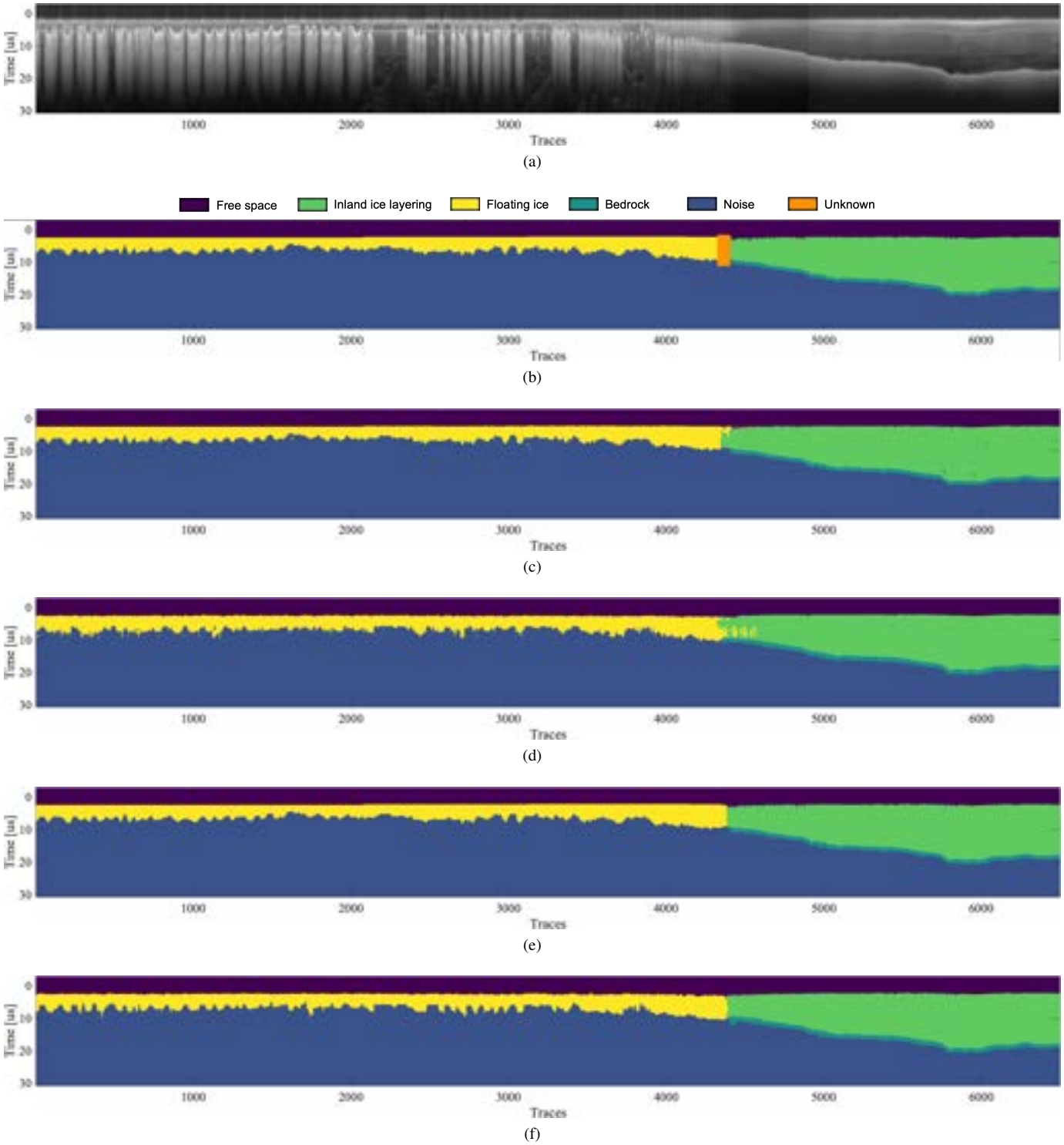
Let us define four experiments:

1) Experiment I: considers data augmentation and reduces the fine-tuning set size (from 8% to 100% of D_{Tr});
2) Experiment II: is the same as Experiment I without data augmentation;
3) Experiment III: is the same as Experiment I without transfer learning;
4) Experiment IV: considers data augmentation and reduces the pre-training set size $D_{T_{\text{unl}}}$ from 100% to 0% of $D_{T_{\text{unl}}}$, the fine-tuning set size fixed to 100% of D_{Tr}.

1) Baseline method: We compare the proposed methods with two approaches in the literature that segment radargrams based on i) the SVM classifier [8] and ii) DL [21]. We also compared the performance of an unadapted (e.g., no transfer learning, weighted and above/subsurface losses, instance normalization, nor the use of vertical patches) state-of-the-art CNN architecture used for image segmentation, the DeepLabV3+ [37], to further test the effectiveness of the techniques proposed in this work.

