Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

IGP
Institut für Geodäsie und Photogrammetrie

Patrick Barton

Machine learning for mapping avalanches in optical satellite imagery

Master Thesis

EcoVision Lab
ETH Zurich

Supervision

EcoVision	**SLF**
Andrés Rodríguez	Elisabeth Hafner
Dr. Jan Wegner	Dr. Yves Bühler

February 2021

Contents

Abstract

Avalanches pose a prominent threat in alpine regions and managing this is a difficult task. Currently, only sparse information on the occurrence of avalanches is available from observation in the field. Remote sensing methods have been proposed to help automate avalanche mapping, but have mainly been limited to synthetic aperture radar. Very high resolution optical satellite imagery provides an alternative and has already been used by the SLF for manually analysing avalanche events. This thesis introduces a method for automating the mapping process in optical imagery on country-wide scale. Models from semantic segmentation tasks are used and extended to tackle the challenges specific to avalanche mapping with a custom architecture. Results demonstrate deep learning methods can be used to map avalanches successfully, achieving an F1 score of 63 %. The methods are evaluated in detail showing their strengths and weaknesses, and are further analysed for their generalisability to new data. The method demonstrates optical satellite images can be used for avalanche mapping, reaching a similar performance to those using SAR satellite images.

Nomenclature

Acronyms and Abbreviations

SLF	Institute of Snow and Avalanche Research
EAWS	European Avalanche Warning Services
DEM	Digital Elevation Model
BGR-NIR	Blue, Green, Red and Near-Infrared channels
SOTA	State of the Art
VHR	Very High Resolution
SAR	Synthetic Aperture Radar
FAR	False Alarm Rate
IOU	Intersection over Union
POD	Probability of Detection
PPV	Positive Predictive Value
FCN	Fully Convolutional Network
BCE	Binary Cross Entropy
RMSE	Root Mean Square Error
GPU	Graphics Processing Unit
SGD	Stochastic Gradient Descent
GDAL	Geospatial Data Abstraction Library
VRT	Virtual Raster Format
RAM	Random Access Memory
VRAM	Video RAM
GT	Ground Truth

Introduction

This report presents my master thesis on automatic avalanche mapping in optical satellite imagery using machine learning. It was conducted at the EcoVision Lab at ETH Zurich in collaboration with the SLF for a period of six months. The term avalanche specifically refers to snow avalanches, which will be assumed from here on.

In the following pages, I detail the various steps undertaken, starting with the motivation for this work and data available. I will review related work in chapter 2 and continue to explain the proposed approach in chapter 3. My findings will then be examined and analysed in detail in chapter 4 before ending with some concluding remarks in chapter 5. All citations can be found at the end of the document. To find the meaning of Acronyms used, please check the list at the beginning of the document.

For information on the implementation details, please see the github repository and corresponding readme files there.

1.1 Motivation

Avalanches pose a significant hazard in alpine regions, such as the Swiss alps. They are responsible for road closures, damage to infrastructure and a number of fatalities every year [23]. Therefore, understanding avalanches and managing this risk is of consequential importance.

As part of this, a good understanding of where avalanches occur and at what size is paramount. Currently however, this information is only available in sparse form from observations in the field. Avalanches are either reported by people living or traveling in the mountains, or when they cause problems such as covering roads or damaging infrastructure. A considerable number of avalanches remain unregistered in areas that are remote, or inaccessible due to the avalanche danger itself.

A more complete picture of where exactly avalanches have occurred could help with various aspects as discussed in [9] including:

- Avalanche Warning: validation of the avalanche forecast made by the SLF twice a day

- Hazard Mapping: to consider when building new infrastructure
- Hazard Mitigation Measures: validation of effectiveness and planning of new infrastructure such as diversion structures
- Risk Management: understanding the severity of events and estimating cost-effective solutions
- Numerical simulations: validation of avalanche models

To achieve such a large scale mapping, remote sensing methods, especially with regard to satellites, have a strong potential to solve the problem. These could capture data for vast areas of land on demand, without any on site involvement. They can therefore be performed safely, independent of the current conditions. Such a mapping process could be deployed on a country-wide level or even a complete mountain range.

Manual mapping as was done by the SLF in [2] requires a lot of time and someone trained to recognise the avalanches. This is a slow and expensive process. It is therefore necessary to automate the mapping process for it to become affordable and the results to become available more immediately.

1.1.1 Sensor options

When it comes to remote sensing, there are two main sensor types to be considered for avalanche mapping. Firstly, there are passive sensors that collect solar radiation reflected from the surface of the earth. These can be obtained in the visible spectrum, infrared and at other frequencies. Secondly, there are active sensors such as Synthetic Aperture Radar (SAR) that use their own energy sources.

The SLF has already performed an investigation into the most appropriate methods in [9], in which they evaluate satellite images from SPOT-6, Sentinel-1 and Sentinel-2 for avalanche mapping. They conclude that optical imagery with BGR-NIR bands from SPOT-6 are most suitable. While Sentinel-2 was found to be too low resolution for avalanche mapping, SAR images from Sentinel-1 still have much potential. Since much of the related work in chapter 2 uses SAR, I summarise the advantages of each in the context of avalanche mapping in Table 1.1. A visual comparison of optical and SAR images is provided in Figure 1.1.

VHR Optical – SPOT-6	**SAR – Sentinel-1**
Daily revisit capability	Acquisition in all light and weather conditions
Spatial resolution of 1.5m well suited for avalanche detection	Suited for detection of large avalanches
Images are more easily interpretable	avalanche deposits show more contrast
Avalanche release zones visible	Data is free of charge

Table 1.1: Advantages of optical and SAR satellite images for avalanche mapping.

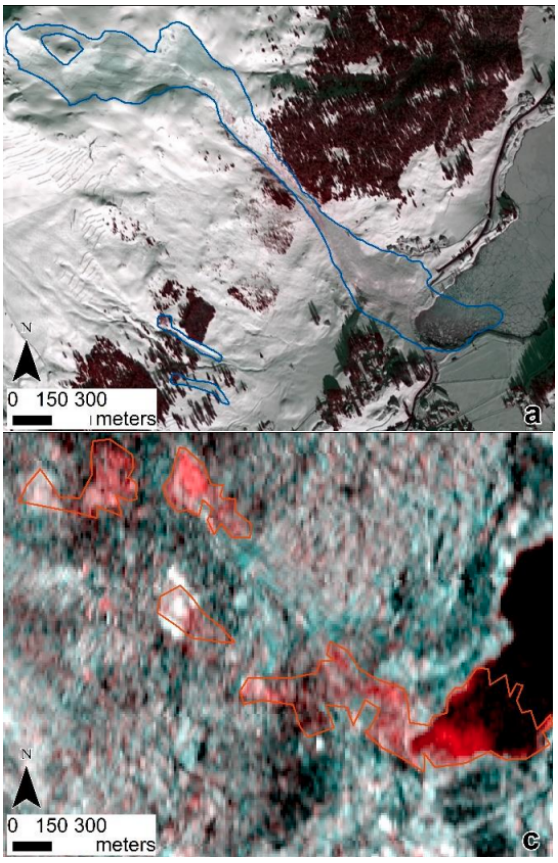


Figure 1.1: Comparison between avalanche mapping in an optical image (top) [SPOT6 ©Airbus DS2019] and SAR image (bottom) [Sentinel-1 ©Copernicus (2019)] from [9]. The SAR image is a RGB composite of the pre-and post-avalanche event local resolution weighting (LRW) image. Red represents the post-event image, while green and blue represent the pre-event image.

1.2 SLF avalanche dataset

The SLF provided a dataset of 24 776 avalanches, labelled in optical satellite images. These were captured by SPOT-6 during two significant avalanche periods on the 24$^{\text{th}}$ January 2018 and 16$^{\text{th}}$ January 2019, hereafter simply referred to as 2018 and 2019. Both times the avalanche warning level was at the maximum level which had not been given since 1999 and covered a considerable area of the Swiss alps, as illustrated in Figure 1.2. Due to the unusual circumstances, the SLF obtained satellite images of the corresponding regions and mapped avalanches manually to help analyse the situation. Relevant information on the dataset is summarised in the following subsections. For more details, please refer to [2].

1.2.1 Satellite images

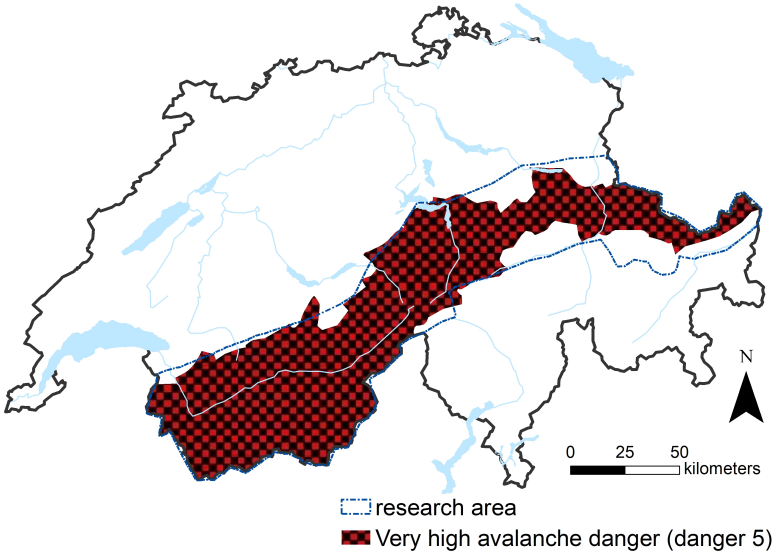
The satellite images are provided in the form of an image mosaic which can be treated like a single image. For convenience, I will refer to each image mosaic as a single image. Therefore, the dataset can be considered to contain two images, which cover an area of 10 276 km^2 and 5105 km^2 in 2018 and 2019, respectively. The images have been orthorectified by Swisstopo with an accuracy (RMSE) of 1.23 m in X, 0.83 m in Y and 0.16 m in Z on 11 ground control points. Due to the acquisition angle of the images, which reached a maximum inclination of 21.5° and 27.6° in 2018 and 2019, respectively, this has resulted in some distortions in steeper terrain. Consequently, avalanches are harder to detect in those places and labels less precise. An extreme example is shown in Figure 1.3, although most areas are affected more mildly.

The satellite images themselves are in the BGR-NIR bands and stored with 16 bit each. This high bit depth is important such that there is enough contrast in shadow areas. The spatial resolution of the images is 1.5 m. While not a problem for this thesis, it should be noted that some areas from 2019 (specifically images ortho_7_xxx) were accidentally delivered at a 2 m resolution. This should be considered when building upon this work.

