How to obtain high resolution snow height maps with geospatial data? A deep learning approach

Alejandro Betato Sancerni
Artificial Intelligence Research Master's Degree, UIMP-AEPIA
Zaragoza, Spain
abetatos@gmail.com

***Abstract*—Accurate prediction of snow depth is crucial for various applications, including hydrological risk assessment, flood forecasting, water resource management, and weather forecasting. Prior research has seen success in utilizing deep learning techniques to generate snow height maps; however, many studies have been constrained by limited geographical coverage or the use of coarse-resolution data. Our approach aims to overcome these limitations by leveraging high-resolution LiDAR maps, satellite imagery, digital elevation models, and three novel time-dependent variables. Moreover, we introduce an adaptation of the original U-Net architecture to function as a pixel-wise regressor. We demonstrate its ability to provide fine-grained and accurate snow depth predictions over large geographical areas. The proposed method is able to model snow depth in the high mountainous region of Davos with average errors of 0.52m and resolutions of 5m.**

***Index Terms*—computer vision, pixel-wise regression, snow depth maps**

I. Introduction

Snow is a crucial component of the Earth's hydrological cycle, a fundamental process that governs the movement and distribution of water on our planet. Precise measurements of snow depth stand as a vital necessity, serving as a cornerstone in the prediction and effective management of water resources. This need for accuracy becomes even more pronounced in regions characterized by rugged mountainous landscapes, where snow plays a pivotal role in shaping seasonal water reservoirs [1].

Beyond its hydrological significance, snow exerts a notable influence on the Earth's energy balance. This is attributed to its high *albedo*, a term that refers to the capacity to reflect a substantial portion of the incoming solar radiation. In this context, the presence of snow cover amplifies the intricate interplay of radiative energy, impacting the amount of energy absorbed by the Earth's surface and the amount reflected back into the atmosphere [2].

In collaboration with Wegaw, a geospatial start-up, this research project aims to harness the potential of reflected light with solar panels. While there's growing interest in utilizing this reflected light for energy generation, there's currently a lack of research regarding the optimal placement of solar panels in such scenarios. The Wegaw group has undertaken a mission to enhance the accuracy of snow depth modeling at high spatial resolution.

The intricate topography of tall mountain ranges affects the spatial variability of snow cover across different scales [3]. Moreover, it also makes many places on Earth inaccessible to traditional methods of measuring snow depth, such as manual snow surveys or remote sensing with optical sensors.

In recent years, satellite imagery has become more accessible to the general public with missions such as Copernicus[1], from the European Space Agency (ESA); or MODIS[2], from the National Aeronautics and Space Administration (NASA). Both missions collect public data on an hourly basis. The high availability in both spatial and temporal resolution of satellite imagery has induced many works that attempt to overcome the snow prediction problem. Although there is a consistent problem in the literature, annotated information for snow depth is scarce and usually found in low spatial resolution. Liu et al. [4] used brightness temperatures measured by passive microwave radiometers to retrieve snow depth over the Arctic Ocean with a spatial resolution of 25km. Moosavi et al. [5] used Moderate-resolution Imaging Spectroradiometer (MODIS) [6] bands with a resolution varying from 250m to 1km to obtain a fractional snow cover in the Central Alborz Mountains.

Airborne Light Detection and Ranging (LiDAR) is nowadays one of the most reliable means of terrain mapping, with spatial resolutions of less than one meter and errors typically under the decimeter [7]–[9]. Due to such precise mapping, LiDAR imagery has become a great tool for modeling at higher resolution. Wulf et al. [10] validated their snow-depth prediction approach with airborne snow depth surveys from the European Alps and the Rocky Mountains. Cartwright et al. [11] used LiDAR imagery to train a snow-depth prediction model in Southwest Alberta. Most of those datasets are not publicly available, limiting both peer review and general public accessibility. Although there are some public datasets, they are generally not sufficiently big to train machine learning (ML) or deep learning (DL) models, which could be a promising way to address the problem.

Classical machine learning approaches include regression tree [12], random forest regression [13], or support vector regression [13]. On the other hand, deep learning approaches include the use of Deep Belief Networks (DBN) [14] or recurrent networks such as the Long-Short Term Memory models (LSTM) [15]. Convolutional Networks (CNN) are also

[1] https://www.copernicus.eu
[2] https://modis.gsfc.nasa.gov

commonly used in this context, such as in [16], whose authors proposed a Deep Residual Network for snow depth prediction at resolutions of $\approx 25km$. To the best of our knowledge, very few research works have tackled the problem at high spatial resolution. Daudt et al. [17] achieved a root-mean-square error (RMSE) between 0.44-0.92m at 10m resolution with the use of a CNN-based on ConvGRU cells [18], which introduced a recurrence allowing the model to take samples from previous days to predict the next one.

The snow depth prediction survey carried out by Eberhard et al. [19] suggests that snow depth maps generated from very high resolution ($\approx 10m$) satellite imagery reach accuracies up to 0.5–1.0 m in terms of RMSE.

The input features are often images derived from satellite imagery and the geomorphology of the area, but there is no consensus on the best predictors. Some of them are commonly used such as the elevation or the surface slope, for this matter we used the insightful knowledge of the Wegaw team which supported us with the selection of input variables. They are discussed in Section II. Furthermore, we encountered a difficulty related to temporal changes. When working with snow depth mapping, the majority of input features remain constant (as they are geomorphological attributes), therefore, we must introduce a feature that depends on the conditions of a given day. In Section II we introduce three different variables based on snow depth measurements collected at meteorological stations.

Along with the suggested features, we propose a novel approach based on the UNet [20] architecture. It was selected as our base architecture due to its effectiveness in segmentation problems, its ability to handle both local and global contextual information, its good performance even with limited training data, and its generative nature. In this context, we have adapted the original architecture to serve as a pixel-wise regressor. Although not novel, there is limited research on its application in regression problems. We decided to investigate its potential in this specific domain.

The main contributions of this work are:

- We present three methods for incorporating temporal variability into the analysis using meteorological station data. They are proven to improve predictions and serve as scaling factors, indicating that the higher the variables the higher the predictions.
- We propose a new method of creating snow depth maps based on an adaptation of the original UNet architecture. Moreover, we are also able to demonstrate its capabilities when utilized as a pixel-wise regressor.
- We present an informative study that may serve as a foundation for future research.
- We introduce an image generation technique that consists in cropping irregular maps into many squared-shape images in an optimized way, making every zone equally present in the final dataset. Moreover, this approach also serves as a data augmentation technique. It is publicly available as a Python library called MAPchete[3].

The rest of the manuscript is organized as follows. Section II provides a review of the selected input features. Section III describes how to process raster files to adapt them to our model's input layers. Section IV covers the UNet architecture and our proposal. Section V reviews the implementation details. Section VI defines the metrics employed in this study. Section VII details the experimentation carried out, including the experiments related to the architecture parametrization, feature selection, and predictions and errors analysis. Section VIII covers the key findings of our work and proposes future lines of research.

II. Dataset

In this section, we introduce the features proposed in our research. These features are essential components of our predictive model and play a crucial role in understanding the factors influencing snow accumulation and distribution. However, it is important to note that the availability of data for certain features, such as wind in high mountainous areas, is limited. This data is typically collected in meteorological stations, which are often located on plains and easily accessible terrain. As a result, there is a lack of data for regions with complex mountainous terrain. To address this limitation, it is necessary to introduce variables that cover the entire range of mountainous areas, including remote and isolated regions.