D. Segmentation results

1) MCoRDS-1 dataset: The proposed method qualitatively segments the radargrams with good overall accuracy (see



(a)

(b)

(c)

(d)

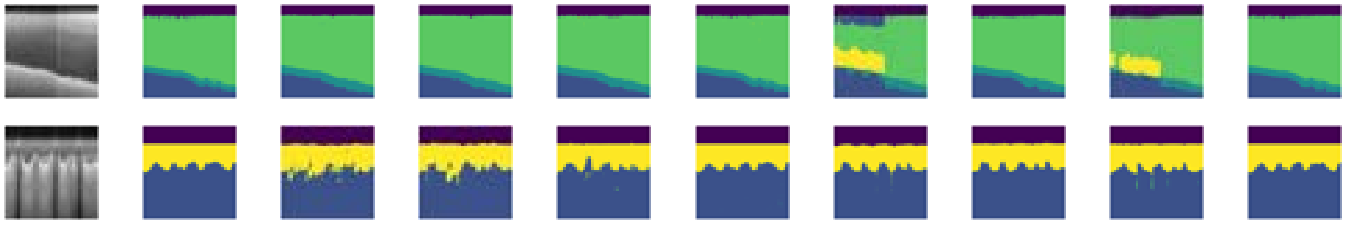
(e)

(f)

Fig. 9. Segmentation map for MCoRDS-1 for Experiment I using all D_{Tr} for fine-tuning: (a) input radargram, (b) reference map, (c) Case I with Configuration I, (d) Case I with Configuration II, (e) Case II with Configuration I, (f) Case II with Configuration II. The orange area represents the grounding area.

TABLE IV
CASE I AND II EXPERIMENTAL SETUPS.

Parameter	Pre-Training Case II	Fine-Tuning Case I	Fine-Tuning Case II
Network trainable weights	62075526	6505219	31036741
Convolutional kernel size in the range and azimuth (k_y, k_x)	(3,3)	(3,3)	(3,3)
Training epochs (ϵ)	20	200	200
Optimizer	Adam	Adam	Adam
Learning rate (l_r)	0.001	0.0001	0.0001
Batch size (N_B)	16 (MCoRDS-1) 4 (MCoRDS-3)	16 (MCoRDS-1) 4 (MCoRDS-3)	16 (MCoRDS-1) 4 (MCoRDS-3)
Loss (L)	L_r and $L_{a/s}$	Weighted sparse categorical cross-entropy loss ($L_{fine-tuning}$)	



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 10. Segmentation maps of MCoRDS-1 for an inland patch (upper line) and a coastal area patch (lower line): (a) input radargram; (b) reference map; for Case I CNN, Experiment I with Configuration II using (c) 67% and (d) 100% of D_{Tr} for fine-tuning, (e) Experiment II with Configuration II with 100% of D_{Tr}, (f) Experiment I with Configuration I with 100% of D_{Tr}; for Case II CNN, Configuration II and Experiment I, using (g) 67% and (h) 100% of D_{Tr}, (i) Configuration II and Experiment II, with 100% of D_{Tr}, (j) Configuration I and Experiment I with 100% of D_{Tr}.

TABLE V
EXPERIMENT I PERFORMANCE WITH CASE I AND II IN WITH CONFIGURATION I-II AND FINE-TUNED WITH 100% OF D_{Tr} COMPARED WITH BASELINE METHODS (MCORDS-1 DATASET).

Methodology	OA	Metric	ν_{fs}	ν_n	ν_b	ν_l	ν_{fl}	Average
Case I Configuration I	98.84%	Sensitivity	0.9904	0.9945	0.9913	0.9694	0.9809	0.9853
		Specificity	0.9995	0.9872	0.9980	0.9970	0.9986	0.9961
Case I Configuration II	97.25%	Sensitivity	0.9920	0.9755	0.9837	0.9887	0.8978	0.9675
		Specificity	0.9961	0.9902	0.9953	0.9952	0.9873	0.9928
Case II Configuration I	**99.58%**	Sensitivity	0.9981	0.9941	0.9989	0.9978	0.9978	**0.9974**
		Specificity	0.9998	0.9998	0.9975	0.9997	0.9986	**0.9991**
Case II Configuration II	97.25%	Sensitivity	0.9940	0.9770	0.9981	0.9880	0.8858	0.9686
		Specificity	0.9957	0.9935	0.9960	0.9930	0.9876	0.9931
CNN proposed in [21]	98.37%	Sensitivity	-	0.9957	0.9817	0.9881	-	0.9882
		Specificity	-	0.9952	0.9831	0.9899	-	0.9871
SVM and handcrafted features [7]	99.09%	Sensitivity	-	0.9947	0.9752	0.9952	-	0.9883
		Specificity	-	0.9902	0.9978	0.9989	-	0.9956