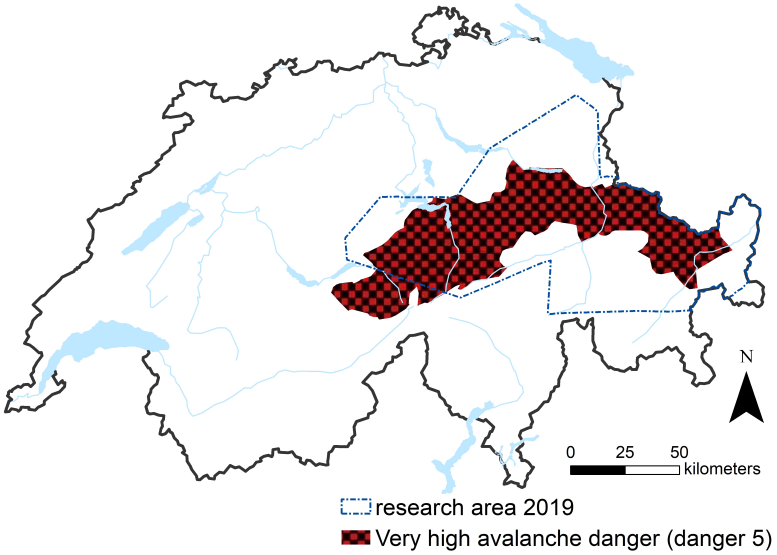
1.2.2 Labels

The manual avalanche mappings were created by an expert at the SLF, and will be referred to as labels, in line with the machine learning literature. They are considered the ground truth in this work. The labels are provided in the form of polygons describing the outline of each avalanche with some additional attributes. This includes for example the type of avalanche and its size. Importantly, labels are also categorised by the quality of their outline, as judged by the person labelling them. There are three levels which feature in images with the colors noted in brackets:

- Exact (Green): the outline is clearly visible everywhere
- Estimated (Orange): the outline is visible in most places
- Created (Red): only release zone or deposit is visible with the rest interpreted according to the terrain



(a) 2018. SPOT-6 images cover $\sim$12 500 km^2



(b) 2019. SPOT-6 images cover $\sim$9500 km^2

Figure 1.2: Area of the Swiss alps that had the maximum avalanche warning level 5 shown in red/black and corresponding area captured by SPOT-6 for the dataset outlined in blue [2].

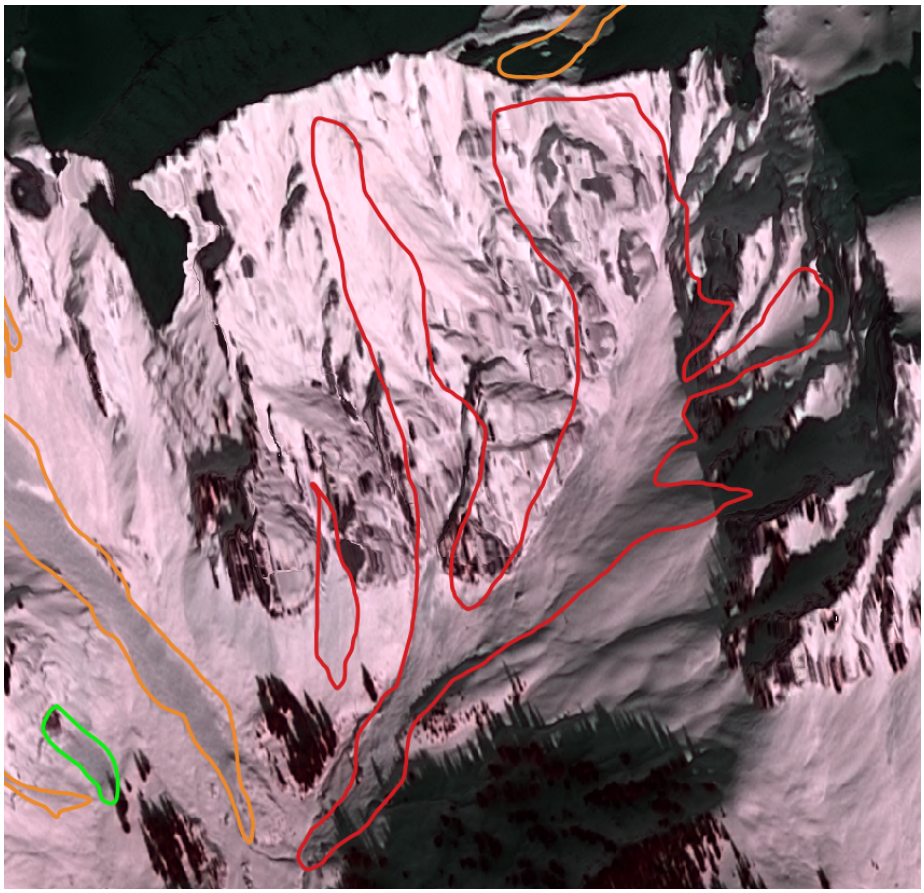


Figure 1.3: An area captured with a high inclination angle resulting in distortions in steep terrain of the orthorectified image. Avalanche mapping is harder in affected areas. The manual mapping outlines are overlayed with the colors as described in subsection 1.2.2. [SPOT6 ©Airbus DS2019]

Most avalanche outlines are not entirely clear (estimated or created), as can be seen in Figure 3.10 of subsection 3.4.2. This poses a particular challenge for any automated algorithm since it has to develop a good enough understanding to be able to interpolate correctly.

It is worth noting that although the satellite images cover a large geographical area, the weather and snow conditions are relatively similar across each image mosaic. While both images were captured with cloud free skies, there are considerable differences between 2018 and 2019. In 2019, the snowline lay lower, with snow fall all the way down to the valleys. Due to the colder weather, avalanches were also drier than in 2018, were the warmer conditions triggered more glide and wet snow avalanches. Since dry avalanches generally leave a smaller deposit and show less contrast in the infrared band, this makes the images of 2019 slightly harder for mapping avalanches.

1.2.3 Label validation

Mapping avalanches in optical satellite imagery is a difficult task, even for humans. In order to validate the correctness of manual labels, the SLF conducted a thorough evaluation of the dataset as described in [9]. Using pictures taken from the valley and by helicopter they compared the quality of labels, as well as checking their correctness. This was done on a relatively small area of 180 km^2 around Davos but is assumed to be similar over the whole area.

They found that avalanches mapped from SPOT-6 imagery had a probability of detection (POD) of 74 % and a positive predictive value (PPV) of 88 %. These metrics are equivalent to the recall and precision respectively, as used in machine learning literature. In particular, small avalanches and those completely in the shade were missed most frequently, as they were often not detectable in the optical images. To be more precise, [9] showed that only 45 % of size 1 avalanches and 66 % size 2 avalanches (as defined by the EAWS) were detected, while almost all larger avalanches were detected. Furthermore, only 15 % of avalanches were detected in shaded slopes compared to 86 % in fully illuminated areas. This is important to remember throughout this work. Although VHR optical satellite images can help detect a large proportion of avalanches, it remains a challenging task.

Related Work

Although automatic avalanche mapping in optical satellite imagery has already been considered in earlier works, more recent work has focused on using Radar instead. Accordingly, I will be examining the existing work using optical images first in section 2.1, before looking at that using radar in section 2.2. A summary of the methods and their limitations is shown in Table 2.1. For a comparison of the sensor types, see subsection 1.1.1.

Furthermore, as the task essentially boils down to a semantic segmentation task, I will also be discussing related work with regard to this from the deep learning field in section 2.3.

Sensor type	**Method**	**Limitations**
Optical	Hierarchical object-oriented image analysis [15]	Sensitive to noise. Cannot distinguish deciduous tree from avalanches
	Additional use DEM. Custom texture filters and k-means clustering [14]	Fragmented predictions
SAR	Difference analysis between pass-overs [6]	Long revisit period of 6 days in the alps
	Detection with VGG16 network in predefined areas [22]	Only classification not segmentation
	Segmentation with UNet [1]	Limitations of SAR

Table 2.1: Summary of related methods and their limitations.

2.1 Optical

Optical satellite images were already considered for automatic avalanche mapping by the SLF back in 2012 [15]. In addition to using aerial images, Very High Resolution (VHR) panchromatic QuickBird satellite images with a resolution of 0.61 m–0.72 m were used as input. These were then processed using traditional methods including hierarchical object-oriented image analysis techniques achieving an accuracy of 90.0 %. The algorithm was however found to

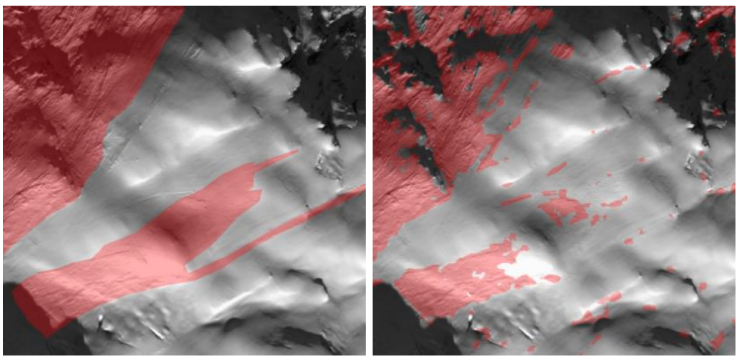


Figure 2.1: Results from Larsen et al. show the ground truth label (left) along with the prediction (right). The predictions are quite fragmented leaving room for improvement.

be sensitive to noise and could not distinguish deciduous trees without leaves from avalanches.

The Norwegian Computing Center built upon this work in the following year, additionally making use of a digital elevation model (DEM) [14]. They also used QuickBird satellite images and processed them with traditional methods involving custom texture filters and k-means clustering. Results were however often fragmented and while no metrics were provided, visual examination in Figure 2.1 shows there is still room for improvement.

Both works were only examined on a relatively small area and hence their generalisability to different terrain, weather, lighting and snow conditions is questionable. Since 2013 no more work on automatic mapping of avalanches in optical satellite images has been published, highlighting that this is a difficult task.

2.2 Radar

It has only been in the last couple of years, that there have been publications on automatic avalanche mapping again. This time however, all research is using Sentinel-1 Synthetic Aperture Radar (SAR), instead of optical data. This is likely due to Sentinel-1 SAR images that have become freely available, which is what all of the recent works have used. Although spatial resolution is a more complex topic for SAR, it can generally be said that Sentinel-1 SAR has a lower resolution of 10 m per pixel.