Within the domain of geospatial analysis, the utilization of raster data is widespread. In this particular section, we introduce all the predictor variables in this format, as it proves to be the most suitable for both geodata and deep learning models. In this representation, each grid cell is assigned a value that encapsulates the distinctive features of a specific geographical area.

Firstly, we introduce the concept of slicing, as it is mentioned in the next sections.

A. Slicing the data

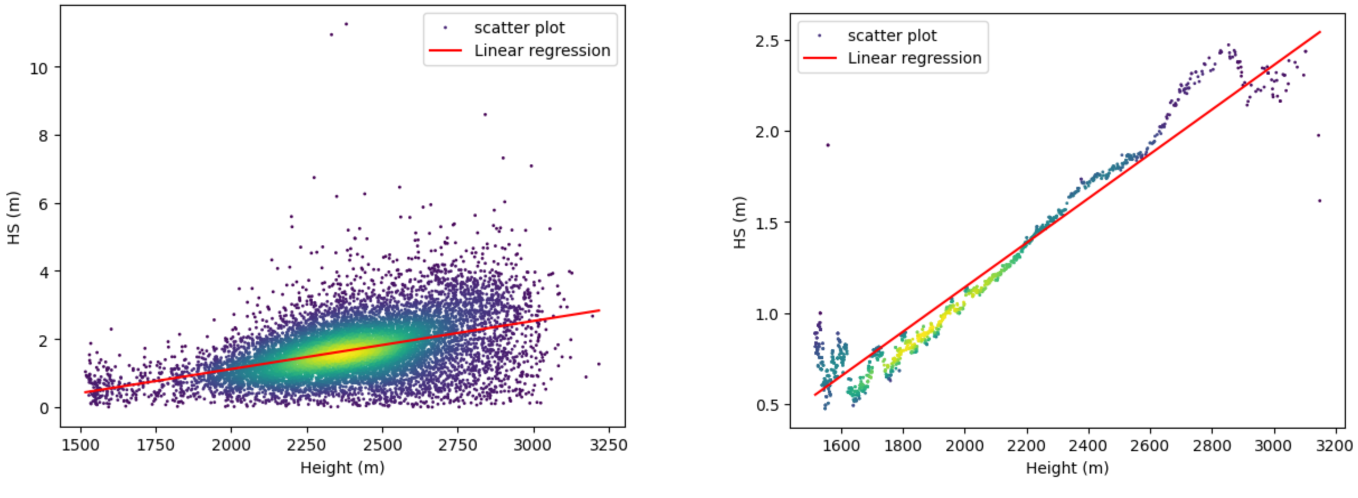
Given an input feature, slicing can be defined as averaging the snow depth in intervals. By averaging the snow depth within these intervals, slicing provides a macro-scale perspective on how snow depth varies with respect to the feature. Fig. 1a plots the snow depth versus elevation in the region of Davos. Fig. 1b plots the results of slicing, averaging the snow depth each meter. As can be observed, the correlation between snow depth and a sample variable becomes more evident after the slicing approach.

There is also another benefit of the slicing approach. For both graphs, we fitted a linear regression obtaining the following formulas that correspond to Fig. 1a and 1b, respectively:

$$HS = 146 \times 10^{-5} h - 1.830 \tag{1}$$

$$HS = 124 \times 10^{-5} h - 1.353 \tag{2}$$

[3] https://github.com/abetatos/mapchete



(a)

(b)

Fig. 1: Snow depth vs elevation on the region of Davos in 2013: (a) before and (b) after slicing. A probability density function (based on Scott's Rule [21]) has been applied for color grading to give a sense of point density.

with h the elevation and HS the snow depth.

The abundance of data points influences the linear regression to be more inclined towards those regions. To overcome this bias and better study the correlations between variables, slicing is employed as a normalization technique. It allows for a more controlled analysis and facilitates a clearer understanding of the relationships between the variables.

B. Snow Depth

In recent years, Unmanned Aerial Systems (UAS) have proven highly effective for geodata measurements, utilizing LiDAR technologies to accurately map surfaces, with errors typically below the decimeter level as demonstrated by Buhler et al. [9]. To derive snow depth information, UAS scan entire mountain ranges mapping the height across the area. By subtracting this height measurement from the known elevation of the bare mountains, the snow depth across the landscape can be precisely mapped.

Although it is an optimal technology for the purpose of snow mapping, it comes with a great cost and thus publicly available data is scarce. The dataset used in this research includes two sources of information obtained with this technology:

- Nine snow depth maps collected from 2017 to 2020 in the Swiss regions of Davos, Laucherenalp, and Saflischpass. These data were provided by Wegaw.
- Eight snow depth maps collected from 2010 to 2016 in the Swiss region of Davos. These data are publicly available at Envidat[4].

This gives us a total of sixteen maps spanning the years 2010 to 2020. In the following sections, the snow depth maps are denoted as HS.

C. Geospatial imagery

Snow accumulation depends on a great number of physical processes but there is no consensus on the variables that should be used to estimate snow depth (HS). Consequently, we selected a set of variables based on the recent literature and the experience of the Wegaw team. Fig. 2 shows a representation of the ten variables considered in this research, which are described in the following.

1) Digital Elevation Model: A digital elevation model (DEM) is a 3D representation of the Earth's surface topography and determines the elevation at a given location. By incorporating a DEM, we enabling out model to understand the spatial characteristics of the environment.

DEM models for Switzerland are publicly available at the Swiss Federal Office of Topography[5]. They consist of a 3D representation of the Swiss Alps with a spatial resolution of 0.5m and are renewed every six years.

2) Slope: The surface slope is, by definition, the derivative of the height. The usefulness of this attribute lies in the fact that snow tends to accumulate more in plain areas, while it is prone to decay in steeper ones. Moreover, the slope plays a role in the amount of radiative energy the surface receives [22].

3) Surface Aspect: The surface aspect shows the degree between the normal of the surface and a given direction $\vec{D}$, which commonly refers to the North vector. For instance, an aspect value of 0° represents a slope facing north, while an aspect of 90° represents a slope facing east. It has been proven to be an effective variable when it comes to snow depth modeling as same-facing surfaces have similar exposure to external factors, such as hourly sun exposure or wind trends [23]. It has a drawback, the discontinuity between 0 and 360º

[4] https://www.envidat.ch/dataset

[5] https://www.swisstopo.admin.ch/en/geodata/height/alti3d.html

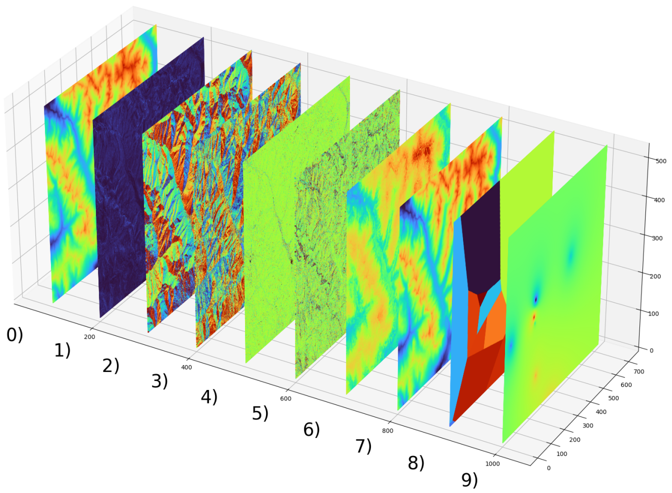


Fig. 2: Representative samples for the test area of our dataset. The input variables are: 0) DEM, 1) Slope, 2) Aspect, 3) FFD, 4) TPI, 5) TPIGWG, 6) SCE, 7) ProbStat, 8) Voronoi, and 9) ProbStat. Values have been normalized to $[0, 1]$. Shift to red means higher values.

could compromise the usefulness of the variable. In Section VII-G we review the impacts of the discontinuity in our setup.