Fig. 9). The network predicts an extremely accurate segmentation map with minimal errors in all experiments. However, the floating and inland ice are better differentiated from the CNN in Case II (Fig. 9(e) and (f)). The segmentation maps with the Case I CNN (Fig. 9(c) and (d)) show segmentation errors in the grounding area. Moreover, in both Cases, Configuration I performance is comparable with those of Configuration II, even if there is little correlation between the test and the training dataset (the training set patches are taken from campaigns different than the test dataset). These results are confirmed by Fig. 10, where we show segmentation maps of Experiments I and II obtained varying the number of fine-tuning samples and Configuration. Experiments with Case I CNN (Fig. 9(c)-(f)) are less affected by the number of fine-tuning samples than those with the Case II CNN (Fig. 9(g)-(j)). This is expected, as the number of parameters to train in the Case I network is dramatically lower. For Case II CNN, experiments with the higher number of fine-tuning samples (Fig. 9(h) and (j)) show better qualitative results for both the inland and the floating-ice patch.

Tables VI, VII, and V show the quantitative results of different experiments with MCoRDS-1 dataset. Table V shows the performance of the baseline methods [8], [21] and the proposed methods for Experiment I with Configuration I and II for Case I and II. The proposed methods can accurately discriminate all the targets with high sensitivity (on average

TABLE VI
EXPERIMENTS I-III OA FOR CASE I AND II CNNS, VARYING THE FINE-TUNING DATASET SIZE (MCORDS-1 DATASET).

	% of D_{Tr}	# Patches	Configuration I Experiment I	Configuration II Experiment I	Configuration II Experiment II	Configuration II Experiment III
Case I	8 %	96	**91.87 %**	75.04 %	86.43 %	55.69 %
	17 %	191	**95.75 %**	95.74 %	95.61 %	63.52 %
	67 %	763	**98.13 %**	96.26 %	96.16 %	66.13 %
	83 %	954	**98.56 %**	96.52 %	96.43 %	66.27 %
	100 %	1145	98.84 %	97.25 %	97.12 %	66.33 %
Case II	8 %	96	91.77 %	80.11 %	74.25 %	71.24 %
	17 %	191	95.58 %	95.32 %	94.27 %	88.18 %
	67 %	763	97.66 %	96.73 %	94.70 %	93.20 %
	83 %	954	98.24 %	96.99 %	95.88 %	94.72 %
	100 %	1145	**99.58 %**	97.25 %	96.75 %	96.01 %

TABLE VII
EXPERIMENT IV OA CASE II CNN (MCORDS-1 DATASET).

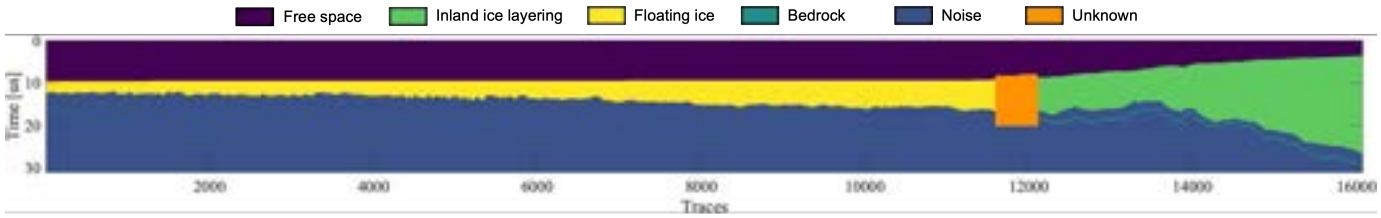
% of $D_{T_{\text{unl}}}$	Patches	Configuration I	Configuration II
0 %	0	98.62 %	96.01 %
33 %	1314	98.80 %	96.53 %
66 %	2629	98.91 %	96.71 %
100 %	3983	99.58 %	97.25 %