The Earth Observation Group from the Norwegian Research Center has developed a system that can detect avalanches within 10min of a Sentinel-1 pass over [6]. This works by analysing the differences in the images between pass overs, achieving a probability of detection (POD) of 67.2 % and false alarm rate (FAR) of 45.9 %. Unfortunately, this method relies on a high pass over frequency, since older avalanches are harder to detect. While higher latitudes have a revisit period of 1–3 days, the alps are only revisited by Sentinel-1 every 6 days making the algorithm less appropriate there.

At the same time, researchers from France applied deep learning techniques for the automatic mapping from a single SAR image [22]. They used a local database of 4000 avalanches to fine tune the last layers of a VGG16 network. This was able to classify whether a image patch contained an avalanche or not with a 77 % confidence. Note while this was the first demonstration of using deep learning methods for avalanche mapping, they did not perform a segmentation of the avalanche but only classified whether known avalanche corridors contained new avalanches.

The most recent work [1] performs complete segmentation of avalanches using deep learning. This is effectively the equivalent of my thesis but for radar data and therefore the most similar. It was however, only published half way through my thesis. Interestingly, they use very similar methods and also make use of a DEM, although they integrate it in the form of attention, to mask out areas where avalanches are unlikely. By training their UNet [21] architecture on a database of 6000 avalanches, manually labelled by experts, they achieve an F1 score of 66 %. Note that, while the task remains the same with different data, VHR optical data still has much potential. In fact, [9] showed that VHR optical images can be used to detect 66 % of size 2 and 91 % size 3 avalanches as defined by EAWS, while only 12 % and 42 % were recovered with Sentinel-1 SAR images, respectively. A similar performance in optical images could therefore provide more precise mappings and be able to detect smaller avalanches as well.

2.3 Semantic segmentation

Semantic segmentation is a computer vision task in which every pixel is assigned a label, according to its underlying contents. This is usually done with respect to the type of objects in the image. For example, one may segment the road, cars, people, trees etc. in an image as is shown in an example from the Cityscapes dataset [4] in Figure 2.2.

While semantic segmentation of images is a more general problem, the same methods can be used for avalanche mapping. The topic has been a huge research field in the deep learning community, with much research being pushed by the autonomous driving industry. Consequently, there is already plenty of related work to build upon as summarised in [19].

One of the state of the art (SOTA) models is Deeplabv3+ introduced in [3]. This uses a dilated resnet backbone, before applying its key atrous spatial pyramid pooling (ASPP) module and upsampling with the help of low-level features from the backbone. With this setup, it manages to achieve a mean IOU of 89 % on the PASCAL VOC 2012 [7] which indicates that it is highly capable of learning a good segmentation. It is however, a relatively big and complex architecture requiring large amounts of training data and processing.

On the other hand, smaller and simpler architectures such as the UNet [21] have been used extensively. UNets have an encoder-decoder structure, whereby the spatial resolution of feature maps is reduced with depth. In the upsampling path features are then combined with those of the corresponding resolution from the encoder with so called skip connections. UNet type architectures have been used successfully in all sorts of segmentation applications ranging from the medical

Figure 2.2: Semantic segmentation example from the Cityscapes dataset [4]. Each color corresponds to an object class such as road, car, or person.

domain to remote sensing [18], where the smaller model is often preferred over the more complex Deeplabv3+.

Methodology

This chapter explains the approach used to solve the problem. The problem definition is clarified first before looking at the reasons for using deep learning. I explain the models used, and architectural changes made. Finally, I explore how the SLF dataset is used and the models are trained.

3.1 Problem definition

The goal of this work, was to develop a method to automatically map avalanches in optical satellite images. It should automate the process of manually labelling the images as was done by the SLF in the data described in section 1.2.

The input is a single orthorectified BGR-NIR image with 1.5 m resolution from SPOT-6. Due to the high costs associated with VHR optical satellite images, no secondary images from the days before are available for comparison, as was the case in [6]. The algorithm must therefore be able to recognise the avalanches from a single image only.

As the output of the algorithm, a map describing the avalanches needs to be produced. Separating different avalanche instances was not required. While the manual labels also included additional attributes such as the avalanche type, this was not expected from the algorithm. The focus was simply to recognise and segment the avalanches from the images. The output is therefore a segmentation mask, whereby each pixel is assigned a label: avalanche or no avalanche.

3.2 Proposed approach

The problem described in section 3.1 is a typical semantic segmentation problem, as it known in the computer vision field. Image segmentation is a common task that has long struggled with more complex examples. It has only been in the last decade, with the advent of deep learning, that the task has seen huge improvements. Fully convolutional networks (FCN) that can handle different input sizes have proven very good at the task, achieving close to human level performance. It is therefore the method of choice, when it comes to image segmentation tasks.

Deep learning methods require large datasets for training, but fortunately, this requirement is fulfilled with the dataset provided by the SLF. The nearly 25 000 manually labelled avalanches can be used to train a network. When combined with transfer learning methods this should be enough data to learn good representations. Related work discussed in section 2.2 supports this assumption, having used transfer learning techniques to train a network on 6000 avalanches only.

Note that deep learning methods all follow the same basic principle. A model acts like a black-box taking an input and directly computing the outputs. This is done through an underlying network which can take on different architectures. The network itself is paramaterised through millions of parameters which first need to be trained to approximate the task, using dataset examples. Once trained, the model can then be used in its black-box form to make predictions directly from the inputs.

Due to the nature of these models, the network architecture effectively defines how much can be learnt and how detailed predictions can be. The predictions themselves are however determined by the training data and process. Therefore, the same models can be used for different tasks by training on different data. Accordingly, I can directly make use of the existing semantic segmentation models described in section 2.3, by training them on the SLF dataset to predict avalanches, instead of cars for example. The models used are examined in detail in the following section.

It should be mentioned that while the this work mainly focused on semantic segmentation, instance segmentation methods were also explored. These are a more advanced form of segmentation that can additionally differentiate between multiple objects of the same class. Problems with avalanche mapping using semantic segmentation models indicated that it might benefit from such a change, although empirical results did not support this. For more information on the method, see appendix A.2.

3.3 Model architecture

First tests were conducted with the UNet architecture [21], which has performed well in related work as discussed in section 2.3. Although it has an appealing light weight structure, it struggled to converge and was yielding poor results. Therefore, I decided to use the more complex Deeplabv3+ [3] instead. Deeplabv3+ proved to perform a lot better and so I used it as a baseline which I expanded upon. In the following subsections I will discuss this baseline and then introduce the modifications I made to form my own Model which I call AvaNet.

3.3.1 Deeplabv3+

Deeplabv3+ is a semantic segmentation model that has achieved SOTA performance on various datasets [3]. The architecture is visualised in Figure 3.1. It works by using a dilated resnet encoder as a backbone for feature extraction (although other backbones may be used in its place). The high-level features

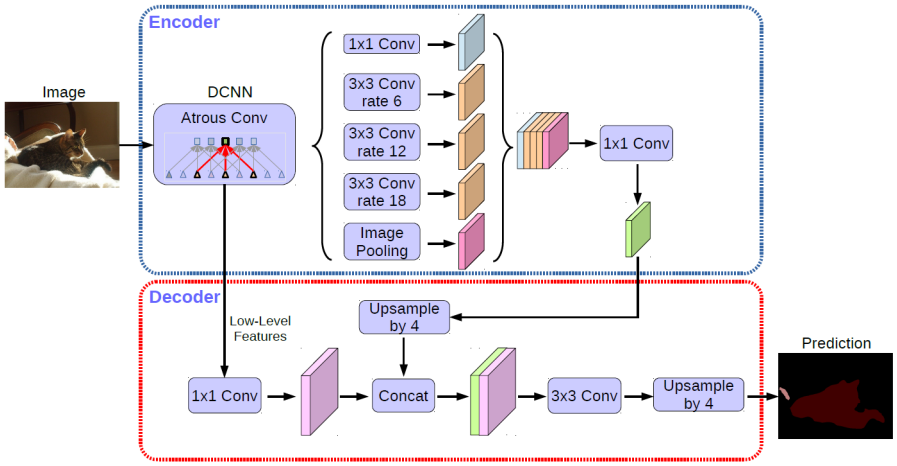


Figure 3.1: Deeplabv3+ architecture from [3]

are then processed with the key atrous spatial pyramid pooling module (ASPP), which uses several dilated convolutions at different rates to capture a wide receptive field. The resulting features are concatenated and combined with a 1x1 convolution, before being fed to the decoder. In the decoder, the high-level features are upsampled bilinearly, and combined with the higher resolution low-level features from the first resnet layer. This is done with a concatenation followed by a convolution. The result is finally upsampled bilinearly by a factor of four to match the input resolution.

3.3.2 AvaNet

AvaNet is a custom architecture based on Deeplabv3+, designed to alleviate some of its shortcomings and improve it for the specific task of avalanche mapping. It adapts the encoder by making it deformable, feeding in offsets from a smaller network called OffsetNet using the DEM only. The decoder is further extended to process features from all backbone layers. An overview is shown in Figure 3.2. The main motivation for this new architecture was to make better use of the DEM which will be explained in subsection 3.4.3. Experiments had shown that Deeplabv3+ was not using the DEM when simply adding it as an extra input channel to the network. Predictions were still being made in flat terrain were avalanches cannot occur. This highlighted that the model was not learning as much as it should and the DEM needed to be integrated into to the model more explicitly.

Although AvaNet makes several changes to the architecture described below, the impact in terms of the number of parameters remains relatively low. In fact, OffsetNet from the deformable backbone only adds 0.2M parameters and the decoder adds 1.7M parameteres. This is relatively little compared to the size of resnet18 with 11M and resnet34 with 21M parameters.

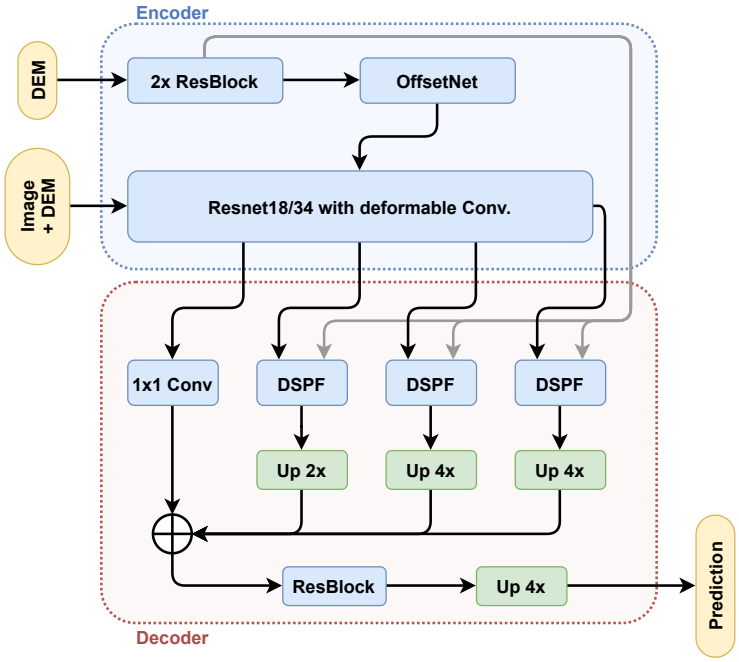


Figure 3.2: AvaNet overview. Details of the individual parts are given in Figures 3.4 to 3.7. A legend of the diagram elements can be found in Figure 3.8.