4) Far From $\vec{D}$: When the aspect is compared with the snow depth using the slicing approach. We noticed that the minimums of the features are well-defined and tend to be in the same direction, while maximums are more sparse. Fig. 3 shows this behavior. We decided to test if Aspect is more informative for lower snow depths computing a new input feature that we called Far From $\vec{D}$ or FFD.

It consists of a rotation of the aspect to the minimum snow direction obtaining values in the range $[-180, 180]$. We then applied the module so that we obtained several values in the range $[0, 180]$. The usefulness of this variable relies on avoiding the aforementioned discontinuity and simplifying the variable, thus making training convergence faster and easier for the model. In Section VII-E we review the impacts of this new feature.

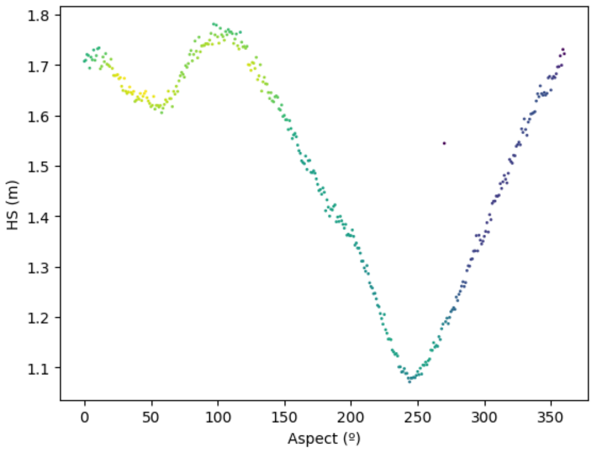


Fig. 3: Sliced snow depth vs slope aspect.

5) Topographic position index: The topographic position index (TPI) is a measure of the elevation of the raster grid cell compared to the elevation of a specified number of neighbor cells. By definition, it determines concave and convex zones, and thus it can predict whether a zone is prone to accumulate snow. It has also been proven to be a good predictor for the purpose of snow depth prediction [17], [24]. The main definition of TPI is the difference between the elevation of a central point and the average elevation of its surrounding points within a defined radius. But the way of computing the difference might vary. Here we focus on two of them:

The first one is provided by the WhiteBoxTools [25] library and it is based on the study of Newman et al. [26]. It is referred to as TPI in the rest of the manuscript. This algorithm adds a regularization in order to obtain values in the range $[-1, 1]$ following:

$$TPI = \begin{cases} \frac{z-\mu}{\mu - z_{\text{min}}} & \text{if } z < \mu \\ \frac{z-\mu}{z_{\text{max}} - \mu} & \text{if } z \geq \mu \end{cases} \tag{3}$$

where z represents the elevation, μ the average elevation, z_{max} the maximum elevation, and z_{min} the minimum elevation.

The second one is provided by Wegaw and it is referred to as TPIWGW. This algorithm does not add a regularization so we filter the maximum and minimum values to ±5 following the advice of the Wegaw team.

6) Snow Cover Extent: The snow cover extent (SCE) is a probabilistic measure that indicates where snow tends to be present. These maps are obtained in two steps:

1) The first part of the process was provided by the Wegaw team and it consists of binary maps that indicate presence of snow in a certain area. It was obtained following the approach by Grizonnet et al. [27], with the use of the following satellite resources: CRYO[6], MODIS[7], and GFS[8]. Wegaw generated a total of 153 maps spanning from 2017 to 2020.
2) Subsequently, we decided to average the images, resulting in a probabilistic map that provides insights into the likelihood of snow coverage in a given area throughout the year.

This particular feature alone has the capability to indicate the distribution of snow, and thus, it could serve as a robust predictor.

7) Temporal dependent variables: Station data for snow measurements refers to meteorological observations collected at specific locations to quantify snow-related parameters. In our context, station data specifically refers to the recorded snow depth.

[6] https://land.copernicus.eu/pan-european/biophysical-parameters/high-resolution-snow-and-ice-monitoring

[7] https://modis-snow-ice.gsfc.nasa.gov/?c=MOD10_L2

[8] https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast

Wegaw provided us with data within the area of interest ranging from 2017 to 2021. Station data consists of scattered measurements on the geography of the area, and thus it needs to be processed before being used as a proper input for deep learning models. In the following, we discuss three different approaches to the same problem.

It is worth mentioning that for some samples our station data is more recent than our snow depth maps (Section II-B). In these situations, we simulate station data by utilizing the snow measurements from the corresponding location within the LiDAR snow depth maps.

The first variable introduced in our work comes from an internal study of Wegaw. They detected that a linear correlation exists between digital elevation models and the snow depth at macro-scale. The company suggested using this feature to our advantage.

We can compute a linear regression with the use of the aforementioned station data such as:

$$HS = \alpha d + \beta \tag{4}$$

where d is the elevation of the station and HS is the snow measurement.

Digital elevation models provide the height at each given point, and thus, once α and β are computed, we can apply the linear regression over the DEM model to generate a new feature that depends on station data, introducing the desired temporal variability.

Initially, we relied solely on DEMStat as our temporal dependent variable. However, we decided to also include new variables based on station proximity to further improve results.

As we have daily data for a station over six years, we can perform statistically relevant analysis. Normalizing snow depth values over the time series could make one station comparable to another, even if their location characteristics are highly different. We introduce two new features where station data has been normalized.

- A Voronoi map is a type of spatial partitioning diagram that divides a space into regions based on the closest proximity to a set of seed points, in our case, station data.

  One drawback to consider is that regions located farther away from measurement stations may exhibit lower accuracy compared to those closer by. Additionally, areas situated between two adjacent stations might experience a feature discontinuity, potentially leading to decreased model performance.
- To avoid the aforementioned discontinuity, we propose a new feature that aggregates scores by considering data from multiple stations. This score is defined as follows:

  $$ProbStat = \frac{\sum_i^N HS_i/d_i}{\sum_i^N 1/d_i} \tag{5}$$

  where N is the number of stations, HS is the snow depth measurement, and d is the distance to the station.

  One advantage of this approach is that it provides a continuous map that more accurately reflects distant areas from the stations. It is important to note that when the distance is zero, it is essential to include an epsilon value to prevent discontinuity.

Hereby we present three different methods of including temporal variability in our dataset. They are a key research focus as a high-quality temporally-dependent variable holds the potential for snow depth modeling on any day of the year.

III. Data Preparation

Although LiDAR maps offer exceptional precision, they do come with a drawback: drone-generated data often takes irregular shapes. This poses challenges when directly applying it to deep learning models, which typically expect square-shaped input data. Additionally, LiDAR-generated maps may contain cells with empty values, commonly referred to as *nodata* values. These empty cells can be attributed to various factors such as sensor limitations in capturing certain areas, occlusions caused by vegetation or buildings, or simply data gaps in the LiDAR scans due to terrain variations or scan angle limitations. It is worth mentioning that areas out of the interest area are also treated as *nodata* values. These limitations condition the generation of images needed to train our proposed model. For the sake of clearance, we refer to each pixel of the image as a grid cell and each image of $N \times N$ grid cells as a tile, which can be used to feed our network.