over 96%) and specificity (on average over 99%). Despite the different prior probabilities of the classes, the less frequent classes, such as the bedrock and the floating ice and crevasses, are well segmented with performance comparable to that of the other classes. The proposed and the baseline methods have extremely high overall accuracy (above 97%). However, the proposed method using the Case II deep CNN with Configuration I outperforms all the others with an OA of 99.58%. Confirming the qualitative results, Table V also shows that the performance of the proposed method using the CNNs in Case I and II with Configuration II are comparable with those with Configuration I (about 1.5% difference). As expected, the Case II CNN performs better than Case I lightweight CNN in both Configurations. This is because the Case II CNN is deeper and can learn semantically more meaningful features. However, a deeper CNN requires more data to be trained in a robust way to avoid overfitting. This is visible in Table VI, where the performance of Case II CNN is strongly correlated with the number of fine-tuning labeled samples used for training. With smaller fine-tuning datasets, the lightweight CNN performs slightly better than the deeper CNN. When we use the largest fine-tuning dataset (100% of D_{Tr}), Case II CNN obtains better results than Case I CNN in Configuration I with an overall accuracy of 98.84% and in Configuration II with an overall accuracy of 97.25%, respectively. When we dramatically reduce the fine-tuning datasets to about 16% of D_{Tr}, the segmentation accuracy slightly decreased but remained above 95% for both Configurations. When the fine-tuning dataset is further decreased to about 8% of D_{Tr}, we identified a cliff of more than 8% in the performance of Configuration I compared to Configuration II. This is expected since the data used for training in Configuration I are from the same campaign used for the inference. Thus, the network is expected to know extremely well the radiometric and geometric characteristics of the data in test campaigns. In Configuration II, the network has a more complex task of generalizing and translating semantically meaningful features extracted from a training campaign other than the campaigns in the test phase.

Looking at the results of Experiment II (no data augmentation) in Table VI, we note that the data augmentation helps in improving the overall accuracy of about 1% for all the experiments but that fine-tuning with only about 8% of the D_{Tr} as the pre-training and the fine-tuning dataset size have a higher impact. The impact of using data augmentation became noticeable when the fine-tuning dataset is reduced to 8% of D_{Tr} as the difference between the performance with and without data augmentation is about 10% with Case II CNN and 15% with Case II CNN, respectively. Finally, looking at Experiment III (no transfer learning–random weight initialization), Case II CNN outperforms Case I CNN with an overall accuracy higher of about 20% on average. This is because the weights of the network core are not updated during fine-tuning. However, Case II CNN performance in Experiment III is lower by about 2% compared to Experiment I. This highlights the importance of transfer learning as it makes Case II adapted network adaptation in Sec III-B2 reach state-of-the-art results. The importance of transfer learning is confirmed in Table VII, which shows the accuracy using Case II CNN in Configuration I and II by varying the size of the pre-training dataset ($D_{T_{\text{unl}}}$). In Configuration I, the performance decreases with the number of patches in the pre-training by about 1% only: when the weights are randomly initialized, the accuracy is about 98.62%, while using all the data in the pre-training, the accuracy raises to 99.58%. This is also true in Configuration II: the accuracy is 96.01% without transfer learning and 97.25% when transfer learning is used with all $D_{T_{\text{unl}}}$.

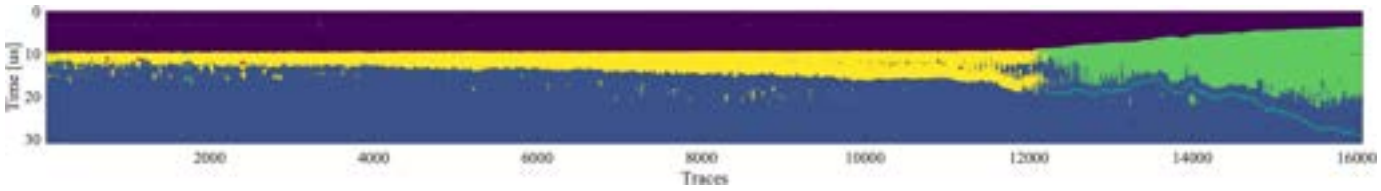
2) MCoRDS-3 dataset: The proposed method qualitatively segments the radargrams with good overall accuracy and the more complex MCoRDS-3 dataset (see Fig. 11). Despite the accurate segmentation map, the floating and inland ice are better differentiated by the CNN in Case II (Fig. 11(e) and (f)). However, similarly to the MCoRDS-1 dataset, the segmentation maps with the Case I CNN (Fig. 11(c) and (d)) show significant segmentation errors in the grounding

(a)

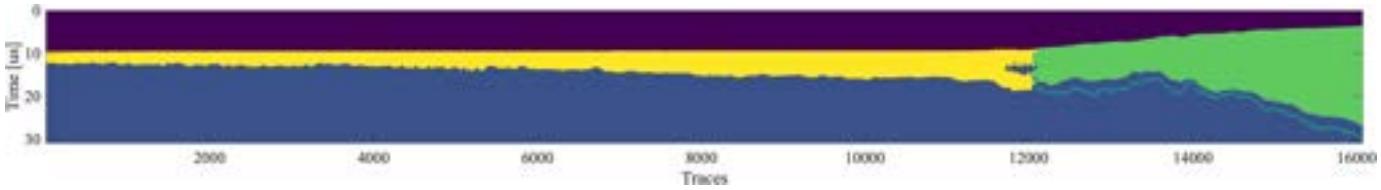


(b)