Deformable backbone

The main change made to integrate the DEM into the model more deeply, was to adapt convolutional kernels in the resnet backbone according to the DEM. The intuition is that the convolution kernel should be transformed according to the underlying terrain. One could for example rotate the kernel to align with the slope, or stretch it to increase the receptive field uphill.

To implement this, I make use of the deformable convolution [5]. This takes an additional 18 channel tensor as input, which encodes the 2D offsets of each kernel element at each location. With this, the kernel can be transformed in an arbitrary way, more general than a simple rotation or dilation. A visualisation of this operator is shown in Figure 3.3. Importantly, the offsets are not set during initialisation but fed in during the forward pass. The offsets are therefore not fixed but learnable, and can be generated according to image content by another network.

With this knowledge, I adapt the standard resnet by replacing the first convolution in each residual block with a deformable one. Note that the parameters do not change, only the operation that is performed. Vitally, this allows the same pretrained weights to be used as those of the original resnet. Pretrained weights are very important for the performance of the model, as demonstrated in appendix A.1. Keeping them is therefore a key point. The offsets themselves are computed by a smaller secondary network, named OffsetNet, based only on the DEM, as shown in Figure 3.4. Offsets are calculated for each spatial resolution of features, with the idea that the kernel transformation would remain

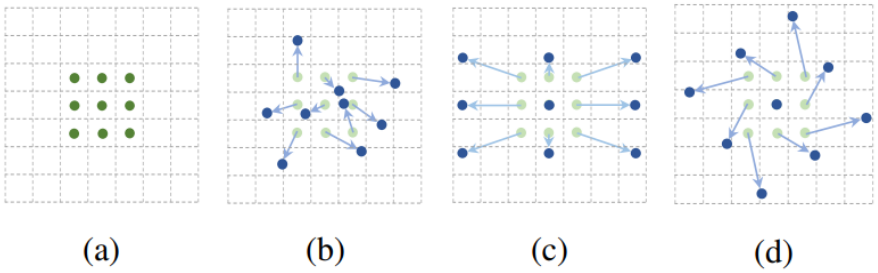


Figure 3.3: Deformable Convolution as described in [5]. The standard 3x3 kernel (a) can be deformed according to 2D offsets of each kernel element. Three examples are shown in (b), (c) and (d).

the same if only dependant on the DEM. A residual block is added when the spatial resolution is downsampled along with a 3x3 convolution, such that the kernel can also be changed to match the new size.

Improved decoder

The decoder of Deeplabv3+, while using some low-level features from the backbone, mainly uses the rich but low resolution high-level features. Not only does this make it hard to recover smaller details and objects, but good features from the first resnet layers need to be propagated through the entire network. By using features from all layers of the backbone, good features from earlier layers can be retrieved at higher resolution and don't have to be propagated through all layers, taking up channels that could be used for new features.

To aggregate features from all backbone layers and increase the receptive field, I use a module similar to the ASPP of Deeplabv3+ which I call DSPF (Deformable Spatial Pyramid Flow). The DSPF shown in Figure 3.5 performs convolutions at different dilation rates but additionally makes these deformable. Offsets are again determined by a network using the DEM only, as is done in the deformable backbone. Additionally, a flow layer described below is added to each module.

Instead of only using this on the final features from the backbone, the DSPF is then used on outputs from all backbone layers as shown in Figure 3.6. Features are subsequently upsampled to match in resolution and combined with another residual block. The final prediction is simply upsampled by a factor of four to match the input resolution.

Flow layer

A key challenge with FCNs is feature propagation. In the standard convolution, information is only propagated to neighbouring pixels requiring many convolutional layers and downsampling operations to achieve a large receptive field. While dilated and deformable convolutions help increase the receptive field and allow information to propagate further, the DEM provides another way.

The spatial gradient of the DEM can be interpreted as a vector field corresponding the downhill direction at every point. In other words, it describes the

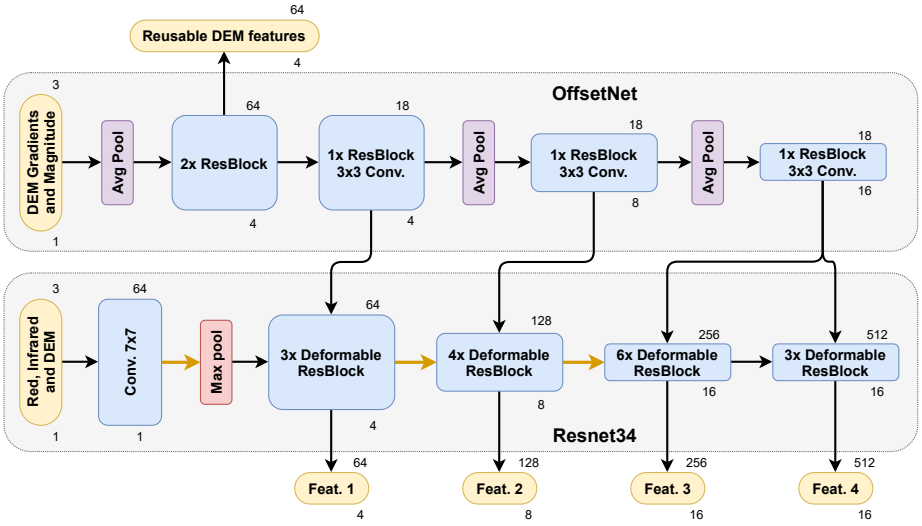


Figure 3.4: The deformable backbone. An adapted version of the resnet34 is shown whereby a secondary network named OffsetNet computes the offsets for each backbone layer. The first convolution in every residual block of the resnet has been replaced with a deformable one. Features of the resnet are output at each layer to be used in the decoder, as will be shown in Figure 3.6. The general DEM features are also output to be reused in the decoder. A legend of the diagram elements can be found in Figure 3.8.

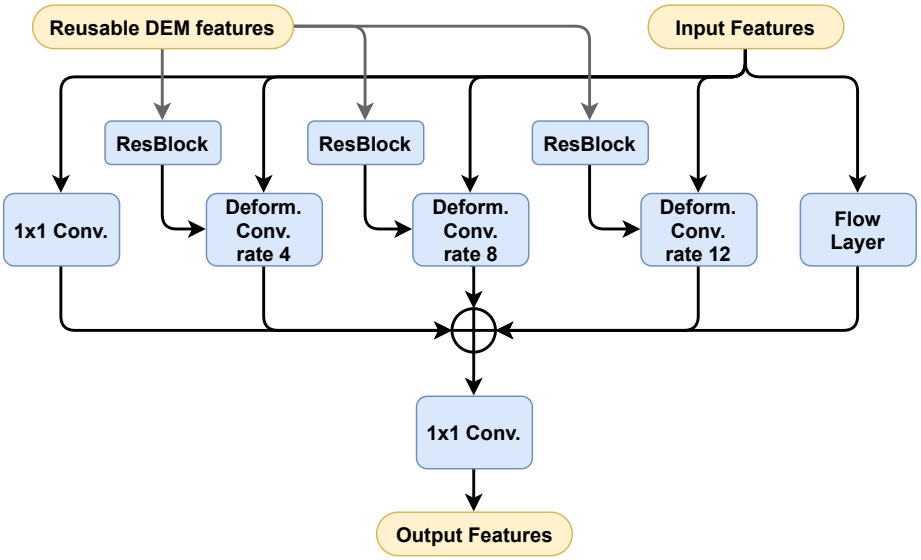


Figure 3.5: The DSPF module. Multiple deformable convolutions are initialised with different dilations. An additional flow layer expanded upon in Figure 3.7 is also included. A legend of the diagram elements can be found in Figure 3.8.

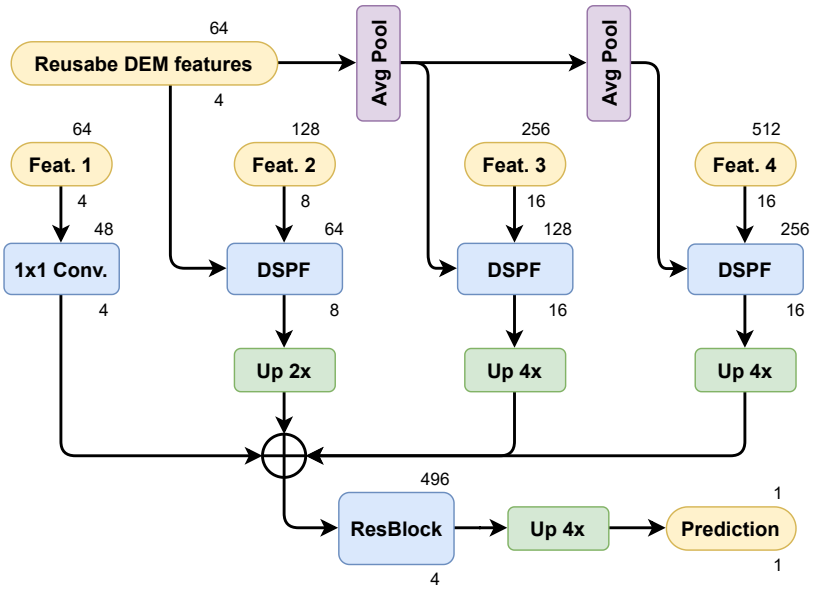


Figure 3.6: The AvaNet decoder takes the outputs of the deformable backbone and features from the DEM from Figure 3.4. The DSPF module is expanded in Figure 3.5. A legend of the diagram elements can be found in Figure 3.8.

direction in which an avalanche would flow. Although there are some exceptions to this, such as when an avalanche can flow uphill due to its momentum, this is true in most cases. Therefore, the DEM gradients can be exploited to propagate information along the gradient field similarly to how the avalanche would flow itself. The effect can also be reversed, propagating information back uphill.