A high density of *nodata* values could have an undesired effect in the training phase. For this reason, we have selected tiles with a maximum of 30% of *nodata* values.

In this setup, if we were to crop rasters into tiles with a random algorithm, grid cells near limit zones could be underrepresented in the final dataset. In order to normalize the density of cell presence in our final dataset, an optimization algorithm was developed. It is an algorithm based on an iterative process. Firstly, it computes a raster indicating the spatial distribution of the generated tiles, then identifies the tile whose grid cells are least frequently represented in the final dataset on average and selects that tile as the new image to generate. The process continues until each grid cell in the original map appears a specified number of times on average in the final dataset. Fig. 4 illustrates the differences between following a random approach (RANDchete) and ours (MAXchete).

To the extent of our knowledge, there is no other deep learning approach that takes this into account. The algorithm can be publicly accessed throughout MAPchete library[9].

We have selected four as the final frequency, which indicates that, on average, each grid cell appears four times in our final dataset. It is worth mentioning that it effectively increases the representation of grid cells in the final dataset, functioning as a valuable data augmentation technique.

In Section VII-D we review how this algorithm also can be used to improve the smoothness of the predicted maps.

[9] https://github.com/abetatos/mapchete

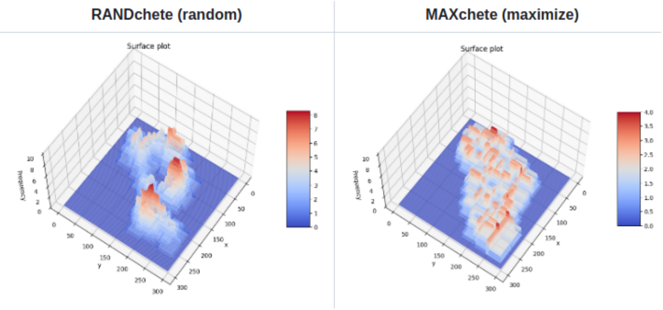


Fig. 4: Number of times a given cell appears in the final dataset with the use of a random approach (RANDchete) and with an optimization algorithm (MAXchete) after 100 iterations in the zone of Saflischpass.

For a given frequency, the number of tiles is dependent on their size. As can be seen in Table I, different configurations led to different numbers of images in our dataset. Among the different sizes considered, we selected 256 × 256 as it is a compromise between a higher spatial coverage and the number of images in the complete dataset.

TABLE I: Number of tiles in our dataset depending on the size of the tiles in the configuration described in Section III.

Size	512 × 512	256 × 256	128 × 128
No. tiles	1353	5333	21145

Once we had defined the input size, we projected the available data onto a consistent coordinate system. In particular, we used the EPSG:2056, which it is specifically designed for Switzerland. Subsequently, each input feature was resized to a uniform resolution of 5m, despite the more typical use of 10m resolutions. This choice was motivated by the availability of LiDAR maps with resolutions as fine as 2m. Our intention was twofold: to augment the number of training images, and to assess whether this improved resolution would yield satisfactory results.

IV. Network Architecture

Autoencoder architectures [28] have become state-of-the-art in approaching diverse issues. In particular, they are able to extract meaningful information, encode it into a dense vector representation, and decode it. There is a particular type, the convolutional autoencoders (CAE), which are able to generate complete images. Among all the approaches based on the encoder-decoder architecture, we have considered the UNet [20] for its proven competitiveness in different fields of application.

The UNet architecture consists of a downsampling path (encoder), an upsampling path (decoder), and a connection between them (bottleneck). The different components are depicted in Fig. 5 and described in the following.

- The encoder is composed of convolutional blocks which are followed by Rectified Linear Unit (ReLU) activation functions and maxpooling layers. The convolutional blocks increase the depth of feature channels, and the activation function introduces non-linearity to capture local information within the input image. Maxpooling layers then reduce the spatial resolution while preserving the most important features. This progressive reduction of dimensions allows the network to capture increasingly abstract and high-level features as it moves deeper into the encoder.
- The bottleneck of the architecture serves as a critical bridge between the encoder and decoder components. While it facilitates the connection between these parts, it also plays a crucial role in ensuring that the high-level features learned in the encoder's contracting path can be effectively utilized in the decoder's expanding path. The bottleneck section extracts complex representations by combining information from multiple channels while disregarding irrelevant information.
- The decoder is designed to reconstruct the segmentation map from the contracted feature map produced by the encoder. Each step in the decoder consists of an upsampling layer, which augments the spatial dimensions, followed by a convolutional block. It is worth noting that there is a skip connection from the encoder path to each step in the decoder. These skip connections enable the network to capture information from both low-level and high-level features. The low-level features capture fine details and local information, while the high-level features capture more global and abstract information. By combining information from different scales and levels of abstraction, the decoder effectively generates an image.

The architecture definition can be addressed by defining two hyperparameters:

- Depth: the number of max-pooling layers.
- Width: after each convolutional block, the number of channels increases, augmenting the feature representation of the input image. We define the width as a factor that increases the number of representative feature channels.

We define the number of channels as $n_{channels} = 2 \times width \times depth$.

The analysis of the parametrization can be found in Section VII-C.

It is important to highlight that the UNet architecture is typically utilized for segmentation purposes. However, to function as a regressor we eliminated the last activation function, which in the original architecture classifies each pixel into a given set of categories. In this adaptation, rather than assigning fixed labels, each pixel in the generated image represents a continuous value: the snow depth.

A. Masked Loss Function

The loss function measures the difference between a model's predictions compared to the expected output (base truth). By utilizing the derivatives of this function, the weights of the neural network can be adjusted in order to diminish the loss function, thus making predictions more similar to the expected

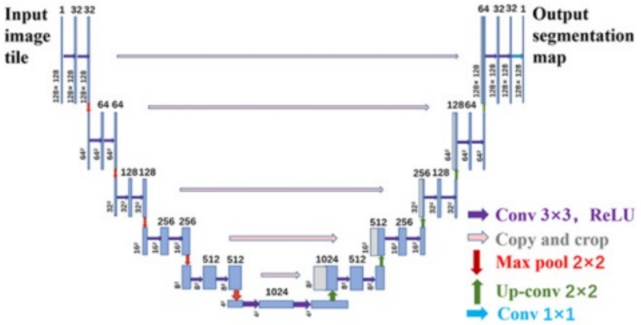


Fig. 5: Original UNet architecture [29] for an image of 32x32 pixels. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

output. A wide range of existing functions from the literature can be employed. However, a challenge arises when dealing with missing data values, as they lack meaningful information and should be disregarded when calculating the loss function. Consequently, it becomes necessary to develop a customized loss function specifically designed for handling such scenarios.

A masked loss function takes into account a masking mechanism for handling missing or irrelevant data. In our case, we used the quadratic loss and masked it following this expression.

Given a prediction $\hat{y}$ and the base truth y we defined our loss function as:

$$\mathcal{L} = \sum_{i,j}^{N} (\hat{y}_{ij} - y_{ij})^2, \, \forall y_{ij} \neq nodata \tag{6}$$

where y is the base truth, $\hat{y}$ is the prediction, and N is the number of pixels.