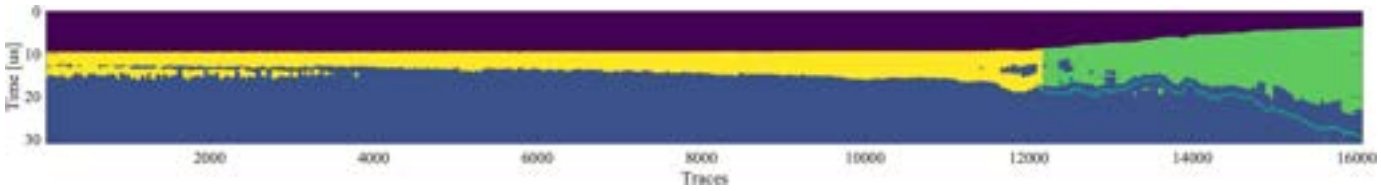
(c)



(d)

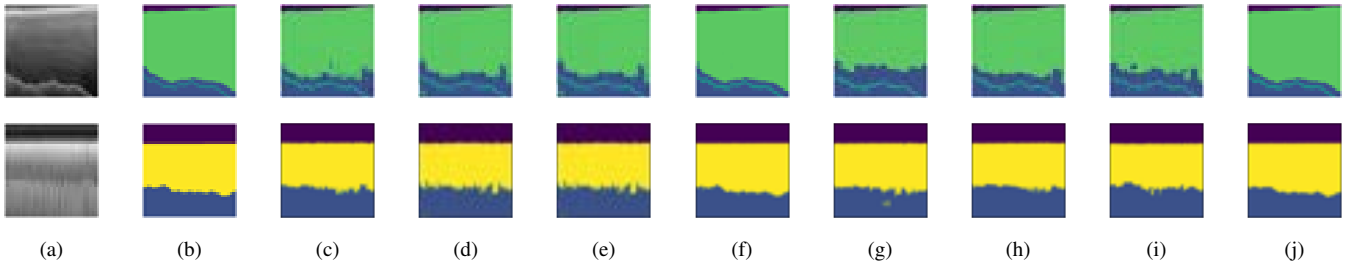


(e)



(f)

Fig. 11. Segmentation map of the MCoRDS-3 for Experiment I using all D_{Tr} for fine-tuning: (a) input radargram, (b) reference map, (c) Case I with Configuration I, (d) Case I with Configuration II, (e) Case II with Configuration I, (f) Case II with Configuration II.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 12. Segmentation maps of MCoRDS-3 for an inland patch (upper line) and a coastal area patch (lower line): (a) input radargram; (b) reference map; for Case I CNN, Experiment I with Configuration II using (c) 67% and (d) 100% of D_{Tr} for fine-tuning, (e) Experiment II with Configuration II with 100% of D_{Tr}, (f) Experiment I with Configuration I with 100% of D_{Tr}; for Case II CNN, Configuration II and Experiment I, using (g) 67% and (h) 100% of D_{Tr}, (i) Configuration II and Experiment II, with 100% of D_{Tr}, (j) Configuration I and Experiment I with 100% of D_{Tr}.

TABLE VIII
EXPERIMENTS I-III PERFORMANCE WITH CASE I AND II WITH CONFIGURATION I-II FINE-TUNED WITH 100% OF D_{Tr} COMPARED WITH BASELINE METHODS (MCORDS-3 DATASET).

Methodology	**OA**	**Metric**	ν_{fs}	ν_n	ν_b	ν_l	ν_{fl}	**Average**
Case I Configuration I	99.26%	Sensitivity	0.9986	0.9885	0.9995	0.9947	0.9989	0.9960
		Specificity	0.9998	0.9980	0.9961	0.9982	0.9991	0.9982
Case I Configuration II	96.43%	Sensitivity	0.9947	0.9773	0.9404	0.8412	0.9448	0.9397
		Specificity	0.9980	0.9573	0.9969	0.9970	0.9916	0.9882
Case II Configuration I	**99.33%**	Sensitivity	0.9974	0.9888	0.9990	0.9990	0.9974	**0.9963**
		Specificity	0.9999	0.9994	0.9963	0.9984	0.9985	**0.9985**
Case II Configuration II	96.55%	Sensitivity	0.9975	0.9609	0.9965	0.8881	0.9783	0.9642
		Specificity	0.9999	0.9741	0.9972	0.9971	0.9817	0.9900
CNN proposed in [21]	98.37%	Sensitivity	-	0.9957	0.9817	0.9881	-	0.9882
		Specificity	-	0.9952	0.9831	0.9899	-	0.9871
SVM and handcrafted features [7]	97.9%	Sensitivity	-	0.9834	0.9530	0.9833	-	0.9733
		Specificity	-	0.9775	0.9970	0.9911	-	0.9885

TABLE IX
EXPERIMENTS I-III OA FOR CASE I AND II CNNS, VARYING THE FINE-TUNING DATASET SIZE (MCORDS-3 DATASET).