To implement a module that performs such feature propagation, the 'grid_sample' function from pytorch comes in handy. It takes a flow field in addition to some features and resamples them according to the field. Usually used in combination with optical flow, the displacement field can be directly calculated from the DEM, in this case. The function itself is however not enough, but needs to be integrated into a module.

I therefore design a iterative module, which I call a flow layer, that performs the grid_sample function multiple times, as shown in Figure 3.7. Firstly, inputs are transformed to an appropriate representation with a 1x1 convolution and passed through a ReLU nonlinearity to achieve an effect similar to the sigmoid function, while avoiding vanishing gradients. The module then iteratively applies the sampling layer, propagating features along the gradient direction. The feedback arrow in the diagram shows that it is the transformed features that are fed to to the grid_sample operation. To be able to stop the flow, when reaching a mountain ridge for example, the features can be set to zero by applying an attention mask. This attention mask is computed from the input features at the beginning of the layer and remains constant through all iterations. With each iteration, the propagated features are added to a hidden state with a 1x1 convolution, similarly to how recurrent neural networks work. After a predefined

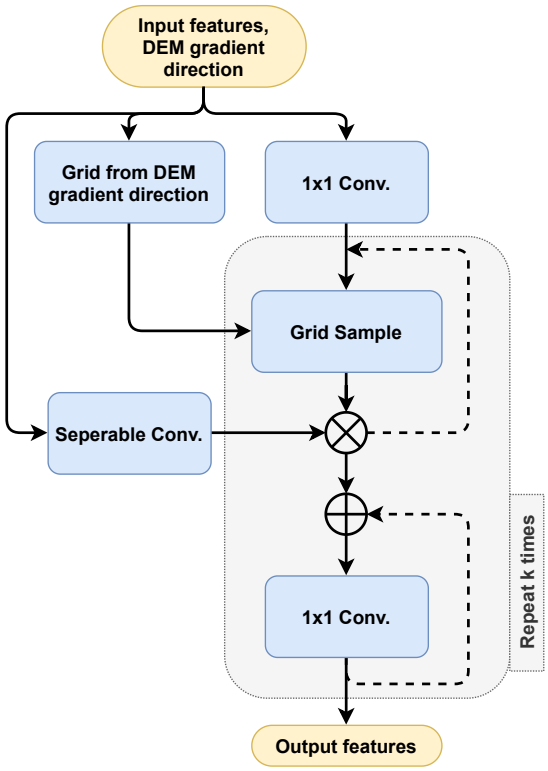


Figure 3.7: The flow layer. Performs iterative grid_sample operations, with feedback loops. A legend of the diagram elements can be found in Figure 3.8.

number of iterations, the the hidden state is output.

3.4 Use of dataset

Deep learning methods are heavily dependent on the data they are trained with. Consequently, the way in which the dataset is used for training plays a significant role in the models performance. The following sections will explain how samples are chosen, the train, validation and testing split, extra maps used, as well as the challenges of the dataset.

One decision made, was to only use red and near-infrared channels from the BGR-NIR images. Ignoring the blue and green channels is justified as the snowy mountains are almost black and white, and contain no useful color information, making them redundant. By disregarding them, the network can be kept smaller and faster. The effect of this decision is examined in section 4.4.

3.4.1 Choice of samples

In segmentation problems involving standard images, each image typically represents one sample. With satellite images, the choice of samples is more complex. Not only are satellite images too large to be processed in one go due to mem-

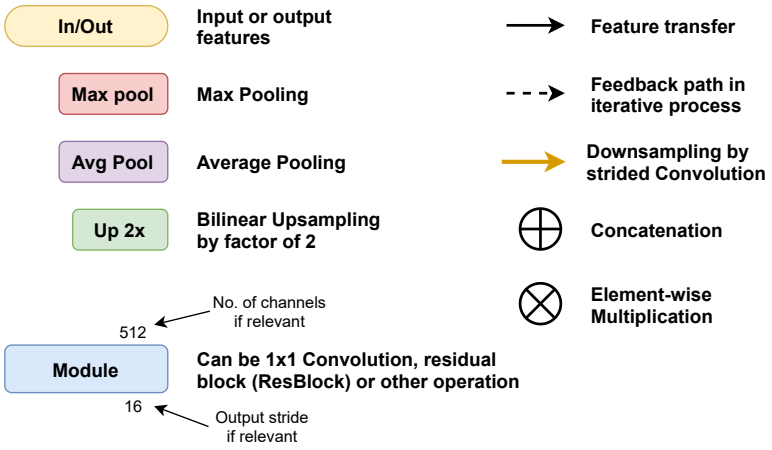


Figure 3.8: Legend of elements used in the architectural diagrams.

ory limitations, but this may also be a suboptimal use of the data. Samples are therefore created by extracting manageable size patches (more information on this in subsection 3.5.4) from the satellite images. This not only solves the memory problem but also allows much flexibility in the way samples are chosen.

Although the SLF dataset contains many avalanches, the area covered by them is still very small with respect to the whole satellite images. This data imbalance needs to be handled very carefully during training, if its associated problems are to be avoided. I therefore decide to focus training mainly on areas where there are avalanches present. This way the number of pixels labelled as avalanche can be kept more in balance with those that are not. In addition, it saves much computation by ignoring less relevant areas where there are no avalanches, reducing the training times required.

To help distribute samples where there are avalanches, I define a sample-point as the center of the location around which the patch will be extracted for that sample. The choice of these sample-points therefore determines the distribution of samples. The simplest solution is to have one sample-point per avalanche, such that each avalanche is more or less covered and in the center of the patch, ensuring there is enough context. Such a choice has two problems however. Firstly, areas where there are many small avalanches that fit in one patch will appear multiple times. Secondly, avalanches that are larger than the patch size will only be included in training partly.

To avoid these problems, I decided to choose sample-points for each avalanche only if no neighbouring sample would overlap. Additionally, I recursively place sample-points in each avalanche until all of its area is covered by samples. The resulting sample distribution over the satellite image then covers all avalanche pixels without overlapping samples. In a last step, I add add a small number of sample-points where there are no avalanches using Monte Carlo sampling. The number is chosen as 5 % of the number of samples with avalanches. This ensures that the data remains roughly balanced while still including examples from other areas than those were avalanches typically occur. A visualisation of the resulting sample-points can be seen in Figure 3.9. The resulting dataset has

Figure 3.9: Distribution of sample-points. Blue shows the binary mask of the avalanche labels. The red crosses represent the sample-points around which samples will be extracted. Most avalanches are covered by one sample-point while bigger avalanches have multiple. In addition, 5 % of samples are chosen where there are no avalanches.

∼20 % of pixels covering avalanches.

Deployment

For the deployment of the model, where predictions are to made across a complete satellite image, a shapefile describing the regions to be predicted is needed. A regular grid of sample-points is then generated within the region, and predictions are made one tile at a time. Samples are chosen to be overlapping by 100 pixels or 150 m, and are cropped after prediction to reduce artifacts near the edges, where there is less context.

3.4.2 Train, validation and test areas

As in any machine learning task, the dataset needs to be split into train, validation and test set. Both the validation and test set are never seen during training, such that the generalisation capability of the model can be evaluated properly. The validation set is specifically used for monitoring the training progress, avoiding overfitting, hyperparameter tuning and comparing model performance. The test set on the other hand, is only used for the evaluation at the end, when no more changes are made. This ensures a fair evaluation.

Since I am working with satellite images, I decided to make this separation in terms of geographical areas. This would allow me to better test the generalisation capabilities of my model, as it is tested on unseen topography. As I have two images available, I choose train, validation and test areas in each one,

allowing me to use them individually and capturing as much variance in the data as possible. Since topography and snow conditions vary geographically, two validation and test areas are used in each satellite image, in a compromise to test generalisability without overfitting to one type of topography or conditions. The areas are shown in Figure 3.10. The training region contains 80 % of the avalanches, validation 10 % and testing 10 %, in each year.

Part of the validation area is chosen around Davos in both images. The reason for this is the availability of extra data from the SLF with regard to the correctness of the dataset labels discussed in subsection 1.2.3. The other regions are chosen to contrast the area of Davos, containing low altitude areas with many small avalanches in 2019 and high altitude glacial areas with many avalanches in close proximity in 2018.

The test areas were importantly chosen such they do not contain any terrain covered in the other year. Therefore all of the test set presents new topography, so the model cannot simply recognise mountains or typical avalanche slopes it may remember from the training data. There are four test regions as shown in Figure 3.10, referred to as:

- Val d'Entremont 2018: Contains many avalanches reaching deep into the valleys through forested areas. Avalanches are often poorly visible with their label quality marked as 'created' (see subsection 1.2.2).
- Furkapass 2018: High avalanche density with lots of large avalanches. Many labels are marked as having 'exact' outlines.
- Safiental 2019: Avalanches are relatively sparse and often of smaller size.
- Engadin 2019: Satellite images captured at a relatively high inclination angle of 23°. There is more distortion in the orthorectified images than typical.

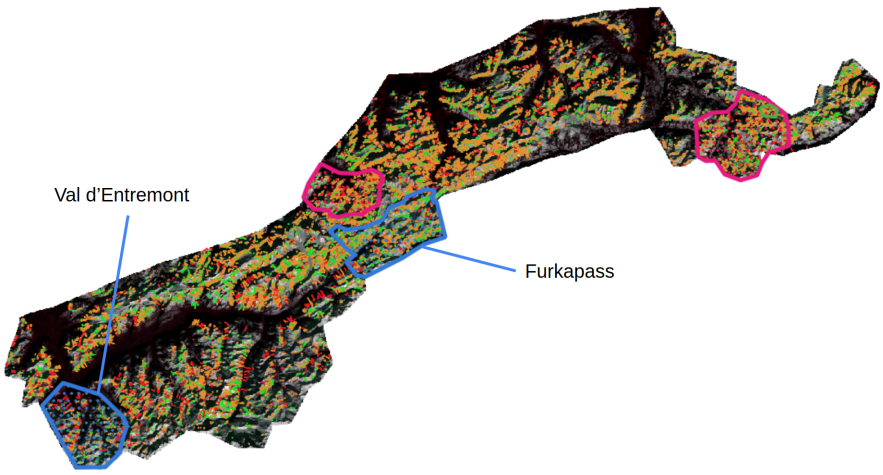
3.4.3 Additional maps

Considering that visual features are often limited when it comes to avalanche mapping, a good understanding of how an avalanche works can often help fill in the gaps. When manually mapping avalanches a topographic map was used in addition to the satellite images. This can help judge whether terrain is steep enough for an avalanche to start, how the avalanche would flow and where it would likely stop.