B. Optimizer

In deep learning models, optimizers are used to update the model weights during the learning process. It is important to note that not all optimizers are equally suitable for all tasks. We selected a combination of two common optimizers:

- SGD [30] is a simple optimization algorithm that updates model parameters based on the negative gradient of the loss function with respect to a batch of images. It performs updates after processing each sample or a small batch of samples. While SGD is straightforward, it can have slow convergence and exhibit oscillations in high-dimensional spaces.
- Adam [31] is an adaptive optimization algorithm that dynamically adjusts the learning rate for each parameter. It combines the benefits of RMSprop [32] and momentum methods, utilizing past gradient information to adapt the learning rate. Adam is efficient and performs well in a variety of scenarios.

For a more in-depth explanation, please refer to Section VII-C.

V. Implementation details

In our research, we employed specific hardware and software resources. Our setup consisted of an AMD Ryzen 7 7700X 8-Core processor and a NVIDIA GeForce RTX 4070 with 12GB of VRAM.

For the implementation of our project, we opted for Python as our primary programming language. We made use of open-source libraries to perform various computations, which are detailed below:

- RichDEM [33]: is an open-source library that provides many utilities for DEM processing. We used it to compute the Slope and the Aspect.
- Whiteboxtools [25]: is an open-source library that provides a wide range of geospatial analysis tools and functions for processing and analyzing geospatial data. We used it to compute the TPI function.

For data preprocessing and analysis, we utilized Rasterio [34], GDAL [35], and QGIS [36].

Our preferred deep learning framework is PyTorch [37].

VI. Performance measures

The first one is the mean absolute error or MAE. This error measure is defined as:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{7}$$

where n is the number of data points, y is the actual snow depth, and $\hat{y}_i$ is the predicted snow depth.

The second one is the root-mean-square error or RMSE. It is commonly used to detect predictions that are sensitive to extreme errors or outliers. RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{8}$$

where n is the number of data points, y is the actual snow depth, and $\hat{y}_i$ is the predicted snow depth.

For the sake of clarity, when we mention errors in the work, we are referring to the mean absolute error. Root-mean-square error only is presented in our final model for study comparability.

We also introduce the R^2 score, which is a measure of the goodness of a function fit.

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2} \tag{9}$$

where n is the number of data points, y represents the truth value, $\hat{y}$ is the predicted value, and $\overline{y}$ is the mean of the truth values.

TABLE II: Acquisition and Area Overview: Various Locations (2010-2022). The area is computed as the sum of each grid cell area, excluding those with a value of *nodata*.

Data partition	**Location**	**Year of acquisition**	**Area (km²)**
Train	Davos	2010	122.51
	Davos	2012	126.48
	Davos	2013	125.71
	Davos	2014	108.55
	Davos	2015	109.31
	Davos	2016	109.54
	Davos	2016	109.96
	Davos	2016	110.8
	Davos	2018	34.47
	Davos	2020	161.08
	Davos	2021	148.78
Val	Laucherenalp	2020	1.14
	Laucherenalp	2022	2.26
	Saflischpass	2022	4.48
	Saflischpass	2022	6.55
Test	Davos	2017	369.19
-	Total	-	1650.81

VII. Experimental Results

For experimentation purposes, we first split the dataset into training (66,4%), test (28,4%), and validation (4,8%) covering a total of 1651km² of raw data. In Table II a more in-depth characterization of the dataset can be found. We used the Davos zone for both training and testing, while the Saflishpass and Laucherenalp regions were used for validation. We decided to split the dataset by location to assess whether our model is prone to overfitting specifically to the Davos area or if it demonstrates generalization capabilities. Moreover, we decided to utilize the largest Davos map only for testing, allowing us to evaluate our model's ability to model snow depth under unfamiliar conditions. Moreover, making predictions over larger geographical regions can mitigate potential statistical errors that may arise from working with smaller maps. An example of our test set can be observed in Fig. 6.

It is worth noting that our dataset only comprises a total of sixteen maps over the span of ten years. As a result, addressing temporal variability can be challenging when we have a limited representation of different time periods.

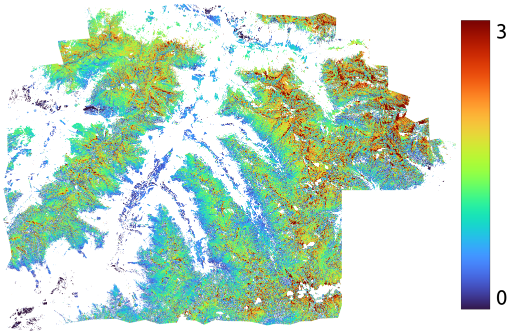


Fig. 6: Test area from our dataset, with the snow depth scale. The maximum has been set to 3.

A. Far From D

As introduced in Section II-C4, we must determine the minimum snow direction to generate this new input feature. To accomplish so, we make use of take advantage of Taylor's Theorem, which states that any derivable function can be approximated by a Taylor polynomial. We define a region where the minimum might be located, between 150 and 360º, and fit the polynomial, inferring the minimum with ease. The average value obtained is of $\vec{D} = 245 \pm 3$.

B. TPI

We conducted a thorough analysis of macro-scale correlations. The findings from this analysis helped us establish the desirable behavior of an ideal TPI. Since the TPI's lower values suggest areas where snow accumulates, it is reasonable to determine that the snow depth in these regions is greater. An effective TPI, in relation to snow depth, should demonstrate a monotonically decreasing relationship. This implies that as the TPI values increase, the snow depth decreases.

Fig. 7 showcases the observed tendency in the TPI's behavior, aligning with our defined criteria. To best represent the data described, we opted to fit the TPI to a sigmoid function due to its ability to accurately capture this behavior.

Between all the filter sizes analyzed $(5, 7, 9, 11, 13, 22, 50)$, we opted for the one that exhibited a superior R^2 score. Our analysis revealed that 11 was the optimal value for both TPI functions.

C. Hyperparameter configuration

We performed a study aimed at identifying the best hyperparameters for our architecture , using TPI, SCE, DEM, aspect, and slope as input variables. We observed that augmenting the depth of the model resulted in improved performance, although we encountered one limitation during this process. The inclusion of the max pooling layer, which reduces the size (width x height) by half at each step, led us to a maximum value of 7 for an input image of 256×256.

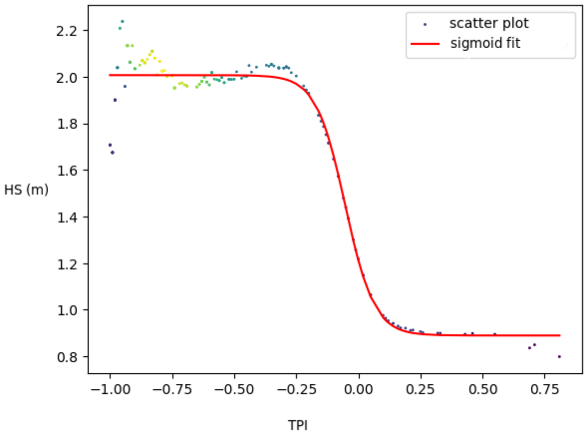


Fig. 7: Sliced snow depth vs TPI.

TABLE III: Average errors (in meters) for the different optimizers considered, using TPI, SCE, DEM, aspect, and slope as input variables. Best result is in bold.