	% of D_{Tr}	**# Patches**	**Configuration I**	**Configuration II**		
			Experiment I	**Experiment I**	**Experiment II**	**Experiment III**
Case I	8 %	79	**93.37 %**	85.30 %	84.45 %	51.02 %
	17 %	157	**97.31 %**	93.18 %	93.17 %	53.76 %
	67 %	630	**99,14 %**	95.67 %	95.66 %	56.43 %
	83 %	787	**99,18 %**	96.01 %	*95.98 %*	59.12 %
	100 %	944	99,26 %	96.43 %	96.40 %	60.88 %
Case II	8 %	79	91.44 %	84.47 %	84.37 %	79.38 %
	17 %	157	97.09 %	92.92 %	92.78 %	88.91 %
	67 %	630	99.12 %	95.72 %	95.51 %	92.31 %
	83 %	787	99.17 %	95.91 %	95.78 %	93.72 %
	100 %	944	**99.32 %**	96.55 %	96.50 %	94.32 %

TABLE X
EXPERIMENT IV OA CASE II CNN (MCORDS-3 DATASET).

% of $D_{T_{\text{unl}}}$	**# Patches**	**Configuration I**	**Configuration II**
0 %	0	98.22 %	94.32 %
33 %	1034	98.48 %	95.75 %
66 %	2069	98.67 %	96.29 %
100 %	3448	99.32 %	96.55 %

area. Moreover, in both Cases, Configuration I performance is comparable with Configuration II. In the segmentation maps with Configuration II, both Case I and II CNNs predict the vertical reflections from the crevasses of the ice shelf as part of the class floating ice a few times (yellow class, left side). Moreover, the CNNs are less accurate in predicting the deepest ice layering (green class, right side) with Configuration II. This is also visible in Fig. 12, where we show segmentation maps of two patches (Fig. 11(a)) varying the number of fine-tuning samples. Experiments with lightweight Case I CNN (Fig. 11(c)-(f)) are less affected by the small number of fine-tuning samples than those with the deeper Case II CNN (Fig. 11(g)-(j)). However, for both cases, experiments with the higher number of fine-tuning samples (Fig. 11(e) and (f)

for Case I) and (Fig. 11(h) and (j) for Case II) show better qualitative results for both inland and floating-ice patches.

Moving to the quantitative results, Table VIII shows that the proposed and the baseline methods have all excellent performance and high overall accuracy above 96 %. However, similarly to MCoRDS-1 results, the deep Case II CNN with Configuration I outperforms all the others. Similarly to the MCoRDS-1 dataset, all the classes are segmented with high sensitivity and specificity on average higher than 93% and 98%, respectively. Despite the strongly unbalanced priors also for MCoRDS-3 dataset, the less frequent classes (bedrock and floating ice) are discriminated with a sensitivity higher than 94% and a specificity higher than 98%.

With the largest fine-tuning dataset (100% of D_{Tr}), Case II CNN outperforms Case I CNN in Configuration I with an overall accuracy of 99.33% and in Configuration II with an overall accuracy of 96.55%, respectively. In all the experiments, the segmentation performance with Configuration II is comparable with those with Configuration I. The performance difference between Configuration I and II is more significant than the MCoRDS-1 dataset. For the MCoRDS-1 dataset, the performance difference is lower than 1% (see Table VI), while for the MCoRDS-3 dataset, the difference varies from 5% to 3% (see Table IX). As expected, dramatically decreasing the fine-tuning datasets to about 17% of D_{Tr}, the overall accuracy decreased but remained above 92% for both Configurations. Looking at the results of Experiment II (removing the data augmentation) in Table IX, the impact of data augmentation is negligible for all the sizes of the fine-tuning dataset. This contrasts the MCoRDS-1 results (data augmentation increased the performance by about 5%) because of greater dataset complexity. Finally, in Experiment III (no transfer learning), Case II outperforms Case I with an overall accuracy of average higher of about 30%. However, Case II performance in Experiment III is lower by about 5% than in Experiment I, highlighting the importance of transfer learning.

This is also confirmed by Table X, which shows the accuracy using Case II CNN in Configuration I and II by varying the size of the pre-training dataset ($D_{T_{\text{unl}}}$). In Configuration I, the performance decreases with the decreasing of the number of patches used for pre-training by only about 1%. When the weights are randomly initialized, the accuracy is about 98.22%, while when using all the data in the pre-training, the accuracy rises to 99.32%. This is also true in Configuration II, where the accuracy is 94.32% without pre-training and 96.55% when pre-training with all $D_{T_{\text{unl}}}$. Here, the unadapted CNN, DeepLabV3+, was tested in the most critical and realistic scenario, Configuration II, obtaining inferior performance in terms of segmentation accuracy (i.e., below 60%) and qualitative results, showing no continuity between the segmented patches. These results emphasize that the unadapted CNN suffered severe overfitting.