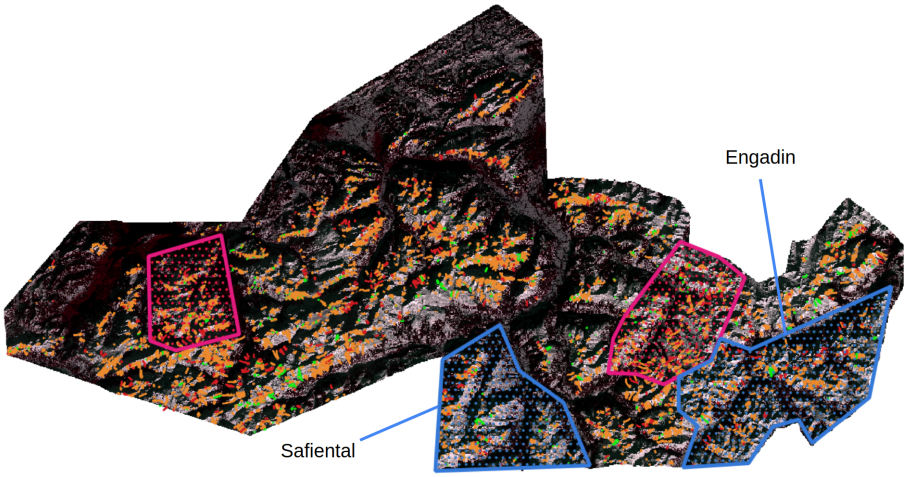
For my model to be able to learn these things, it must be given this information. I therefore include the DEM of Switzerland, shown in Figure 3.11, as an extra channel to the model. The DEM is an image with the height encoded at every pixel. Terrain steepness and gradient direction, which are important features for avalanche mapping, can be derived from it. Having a resolution of 2 m, this was upsampled bilinearly to match the 1.5 m resolution of the satellite images.

3.4.4 Challenges

There are some challenges when it comes to learning from this dataset. Not only are the differences between avalanche and no avalanche often very small, but



(a) 2018 [SPOT6 ©Airbus DS2018]



(b) 2019 [SPOT6 ©Airbus DS2019]

Figure 3.10: SPOT-6 optical images with the avalanche outlines and regions overlayed. Avalanches appear in three colors corresponding to the quality of the avalanche outline; Green: exact, Orange: estimated, Red: created. Validation areas are shown in pink. Test regions are blue and labelled with the names they are referred to.

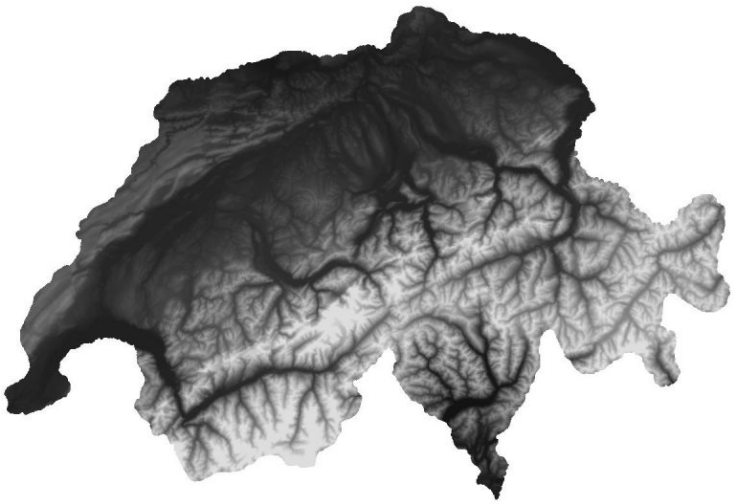


Figure 3.11: DEM of Switzerland. Brighter values are higher.

the mapping itself is not well defined. There are several factors that run across a continuous spectrum but need a line to be drawn at some point. For instance, avalanches of different age exist and generally become less visible when they are older. Somewhere a decision needs to be made on whether an avalanche is too old or too faded to be mapped. A similar problem exists when looking at how big a snow slide must be to count as an avalanche. Smaller slides or rolling snowballs are typically not classified as avalanche, but again, there is a continuous spectrum.

Consequently, avalanche mapping can be quite a subjective process and there is some uncertainty in the mapped labels. This can be challenging to learn from since the dataset can contain contradicting examples. In addition, some avalanches might have been missed completely further leading to inconsistent labels. Overall, the dataset should however be large enough that a reasonable average of these inconsistencies can be learnt.

3.5 Training

The network was trained with a weighted binary cross entropy (BCE) loss and the Adam optimizer. The learning rate was initialised to 1×10^{-4} and reduced by a factor of 4 after 10 epochs. An effective batch size of 16 was used with patches of 512×512 pixels. The loss function and data augmentation used are discussed more in depth in subsections 3.5.1 and 3.5.3, respectively. A summary of the most important parameters is given in Table 3.1.

Although there is plenty of research suggesting SGD generalises better than Adam [13], I could not get SGD to converge to any reasonable solution despite heavy testing. Therefore, I decided to stick with the Adam optimizer which is used with default momentum. Empirically, I found that weight decay hindered the optimisation problem from converging, and since I did not have any trouble with overfitting, weight decay was set to zero.

Parameter	Value
Loss function	weighted BCE
Optimizer	Adam
Initial learning rate	1×10^{-4}
Effective batch size	16
Patch size	512×512
Max epochs	18

Table 3.1: Summary of training parameters

3.5.1 Loss function

For training, I used a weighted BCE loss. Since some avalanche labels are of better quality than others, it makes sense to weight labels that are exact more than those that are uncertain. I decided to use following weights for each category of label:

- Exact: 2
- Estimated: 1
- Created: 0.5
- Background: 1

These weights were chosen such that the background has the same weight as estimated labels which are the most common type of avalanche label. Exact avalanches have double the weight and created ones which are very uncertain are given even less weight than the background.

In addition to the weighting according to the label, the loss weighting was decreased near the edges of the image patch. Pixels near the patch edges are missing much context which is very important for avalanche detection as described in subsection 3.5.4. As the edges with missing context make up a considerable area of the patch, this can lead the model to overly focus on local features such as surface roughness. By reducing the weight of the loss towards the edges, it should help the model learn to understand the bigger picture. I therefore start reducing the loss weighting linearly 100 pixels from the edge, down to 10 % at the edge. Note that it is not set to zero as this leads to random predictions on the edges which is not desirable.

Other loss functions were also tested for comparison. The focal loss [17], which has been used in instance segmentation to handle the class imbalance problem, was tested for example. This did however, not manage to converge to the same performance. Off course, an unweighted version of the BCE loss was also tested, but, while performing similarly to the weighted BCE, it did not show any gain over the weighted version.

Deep supervision

Deep supervision refers to the technique of calculating the loss over intermediate layers, in addition to the final prediction. With the more complex decoder of AvaNet introduced in subsection 3.3.2, I decided to use this to help the model

converge. The decoder has multiple paths from each layer of the backbone. To guide the network to predict avalanches from each backbone layer, before combining them into the final prediction, features are additionally fed through a 1x1 convolution and upsampled to the original image resolution. The loss is then calculated by additionally computing it for each of the intermediate predictions and adding them with one quarter of the weight of the final prediction.

3.5.2 Preprocessing

The only form of preprocessing done on the input images is data redistribution. Most importantly, this refers to standardisation of input data. This simply ensures that the inputs have a zero mean distribution with a variance of one, with respect to the statistics of the whole dataset.

In addition, I checked the input data distributions to ensure that these were more or less evenly distributed. Extremely skewed distributions can make learning harder and convergence worse. While I initially thought that my data was well distributed, it later turned out I had a bug in this calculation. Upon reexamining the input distribution it turned out I had a bimodal distribution with one particularly sharp peak corresponding to the shadows, as shown in Figure 3.12a. Since the rest was more evenly distributed I decided to flatten this peak by redistributing only negative inputs (after standardisation). The new negative values are calculated as follows while the positive values remain the same:

$$x = -3 * x^2 \tag{3.1}$$

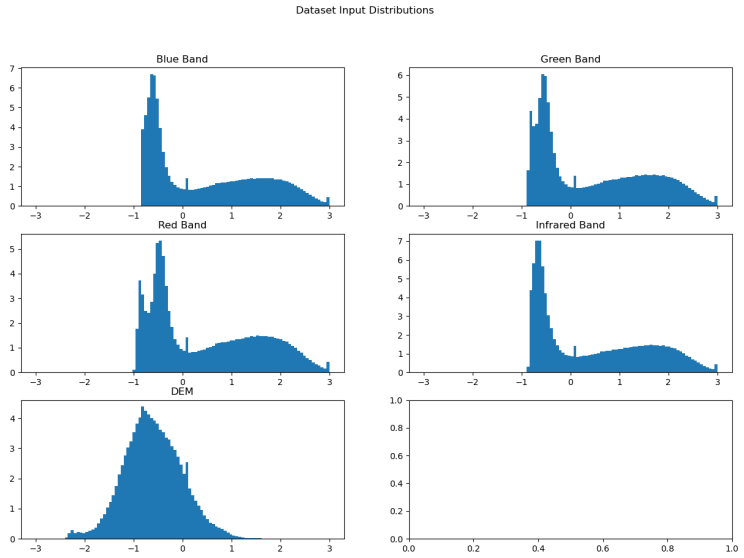
This spreads the values while affecting smaller values more. This way there is a smooth transition from the negative values that are redistributed, to the positive values that remain the same. The resulting distribution is more evenly spread as highlighted by Figure 3.12b. Note that the peak at zero is from 'no data' pixels in the image raster.

3.5.3 Data augmentation

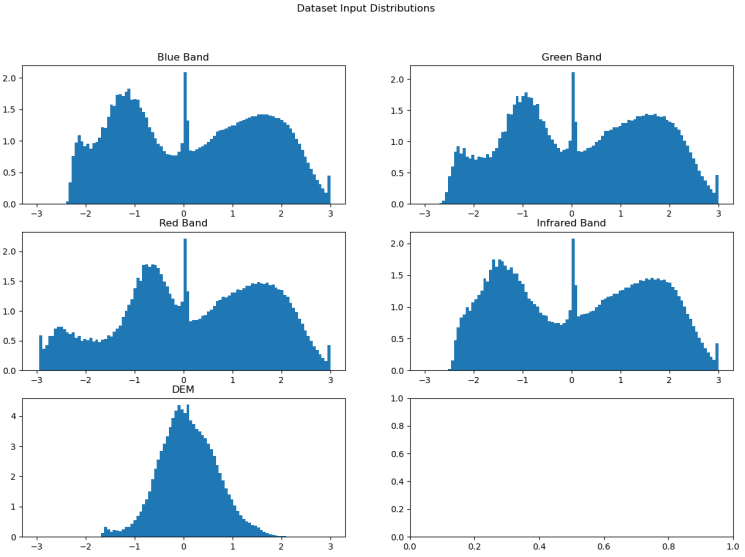
Data augmentation is a necessary part of training with the avalanche dataset. Since satellite images can be interpreted as a single image taken at a specific time, the snow and weather conditions are very similar across the entire image. Although the images cover a considerable geographic area, which means there will be some variation across it, the variance between images taken on different dates will be far greater. In this light, my dataset only contains two different conditions: those from January 2018 and those from January 2019.