Optimizer	SGD	Adam	SGD + Adam
MAE (m)	0.563	0.572	**0.540**

Subsequently, we explored the impact of varying the width parameter on the model's performance. We conducted an in-depth study, training the model with different channel widths (5, 7, 10, 12, 15, and 20). To assess the effectiveness of each width configuration, we evaluated the average error incurred during these experiments. Surprisingly, we found that a larger width did not necessarily yield a superior outcome. More specifically, we identified the width of 12 as the most advantageous choice. This particular width configuration demonstrated a remarkable 6% reduction in error compared to the second-best alternative (10).

With respect to the optimizer, we propose a composite one that combines SGD and Adam. Adam consistently provided stability while SGD exhibited instability but yielded superior values. This behavior led us to the idea of utilizing two optimizers in the same training procedure, SGD in order to improve the results and Adam to attain a more stable outcome.

For SGD we used a momentum of 0.9, a weight decay of 10^{-5}, and a learning rate of 10^{-3}, while for Adam we used a learning rate of 10^{-3}.

In Table III we can observe how the combination of SGD + Adam is not only stable but yields the most favorable outcomes resulting in an average error reduction of 2.3cm when compared to the use of SGD solely.

It is worth mentioning that this study is constantly validated throughout every experiment. A model with SGD and a model with SGD+Adam were always validated, observing this trend in error reduction when fine-tuning with Adam.

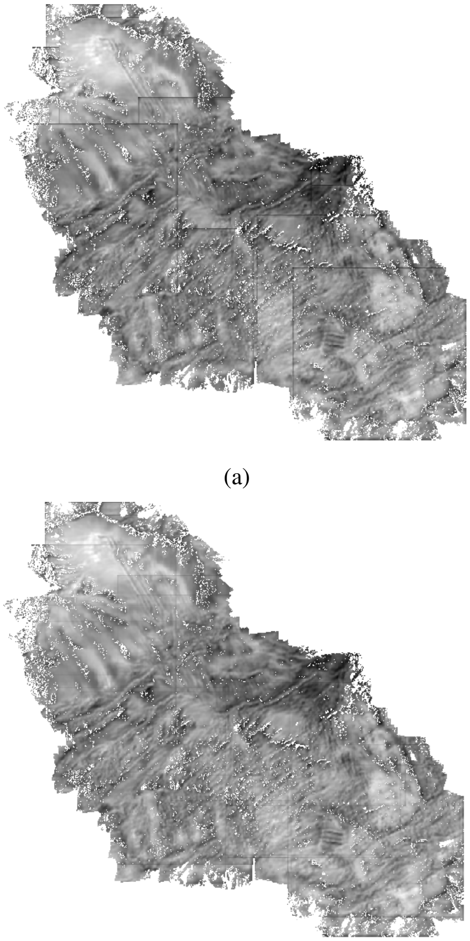
D. MAPchete as map generator

In the geospatial field, data is commonly sliced into sequential tiles that do not overlap. When recreating the complete map as limit areas between sequential tiles have a discontinuity in predictions. However, due to the nature of the Mapchete approach, we obtain tiles that do overlap, and thus we can follow an optimization procedure to avoid the aforementioned behavior. We decided to average the predicted tiles over space. Fig. 8 depicts an example of the map recreation procedure. The edge of the predicted tiles can be discerned when overlapping, but MAPchete offers a much more smooth map.

In order to quantify the effect of averaging, we studied how predictions change in the test area using the best model of the preceding section. We observe both a reduction in error $-10^{-3}cm$ and in standard deviation $-1cm^2$, which reflects how predictions are better and less sparse over space.

E. Feature informativity

We conducted a new experiment to determine which of the input variables was more informative. Table IV shows the results of an ablation study, which consisted of training and testing a model leaving out one of the variables. Notice that, in some cases, we took out two variables due to the correlation between them: DEM and DEM, and TPI and TPIWGW. This is pursued to see if eliminating both features could have positive effects on our model.


(a)

(b)

Fig. 8: Map recreation using (a) overlapping and (b) MAPchete in the area of Saflischpass.

The following conclusions can be drawn from the results:

- DEMStat is the feature that performs the worst. One potential justification behind this statement is that the linear relationship between altitude and snow depth may not work as a sufficiently reliable estimator, in contrast to our initial presumptions. Another potential reason is that without DEMStat, the system is overfitting to data from a specific day, given the constraints of our limited validation dataset. Consequently, when introducing temporal variability, the outcomes for that particular day deteriorate. However, these outcomes could likely improve with a more extensive validation dataset spanning multiple days.
- Notably, aspect emerges as the third most significant

TABLE IV: Average errors (in meters) achieved in the ablation study, using all the variables but the one shown. Best result is in bold.

Deleted variable	FFD	DEMStat	TPIWGW	TPI	TPI & TPIWGW	Slope	Aspect	**SCE**	DEM	DEM & DEMStat
Average error (m)	0.536	**0.528**	0.546	0.570	0.561	0.539	0.558	0.560	0.548	0.529

variable, which could imply that the model is non encountering the issues we initially anticipated in Section II-C3.

- TPI is the variable that performs the best. When left out the model enworsed by 1cm in predictions with respect to the second variable (SCE). Namely, the best predictors are TPI, SCE, Aspect, DEM, Slope, FFD, and DEMStat.

F. Iterative Predictor

To determine which variables should be included in our final dataset, we follow an iterative process. At each step, a model is trained adding variables one by one. The order of incorporation is that of the best variables found in the previous experiment.

Table V shows a deep improvement that stabilizes around 0.53m. With only SCE, TPI, aspect, and DEM the error is 0.532m on average, which is a 0.56% worse than the best model. Although an improvement is noted, we would like to highlight that if computational efficiency is pursued these variables could be sufficient for the purpose of snow depth predicting. In the following, we refer to this configuration as the *Efficient Set*.

TABLE V: Average errors (in meters) achieved with the iterative strategy, where the last variable added is the one shown. Best result is in bold.

Added variable	TPI	SCE	Aspect	DEM	Slope	FFD	DEMStat
Average error (m)	0.607	0.576	0.548	0.532	0.531	0.530	**0.529**

G. Aspect projection

Aspect is a non-continuous function with a step between 0 and 360º. This could represent a problem in the training procedure, as it is an extra characteristic the model should learn. Although this problem can be diminished if enough data is fed to the model, a common alternative found in the literature is to decompose it into sine and cosine in order to avoid any discontinuity. This procedure augments the number of input features and thus increases the computational time.

In this research, we train a model using the projections of the aforementioned functions instead of the aspect. We used the variables of the Efficient Set for this experiment.

We observe an average error of 0.533m, which is slightly higher than the ones obtained in the previous experiment. Therefore, we are able to conclude that for this dataset of 5333 images, we do not need the decomposition as no improvement is observed.

H. Another temporal dependency

As detailed in Section II-C7, we here introduce temporal variability into our model by creating three new features. We here test the last two of them: Voronoi and ProbStat.

We proceed to train two models using the information gathered from the preceding sections, specifically utilizing the Efficient Set. When incorporating Voronoi, we observe an error of 0.527m; whereas with ProbStat, the error improves slightly to 0.525m. This results in a reduction of error by 0.93% and 1.3%, respectively, compared to the model trained solely with the Efficient Set. To analyze in detail the impact of Probstat on our test set, we substitute this feature with a mask with values of 0, 1, and 0.5, and then assess how predictions deviate from the labeled dataset.

Fig. 9 illustrates the behavior of ProbStat. As expected, higher ProbStat values lead to higher predictions of snow depth. This characteristic can be valuable when there are enough monitoring stations in the area, as it adjusts snow predictions.