3) Computational load analysis: Due to the significant number of trainable weights of the CNNs, especially the W-Net and the adapted CNN of Case II, a powerful GPU is required to execute these experiments. Here we employed an Nvidia Tesla T4 GPU and an AMD EPYC 7V12 CPU with 64 GB of memory, specifically designed to run large and complex computational loads. The GPU performs differently in fine-tuning depending on the dataset (i.e., MCoRDS-1 or MCoRDS-3) and the CNN (i.e., MobileNet and W-Net). With the MCoRDS-1 dataset, the machine requires less computational time than the MCoRDS-3 dataset because of the lower spatial resolution of MCoRDS-1. For one MCoRDS-1 patch, Case I CNN takes an average of 11ms, and Case II CNN 31 ms. For one MCoRDS-3 patch, Case I CNN takes an average of 75ms, and Case II CNN 119 ms. Finally, since a pre-trained CNN can be easily retrieved, Case I does not need to be manually pre-trained. However, the Case II W-Net has to be manually pre-trained with $D_{T_{\text{unl}}}$. This also requires an extra computational cost of about around 20 hours.

VI. Conclusions and Future Works

We successfully proposed a novel framework for transfer learning for weakly supervised RS data segmentation. Within the framework, we propose two transfer learning approaches to address the lack of labeled data in the RS domain. The experimental results show that the proposed approaches can accurately identify the target classes and distinguish between classes in the inland and coastal area (novel in the literature) despite the strongly unbalanced priors.

The results proved the effectiveness of the transfer learning framework, which allowed the robust pre-train of a lightweight and a deep CNN to perform radargram segmentation accurately. The lightweight CNN strongly increases the performance from about 60% (random initialization) to more than 96% when pre-trained with ImageNet. When pre-training with radar data, the deep CNN accuracy improves from 94% to more than 96%. However, Case II is more computationally expensive in terms of training time and hardware resources. Moreover, transfer learning effectiveness is evident when fine-tuning with a reduced amount of labeled samples (a hundred). The deeper CNN performs better than the lightweight CNN when fine-tuned with a larger dataset at a higher computational time cost. However, as the size of fine-tuning dataset decreases, the lightweight CNN performs better than the deeper CNN. Therefore, the lightweight CNN pre-trained on ImageNet is extremely convenient when the computational resources are constrained or the labeled data availability is limited. On the other hand, the deeper CNN pre-trained on radar data is more suitable for more accurate segmentation maps. Further, data augmentation has a limited impact on large and medium datasets. However, on small and simple datasets, like MCoRDS-1, the performance of training with data augmentation improves by about 5%. For a more complex dataset, like MCoRDS-3, the accuracy when training with a minimal dataset with and without data augmentation is similar (about 80%). Remarkably, the performance using the same campaigns for training and testing are comparable (a few % lower) to using different campaigns for training and testing. This indicates that radargrams can be accurately segmented with the existing fine-tuned networks without requiring further training. Additionally, the inferior results obtained by the unadapted CNN and without transfer learning prove the proposed techniques' effectiveness in analyzing RS data with DL approaches.

In future works, we plan to test the proposed framework and methodologies to detect more subsurface targets, such as basal ice, and study how to adapt the networks to segment radargrams independently of their acquisition location (e.g., Antarctica, Greenland) or sensor (e.g., MCoRDS-1, MCoRDS-3). Moreover, we aim to use the proposed approaches to analyze planetary radar sounder data of icy and arid areas.

ACKNOWLEDGEMENT

We acknowledge the use of data and data products from CReSIS generated with support from the University of Kansas, NASA Operation IceBridge grant NNX16AH54G, NSF grants ACI-1443054, OPP-1739003, and IIS-1838230, Lilly Endowment Incorporated, and Indiana METACyt Initiative.

REFERENCES

[1] B. Smith, H. A. Fricker, A. S. Gardner, B. Medley, J. Nilsson, F. S. Paolo, N. Holschuh, S. Adusumilli, K. Brunt, B. Csatho *et al.*, "Pervasive ice sheet mass loss reflects competing ocean and atmosphere processes," *Science*, vol. 368, no. 6496, pp. 1239–1242, 2020.

[2] H. Pritchard, S. R. Ligtenberg, H. A. Fricker, D. G. Vaughan, M. R. van den Broeke, and L. Padman, "Antarctic ice-sheet loss driven by basal melting of ice shelves," *Nature*, vol. 484, no. 7395, pp. 502–505, 2012.