For an automatic mapping algorithm to be of any use, it must work for new images with different weather and snow conditions as well. I therefore apply heavy data augmentation during training to compensate this lack of different conditions and teach robustness against such variations. The following will examine the data augmentation techniques used and the conditions they should help generalise against.

Rotation and flipping – These were completely randomised with a uniform distribution spanning the whole 360°. This not only helps generalise with respect



(a) Distribution when only standardising inputs. A large spike is visible corresponding to shadow areas.



(b) Distribution after applying Equation 3.1 to negative values. The resulting distributions are more evenly spread.

Figure 3.12: Data distributions of the standardised inputs averaged over the dataset. Each input band is shown separately with the y axis representing the percentage of values in each histogram bin. The central peak corresponds to the no data value of the satellite images.

to different avalanche shapes but more importantly against images taken at a different time of day or year which would lead to a different lighting angle. While a bias depending on terrain aspect might be meaningful for avalanche mapping in a single image, it may not make sense with new images. Avalanche mapping in April might for example detect most avalanches on southern faces that are heated by the sun, while an example from January with less solar irradiation may show a more even distribution across all aspects. By completely randomising the rotation it forces the network to focus on features that indicate avalanches independent of lighting and terrain aspect.

Mean shifting and variance scaling – Randomly shifting the mean and scaling the variance of the input image is important for generalising to different times of day, year, and even atmospheric conditions. These can lead to different levels of brightness and contrast in the images. By performing the augmentation it should become more robust to such changes instead of overfitting to specific absolute values.

Patch shifting – Patches are usually extracted with the corresponding sample point at its center as described in section 3.4. Patch shifting means that patches are extracted slightly offset from the sample point. This way the model becomes robust to position shifts, and importantly, to avalanches that are only partly visible as they go off the edge of the patch. This is a likely scenario when deploying the algorithm and therefore needs to be trained for.

The augmentation techniques mentioned above were all with regard to the satellite images. For the DEM, variance scaling is skipped since the values directly correspond to the elevation, and their gradient to the slope magnitude. This information and its distribution does not change with new images but has a physical meaning and should therefore not be manipulated. I did however perform some mean shifting on the DEM independent of that of the satellite images. This can reach a maximum of 1000 m and reflects the fact that avalanches may occur at lower altitudes in winters where the snow line is lower.

3.5.4 Patch size

The patch size was chosen as large as possible. Patch sizes of 128×128 pixels and smaller did not yield good results. At least 256×256 pixels are needed to make meaningful predictions. This can be explained intuitively since many avalanches are only partly visible requiring some form of interpolation or even extrapolation to predict the whole avalanche. It is therefore important to have as much of the avalanche and its context in the patch. In [2] it is shown that only 31 % of avalanches are smaller than $10\,000\,\mathrm{m}^2$. 53 % are between $10\,000\,\mathrm{m}^2$–$80\,000\,\mathrm{m}^2$ and the remaining 16 % larger with some reaching above $500\,000\,\mathrm{m}^2$. Assuming square shaped avalanches, this puts the length of 53 % below 282 m and the largest avalanches at ∼700 m. Of course avalanches aren't square, but it would be a reasonable assumption that the majority of avalanches are shorter than ∼300 m and only a few reach beyond 1 km. Since the images are at a resolution of 1.5 m this corresponds to 200 pixels and 667 pixels, respectively. This matches with the minimum patch size of 256×256 mentioned before, which can cover many avalanches completely.

While initially limited to 256×256 pixels due to memory limitations of the

GPU, I was able to increase this to 512×512 after tuning hyperparameters such that the model was converging with smaller backbones. Note that this corresponds to $768\,\mathrm{m} \times 768\,\mathrm{m}$ which should allow most avalanches to be contained in a patch completely. Due to extracting patches from a large satellite image, it can however still happen that only part of an avalanche is visible. The likelihood of this can be reduced by further increasing the patch size but it should be noted that this will likely also happen during testing when the avalanche locations are unknown. It is therefore important to also train with such examples.

3.5.5 Memory issues and limitations

The training process is quite resource intensive. Consequently, I experienced various issues, including limited batch and patch sizes, memory overflows and read bottlenecks.

Both larger patch sizes and larger batch sizes could potentially further improve the models performance. These are however limited by the GPU memory available. The minimum batch size per GPU such that batch norm layers still work is 2. With this constraint, the maximum patch size I could train with was 512×512 pixels. Above this the 11 GB VRAM limit was exceeded. The patch size could therefore only be increased further by reducing the model size.

Since a batch size of 2 is too small for proper convergence during training, I trained on up to 4 GPUs at a time. Additionally, I accumulated gradients over two iterations before updating weights. This way I reach an effective batch size of 16 while still using large 512×512 pixel patch sizes.

Another problem I encountered was that training was slow due to a memory access bottleneck. When training, patches have to be extracted from various locations in the satellite image. To avoid having to read the whole satellite image, these are saved in a tiled format such that only the relevant areas have to be loaded into memory. This was however still a bottleneck as reading was taking longer than the computation. To reduce the load on memory access I used batch augmentation as introduced in [11]. This works by using the same sample in a batch multiple times with different augmentations. This way only half the number of read operations are needed, and in fact, this has also been shown to improve convergence and generalisation [12].

Finally, I had the problem that GDAL was using too much RAM. I had to reserve over 200 GB RAM when running on the Leonhard computing cluster. Consequently, I could not run many tests in parallel and had longer queuing times. It turned out this was because of using a VRT raster. Although the maximum allowed RAM is set to 5 % by default, this was found to also scale with the number of process, number of rasters in the VRT, and other factors as I described in [8]. Having discovered that RAM usage scales with these factors, I could set the limit accordingly low which solved the problem.

Experiments and discussion

Models are evaluated on the test regions described in subsection 3.4.2, which were specifically reserved for this. Unless otherwise stated, metrics are evaluated across all regions and dataset samples are chosen the same way as for training (see subsection 3.4.1). Deeplabv3+ is taken as a baseline and my own model AvaNet is compared to this. Furthermore, although not all avalanche labels are of the same quality as discussed in subsection 1.2.2, I decided to evaluate with respect to all mapped avalanches regardless of there quality and compare their average accuracy instead. Unfortunately, the related methods from chapter 2 cannot be compared on the same data directly as no code is available.

The metrics used for the evaluation will be discussed first before presenting both quantitative and qualitative results. A test across a complete region is also examined in subsection 4.2.3. Furthermore, the generalisation ability of the models is examined and ablation study is performed.

4.1 Evaluation metrics

Two types of metrics are used to evaluate model performance. On the one hand, metrics that work on a pixel-wise level, and on the other, ones that work on a per avalanche basis.

4.1.1 Pixel-wise metrics

Pixel-wise metrics are those that are evaluated at the pixel level and make no distinction between different avalanche instances. Every pixel is treated independently, and consequently, the same score can be achieved by only few avalanches being detected very precisely, or by many avalanches that are partially detected. This ambiguity is tackled by the object-based metrics explained in the next subsection 4.1.2.

Three pixel-wise measures are used: precision, recall and F1 score. The F1 score is useful for comparisons, as it summarises the other two metrics with the harmonic mean. All metrics require a hard prediction for each pixel: avalanche or not. This is obtained by thresholding the predicted probabilities, such that all values above 0.5 count as avalanche and those below as no avalanche (or background as I also call it).

The metrics can be computed from the entries of the confusion matrix which are as follows:

- TP: true positives
- TN: true negatives
- FP: false positives
- FN: false negatives

In this case, the positive labels are avalanches and negative labels background. The metrics can be computer as:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\,Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

What these three metrics do not take into account, is the number of true negatives. Since avalanches are rather sparse compared to the whole satellite images, true negatives are also important to consider. I therefore also evaluate the same metrics but with respect to the background class, i.e. the background is taken as the positive label while the avalanche is the negative label. However, unless otherwise stated, the metrics are with respect to the avalanches.

4.1.2 Object-based metrics

In addition to the pixel-wise measures, I evaluate the accuracy of predictions on an object level. The accuracy is simply defined as the percentage of avalanches detected. To determine whether an avalanche is detected I set a threshold on the percentage of the area that must be classified as avalanche. Two thresholds are used: 50 % and 80 %. These are referred to as "Acc. 50%" and "Acc. 80%" in the tables below. An accuracy of 60 % at the 80 % threshold therefore means that 60 % of avalanches had more than 80 % of their area correctly predicted.

4.2 Results

Results are examined in three parts. Firstly, quantitative results are analysed with respect to the metrics discussed in section 4.1. In the second part, the results are evaluated qualitatively with examples to give a better understanding visually. Finally, predictions across an entire region, rather than just the dataset samples, are examined in subsection 4.2.3.

Models were trained on the complete dataset, including 2018 and 2019, with the procedure described in section 3.5. Both Deeplabv3+ and AvaNet are evaluated with the resnet34 version of the backbone. For a comparison on different backbones see section 4.4.

Model	Precision	Recall	F1 Score	Acc. 50%	Acc. 80%
Deeplabv3+	**63.7**	62.8	61.7	66.3	45.9
AvaNet	62.9	**66.9**	**63.1**	**70.8**	**50.3**

Table 4.1: Precision, recall and F1 score are with respect to the avalanches when evaluated on the test regions. All values are given as percentages.

Model	Precision	Recall	F1 Score
Deeplabv3+	90.1	90.6	89.6
AvaNet	91.0	89.2	89.3

Table 4.2: Precision, recall and F1 score are with respect to the background class, when evaluated on the test regions. All values are given as percentages.

4.2.1 Quantitative

Table 4.1 summarises the results on the test set. It can be seen that AvaNet performs only slightly better than the Deeplabv3+ baseline. An F1 score of ∼63.1 % is achieved on the avalanches by AvaNet, and 61.7 % by Deeplabv3+. Note this is comparable, although slightly under, the F1 score of 66 % achieved by [1] on SAR as mentioned in section 2.2.