After confirming that ProbStat behaves as anticipated, we also point out the lack of temporal representation in our dataset as the main limitation, as discussed in Section VII-E. This scenario may lead to a situation of overfitting, hindering our model's ability to generalize.

It is worth mentioning that the unscaled predictions should form a linear relationship, with each prediction value matching the labeled one. However, as shown in Fig. 9, for higher depth values, the predictions tend to underestimate. In the next section, we take a deeper look into the error distribution.

We select the model trained with the Efficient Set and

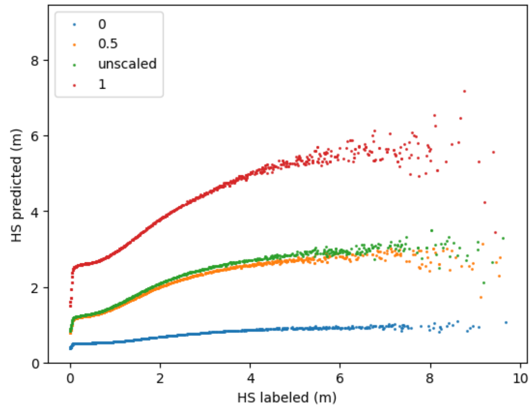
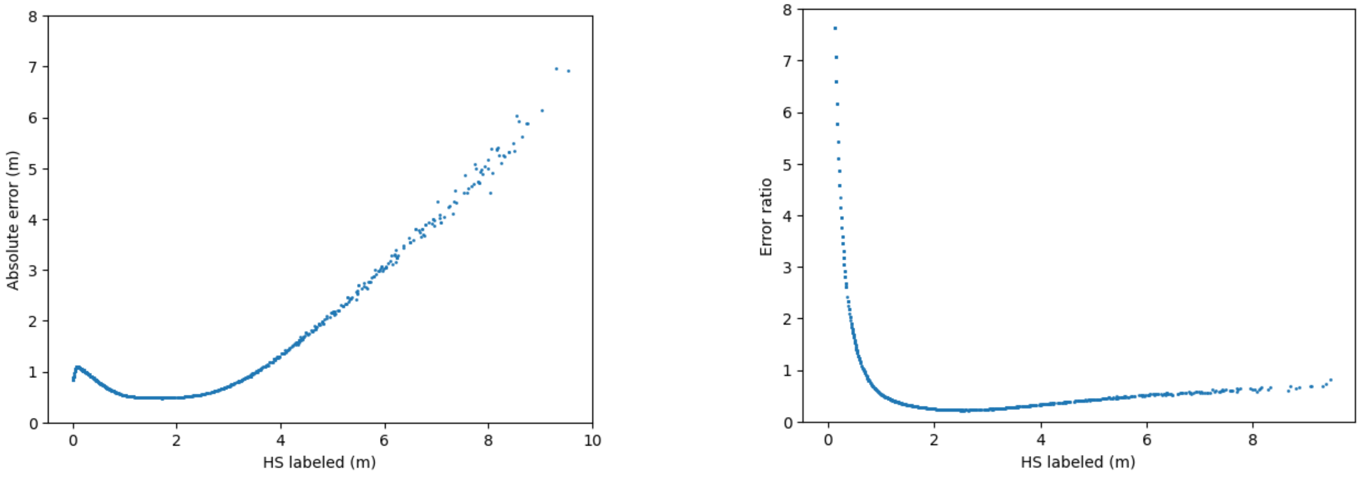


Fig. 9: Predictions in different masking configurations for ProbStat. Unscaled refers to the original feature.



(a)

(b)

Fig. 10: Error distribution with respect to snow depth (HS): (a) absolute error and (b) error ratio.

Probstat as our best model, achieving an MAE of 0.525m and an RSME of 0.634m.

I. Error distribution

By analyzing the error distribution, we gain insights into the accuracy and reliability of our predictive model. Such insights are pivotal in fine-tuning the model's parameters and identifying potential biases.

Fig. 10 shows the absolute error and the error ratio, defined as $Absolute\ error/HS$. The lowest error ratio is found at a snow depth of 2.52m. In terms of absolute error, the minimum occurs at a snow depth of 1.72m, with an MAE of 0.480m, which is 8.5% less than the average error. This information could prove valuable in an industrial product context, where predictions could be accompanied by a confidence interval based on the error distribution.

When we examine the error ratio more closely, it becomes apparent that our model experiences a significant increase in the error ratio for predictions below 0.5m. An interesting observation about the dataset is that snow depths under 0.5m are exclusively found in a specific region. This region possesses a distinctive characteristic: close to zero snow depth and predictions ranging from 0.5 to 1 meter. In our training set, only 14% of the samples have snow depths under 50cm. This underscores the significance of representing features accurately, specifically the limited representation of low snow depth in this case. On the other hand, the absolute error reveals that predictions for greater snow depths tend to be less precise compared to those for shallower depths. As mentioned in the previous section, there is a tendency for underprediction in higher snow depths. This may indicate that not only is more data needed in lower snow depths, but also in higher ones. In our training set, only 9.7% of the samples correspond to depths over 3m and only 3.0% over 4m. Both effects together suggest that an effort should be put into creating a more balanced dataset in terms of snow depths.

VIII. Conclusions

Our research addressed the challenge of generating accurate snow depth maps utilizing geospatial and satellite imagery. By employing the UNet architecture in a regressor mode, we have demonstrated its effectiveness in handling complex spatial data and producing high-quality results. Resolution also plays an important role in snow depth mapping, a higher resolution leads to more complex terrain and thus more complex snow accumulation patterns. To our extent, this is the first model to map accurately at only 5m resolution. Furthermore, recognizing the importance of temporal dependencies in modeling snow, we introduced three novel features: DEMStat, Voronoi, and ProbStat. Among them, the Probstat model achieved the best performance with an MAE of 0.525m error and an RSME of 0.634m.

We would like to highlight two contributions that could likewise enhance outcomes in other studies. Firstly, we introduce MAPchete, the novel data augmentation and preprocessing method. Secondly, the combination of two popular optimizers, SGD and Adam, have proven to be a powerful tool to enhance outcomes.

Our work contributes to the field of predictive snow depth modeling laying the groundwork for refining these models and leveraging novel features.

In this sense, four major areas for future investigation arise: modifying the architecture, working at other spatial resolutions, adding more input features, and gathering more data.

The first logical step involves exploring alternative architectural variations that have demonstrated success in related fields. More complex architectures could lead to more fine-grained predictions in regions with complex snow accumulation patterns.

Furthermore, extending the spatial resolution at which predictions are made represents a valuable strategy for enhancing the model's performance. While the current study employed a resolution of 5m, future research could evaluate the effectiveness of the UNet architecture at different resolutions.

Our current approach for snow depth prediction primarily relies on the UNet architecture and associated spatial information. Future investigations could incorporate additional features beyond spatial data, such as meteorological variables, historical snowfall patterns, or different terrain characteristics.

We detected little representation of data at shallow and deep snow depths. This could be approached two ways, whether gathering more data under those conditions or preprocessing the dataset having a more even distribution in terms of snow depths.

Acknowledgments

We would like to express our sincere gratitude to the Wegaw team for their kind contribution of data and unwavering support throughout the preprocessing phase of our project. We are also thankful to the master's degree tutors for their valuable assistance in the areas of deep learning and document review for our work. Additionally, we extend our appreciation to the main open-source libraries that played a pivotal role in facilitating our research, namely Rasterio, RichDEM, WhiteBoxTools, GDAL, and Torch. Your contributions have been instrumental in the success of our project, and we are truly thankful for your support.