[3] L. Bruzzone, J. J. Plaut, G. Alberti, D. D. Blankenship, F. Bovolo, B. A. Campbell, D. Castelletti, Y. Gim, A.-M. Ilisei, W. Kofman *et al.*, "Jupiter icy moon explorer (juice): Advances in the design of the radar for icy moons (rime)," in *2015 IEEE international geoscience and remote sensing symposium (IGARSS)*. IEEE, 2015, pp. 1257–1260.

[4] L. Bruzzone, F. Bovolo, L. Carrer, E. Donini, and S. Thakur, "Stratus: A new mission concept for monitoring the subsurface of polar and arid regions," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, 2021, pp. 661–664.

[5] L. Bruzzone, F. Bovolo, S. Thakur, L. Carrer, E. Donini, C. Gerekos, S. Paterna, M. Santoni, and E. Sbalchiero, "Envision mission to venus: Subsurface radar sounding," in *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, pp. 5960–5963.

[6] E. Donini, S. Thakur, F. Bovolo, and L. Bruzzone, "An automatic approach to map refreezing ice in radar sounder data," in *Image and Signal Processing for Remote Sensing XXV*, vol. 11155. International Society for Optics and Photonics, 2019, p. 111551B.

[7] A.-M. Ilisei, M. Khodadadzadeh, A. Ferro, and L. Bruzzone, "An automatic method for subglacial lake detection in ice sheet radar sounder data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 6, pp. 3252–3270, 2018.

[8] A. M. Ilisei and L. Bruzzone, "A system for the automatic classification of ice sheet subsurface targets in radar sounder data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 6, pp. 3260–3277, 2015.

[9] D. M. Schroeder, "Pathways to multitemporal radar sounding in terrestrial glaciology," in *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2020, pp. 3731–3734.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[11] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 201–208.

[12] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," *arXiv preprint arXiv:1711.08506*, 2017.

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

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[15] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[16] M. Xie, N. Jean, M. Burke, D. Lobell, and S. Ermon, "Transfer learning from deep features for remote sensing and poverty mapping," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[17] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National science review*, vol. 5, no. 1, pp. 44–53, 2018.

[18] L. Bruzzone and M. Marconcini, "Domain adaptation problems: A dasvm classification technique and a circular validation strategy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 5, pp. 770–787, 2009.

[19] X. Yao, J. Han, G. Cheng, X. Qian, and L. Guo, "Semantic annotation of high-resolution satellite images via weakly supervised learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 6, pp. 3660–3671, 2016.

[20] A. Ferro and L. Bruzzone, "Analysis of radar sounder signals for the automatic detection and characterization of subsurface features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4333–4348, 2012.

[21] E. Donini, F. Bovolo, and L. Bruzzone, "A deep learning architecture for semantic segmentation of radar sounder data," *IEEE Transactions on Geoscience and Remote Sensing*, 2021.

[22] M. H. García, E. Donini, and F. Bovolo, "Automatic segmentation of ice shelves with deep learning," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, 2021, pp. 4833–4836.

[23] M. H. Garcia, E. Donini, and F. Bovolo, "Transfer learning for the semantic segmentation of cryoshpere radargrams," in *Image and Signal Processing for Remote Sensing XXVII*, vol. 11862. SPIE, 2021, pp. 223–233.

[24] V. V. Bogorodsky, V. V. Bogorodskiĭ, C. R. Bentley, and P. Gudmandsen, *Radioglaciology*. Springer Science & Business Media, 1985, vol. 1.

[25] M. G. Cavitte, D. A. Young, R. Mulvaney, C. Ritz, J. S. Greenbaum, G. Ng, S. D. Kempf, E. Quartini, G. R. Muldoon, J. Paden *et al.*, "A detailed radiostratigraphic data set for the central east antarctic plateau spanning from the holocene to the mid-pleistocene," *Earth System Science Data*, vol. 13, no. 10, pp. 4759–4777, 2021.

[26] G. d. Q. Robin and D. Millar, "Flow of ice sheets in the vicinity of subglacial peaks," *Annals of Glaciology*, vol. 3, pp. 290–294, 1982.

[27] L. Lindzey, "A brief introduction to ice-penetrating radar [online]," *lindzey.github.io*, 2015.

[28] C. Oliver and S. Quegan, *Understanding synthetic aperture radar images*. SciTech Publishing, 2004.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" *arXiv preprint arXiv:1411.1792*, 2014.

[30] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, pp. 901–909, 2016.

[31] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," *arXiv preprint arXiv:1702.03275*, 2017.

[32] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint arXiv:1607.08022*, 2016.

[33] F. Rodriguez-Morales, S. Gogineni, C. J. Leuschen, J. D. Paden, J. Li, C. C. Lewis, B. Panzer, D. G.-G. Alvestegui, A. Patel, K. Byers *et al.*, "Advanced multifrequency radar instrumentation for polar research," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 5, pp. 2824–2842, 2013.

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[35] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[36] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.

[37] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.