References

[1] David R DeWalle and Albert Rango. *Principles of snow hydrology*. Cambridge University Press, 2008.

[2] Li Xu and Paul Dirmeyer. Snow–atmosphere coupling strength. part ii: Albedo effect versus hydrological effect. *Journal of Hydrometeorology*, 14(2):404–418, 2013.

[3] Chloé Largeron, Marie Dumont, Samuel Morin, Aaron Boone, Matthieu Lafaysse, Sammy Metref, Emmanuel Cosme, Tobias Jonas, Adam Winstral, and Steven A Margulis. Toward snow cover estimation in mountainous areas using modern data assimilation methods: A review. *Frontiers in Earth Science*, 8:325, 2020.

[4] Jiping Liu, Yuanyuan Zhang, Xiao Cheng, and Yongyun Hu. Retrieval of snow depth over Arctic sea ice using a deep neural network. *Remote Sensing*, 11(23):2864, 2019.

[5] Vahid Moosavi, Hossein Malekinezhad, and Bagher Shirmohammadi. Fractional snow cover mapping from MODIS data using wavelet-artificial intelligence hybrid models. *Journal of Hydrology*, 511:160–170, 2014.

[6] Christopher O Justice, Eric Vermote, John RG Townshend, Ruth Defries, David P Roy, Dorothy K Hall, Vincent V Salomonson, Jeffrey L Privette, George Riggs, Alan Strahler, et al. The Moderate Resolution Imaging Spectroradiometer (MODIS): Land remote sensing for global change research. *IEEE Transactions on Geoscience and Remote Sensing*, 36(4):1228–1249, 1998.

[7] Jesús Revuelto, Esteban Alonso-Gonzalez, Ixeia Vidaller-Gayan, Emilien Lacroix, Enaut Izagirre, Guillermo Rodríguez-López, and Juan Ignacio López-Moreno. Intercomparison of uav platforms for mapping snow depth distribution in complex alpine terrain. *Cold Regions Science and Technology*, 190:103344, 2021.

[8] Jeffrey S Deems, Thomas H Painter, and David C Finnegan. Lidar measurement of snow depth: a review. *Journal of Glaciology*, 59(215):467–479, 2013.

[9] Yves Bühler, Marc S Adams, Ruedi Bösch, and Andreas Stoffel. Mapping snow depth in alpine terrain with unmanned aerial systems (UASs): potential and limitations. *The Cryosphere*, 10(3):1075–1088, 2016.

[10] Hendrik Wulf, Bernhard Sassik, Gillian Milani, and Reik Leiterer. High-resolution snow depth monitoring for entire mountain ranges. In *7th Swiss Conference on Data Science*, pages 1–4. IEEE, 2020.

[11] Kelsey Cartwright, Craig Mahoney, and Chris Hopkinson. Machine learning based imputation of mountain snowpack depth within an operational Lidar sampling framework in Southwest Alberta. *Canadian Journal of Remote Sensing*, 48(1):107–125, 2022.

[12] Adam Winstral, Kelly Elder, and Robert E Davis. Spatial snow modeling of wind-redistributed snow using terrain-based parameters. *Journal of Hydrometeorology*, 3(5):524–538, 2002.

[13] Yanxing Hu, Tao Che, Liyun Dai, and Lin Xiao. Snow depth fusion based on machine learning methods for the Northern Hemisphere. *Remote Sensing*, 13(7):1250, 2021.

[14] Jiwen Wang, Qiangqiang Yuan, Huanfeng Shen, Tingting Liu, Tongwen Li, Linwei Yue, Xiaogang Shi, and Liangpei Zhang. Estimating snow depth by combining satellite data and ground-based observations over alaska: A deep learning approach. *Journal of Hydrology*, 585:124828, 2020.

[15] Alireza Yekta Meyal, Roelof Versteeg, Erek Alper, Doug Johnson, Anastasia Rodzianko, Maya Franklin, and Haruko Wainwright. Automated cloud based long short-term memory neural network based swe prediction. *Frontiers in Water*, 2:574917, 2020.

[16] De Xing, Jinliang Hou, Chunlin Huang, and Weimin Zhang. Estimation of Snow Depth from AMSR2 and MODIS Data based on Deep Residual Learning Network. *Remote Sensing*, 14(20):5089, 2022.

[17] Rodrigo Caye Daudt, Hendrik Wulf, Elisabeth D Hafner, Yves Bühler, Konrad Schindler, and Jan Dirk Wegner. Snow depth estimation at country-scale with high spatial and temporal resolution. *ISPRS Journal of Photogrammetry and Remote Sensing*, 197:105–121, 2023.

[18] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

[19] Lucie A Eberhard, Pascal Sirguey, Aubrey Miller, Mauro Marty, Konrad Schindler, Andreas Stoffel, and Yves Bühler. Intercomparison of photogrammetric platforms for spatially continuous snow depth mapping. *The Cryosphere*, 15(1):69–94, 2021.

[20] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[21] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[22] Linglong Zhu, Yonghong Zhang, Jiangeng Wang, Wei Tian, Qi Liu, Guangyi Ma, Xi Kan, and Ya Chu. Downscaling snow depth mapping by fusion of microwave and optical remote-sensing data based on deep learning. *Remote Sensing*, 13(4):584, 2021.

[23] Xin-Yue Zhong, Tingjun Zhang, Hang Su, Xiong-Xin Xiao, Shu-Fa Wang, Yuan-Tao Hu, Hui-Juan Wang, Lei Zheng, Wei Zhang, Min Xu, et al. Impacts of landscape and climatic factors on snow cover in the altai mountains, china. *Advances in Climate Change Research*, 12(1):95–107, 2021.

[24] Jesús Revuelto, Paul Billecocq, François Tuzet, Bertrand Cluzet, Maxim Lamare, Fanny Larue, and Marie Dumont. Random forests as a tool to understand the snow depth distribution and its evolution in mountain areas. *Hydrological Processes*, 34(26):5384–5401, 2020.

[25] JB Lindsay. The whitebox geospatial analysis tools project and open-access gis. In *Proceedings of the GIS Research UK 22nd Annual Conference, The University of Glasgow*, pages 16–18, 2014.

[26] DR Newman, JB Lindsay, and JMH Cockburn. Evaluating metrics of local topographic position for multiscale geomorphometric analysis. *Geomorphology*, 312:40–50, 2018.

[27] Manuel Grizonnet. An operational snow cover product from Sentinel-2 and Landsat-8 data for mountain regions. In *La montagne, territoire d'innovation*, 2017.

[28] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991*, 2020.

[29] Getao Du, Xu Cao, Jimin Liang, Xueli Chen, and Yonghua Zhan. Medical image segmentation based on u-net: A review. *Journal of Imaging Science and Technology*, 2020.

[30] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952.

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[33] Richard Barnes. *RichDEM: Terrain Analysis Software*, 2016.

[34] Sean Gillies et al. Rasterio: geospatial raster i/o for Python programmers, 2013–.

[35] GDAL Development Team. *GDAL - Geospatial Data Abstraction Library, Version x.x.x*. Open Source Geospatial Foundation, 201x.

[36] QGIS Development Team. *QGIS Geographic Information System*. QGIS Association,

[37] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.