Precision and recall are relatively balanced, with 62.9 % and 66.9 % respectively, for AvaNet. While the precision is slightly higher for Deeplabv3+, AvaNet has a significantly higher recall. Avalanches are detected with an accuracy of 70.8 % at a 50.3 % threshold and 50.3 % at a 80 % threshold for AvaNet. Considering the labels taken as ground truth only have a POD 74 % themselves [9] this is a reasonably good result.

The metrics are further evaluated with respect to the background class, to take into account true negatives as well. Since the dataset is imbalanced with avalanche pixels making up ∼20 % of all pixels, this therefore shows significantly better results. Table 4.2 shows an F1 score of almost 90 % is reached on the background class for both models.

Having evaluated metrics on all labels regardless of their quality, I examine the difference in accuracy per label quality. Statistics when requiring a minimum detection area of 70% are shown in Table 4.3. As expected, exact labels are detected best, with an accuracy of 72.1 % with AvaNet. Note while Deeplabv3+ has a lower accuracy, it still follows the same pattern and can be attributed to its lower recall but higher precision. The accuracy drops ∼15 % for estimated avalanches and a further ∼15 % for created avalanches. While the label quality is correlated with detection rates as expected, note that exact avalanches are still only detected to ∼70 % although they are classified as well visible. On the other hand, a relatively large percentage of created avalanches are detected, considering they are only partly visible.

Furthermore, Table 4.3 shows the average size of detected and undetected areas of avalanches visible in a patch. While the difference is not huge, it shows smaller areas are harder to detect. This was also observed in manual mappings from the SLF [2] and appears to remain the case for deep learning methods.

Model	Avg. area detected [m^2]	Avg. area undetected [m^2]	Acc. Q1 [%]	Acc. Q2 [%]	Acc. Q3 [%]
Deeplabv3+	33'747	25'036	68.7	52.0	38.2
AvaNet	33'557	24'390	72.1	56.7	43.4

Table 4.3: Detection statistics on the complete test set when an 70% coverage is required for an avalanche to count as detected. The average areas correspond to the area of an avalanche instance visible in a patch and may be smaller than the avalanche instance itself. Accuracy is compared for the different quality labels from subsection 1.2.2 – Q1: exact, Q2: estimated and Q3: created.

Region	Precision	Recall	F1 Score	Acc. 50%	Acc. 80%
Val d'Entremont 2018	49.6	72.0	57.3	76.0	58.8
Furkapass 2018	68.3	68.6	67.3	73.4	53.5
Safiental 2019	59.7	60.3	61.0	59.2	35.9
Engadin 2019	64.0	64.4	61.4	64.3	38.8

(a) AvaNet

Region	Precision	Recall	F1 Score	Acc. 50%	Acc. 80%
Val d'Entremont 2018	51.5	68.6	56.8	73.4	53.8
Furkapass 2018	69.7	64.0	66.6	68.2	48.7
Safiental 2019	61.7	55.6	58.7	53.7	32.8
Engadin 2019	63.0	60.5	59.3	61.0	35.7

(b) Deeplabv3+

Table 4.4: Model performance on the test set split by region for AvaNet and Deeplabv3+. Metrics are given as percentages and with respect to the avalanche label.

To pinpoint what is easy and what challenging for the model, the test set regions are further analysed individually for AvaNet. Table 4.4 shows the results. The Val d'Entremont shows the worst performance with an F1 score of 57.3 %. This is likely due to the large amount of avalanches that are poorly visible and often reach deep into the valley through forested areas, as mentioned in subsection 3.4.2. Since those avalanches often cover a large distance, they only appear on patches partly and are missing context. Furthermore, there are relatively few such examples in the training set to learn from. The Furkapass region on the other hand performs much better. This may be attributed to the better visibility and accordingly better label quality there.

Interestingly, while the areas from 2019 show similar or even better F1 scores and recall than those in 2018, the achieved accuracy is lower. Both the 50 % and 80 % accuracy levels are more than 10 % lower for both regions in 2019. Since the F1 score does not show the same drop, this indicates that a larger proportion of avalanches is detected to some extent, but the area covered per avalanche is generally smaller.

4.2.2 Qualitative

To get a better understanding of the performance and where the problems are, I examine some examples visually. Since Deeplabv3+ and AvaNet both make very similar predictions, most examples are shown for AvaNet only, while the last example compares them. The label colors correspond to those defined in subsection 1.2.2. Green labels are best visible, orange less so and red are only partly visible. The thresholded predictions represent a binary mask where red means an avalanche is predicted. On the right hand side of each figure the raw predictions are shown which represent the probability of an avalanche. Dark red values indicate it is highly likely that it is an avalanche, yellow is uncertain and fades away with a blue hue where there is definitely no avalanche.

A typical example where predictions are good is shown in Figure 4.1. Avalanches with different quality labels and sizes can be seen. The big exact avalanche in green is mapped very precisely. The smaller avalanches are mapped less precisely, but importantly, all avalanches are detected (ignoring those right on the edges). Figure 4.2 shows that even avalanches labelled as poorly visible indicated by the red label, can also be detected well in some cases. While definitely not true for all examples, as seen in subsection 4.2.1, it shows the model is not necessarily limited to avalanches that are clearly visible but can handle harder examples too.

Figure 4.3 is another example where predictions are relatively good but less precise than manual labels. When looking at the raw predictions, rather than the thresholded ones, the individual avalanches can be seen with a lot of detail representing the labels quite well. This highlights the important point, that a lot of information is lost when thresholding. As explained in section 4.1, all metrics are calculated with the thresholded predictions and therefore these details are not considered in the metrics. Another thing this example highlights, is fact that the avalanche mapping is a difficult task even for experts. Notice what might be an older large avalanche deposit in the bottom right of the image. In the labels, only a smaller newer avalanche that ran out on top of this has been manually labelled. In contrast, the model predicts the whole deposit as an avalanche.

Another interesting example with wrong predictions that may be caused by the subjectiveness of avalanche mapping, is shown in Figure 4.4. The input image clearly shows some features that indicate snow slides, snow balls or an avalanche. While the avalanches were included in the manual mapping in this case, it might be debatable whether to classify this as an avalanche or not. The fact that the model does not predict any avalanches, although the features are well visible indicates that such examples might typically not have been mapped in the dataset.

An example where predictions are quite bad is shown in Figure 4.5. The precision of the prediction is very poor, with large avalanches being predicted although the manual labels only show a few thin avalanches. The cause of this is not clear although it may be due to the snow texture.

Finally, an example comparing Deeplabv3+ and AvaNet is shown in Figure 4.6. The predictions are very similar for the labelled avalanche although AvaNet is

slightly more precise. This is however, mostly due to the threshold chosen and the predicted probabilities actually show a very similar pattern. The bottom left of the image is of more interest. The snow surface is more rough there and Deeplabv3+ is unsure if this is an avalanche or not. AvaNet on the other hand is certain that there is no avalanche there, indicating it has a better understanding.

More examples can be found in appendix A.4.

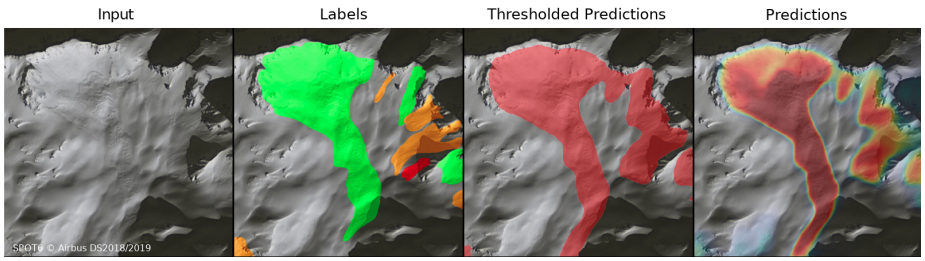


Figure 4.1: Predictions on well visible avalanches are accurate. Other avalanches are also detected although an accurate outline is not predicted on smaller avalanches.

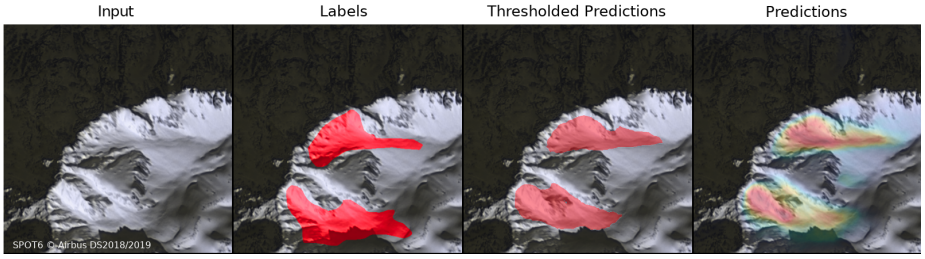


Figure 4.2: Even avalanches in the worst category of outline quality can be mapped my the model effectively in some cases.

4.2.3 Whole Region

While all previous results were evaluated on the dataset patches chosen as described in subsection 3.4.1 only, experiments were also done across whole regions. This is achieved by running the model across larger areas one patch at a time (see 'Deployment' in subsection 3.4.1). The resulting tiling effect can be seen in Figure 4.7, although it does not impact predictions much. It could be reduced by using a larger tile size.

Naturally, the recall remains the same when testing on a whole region, since all avalanche labels are already covered in the dataset. The precision does however drop as more false positives are predicted. For the Furkapass region the precision dropped to 50.9 %, while it dropped down as far as 22.3 % for Val d'Entremont region, found to be the most challenging in Table 4.4. Note the Val d'Entremont region only has a small area covered by avalanches. Consequently, the recall on the background class remains at over 95 % for both regions.

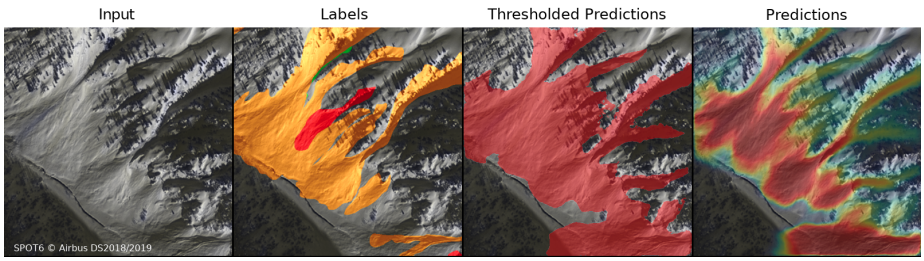


Figure 4.3: A lot of detail is in the raw predictions which represent the likelihood of an avalanche. Often individual avalanches can be recognised here, before thresholding in which much information is lost.

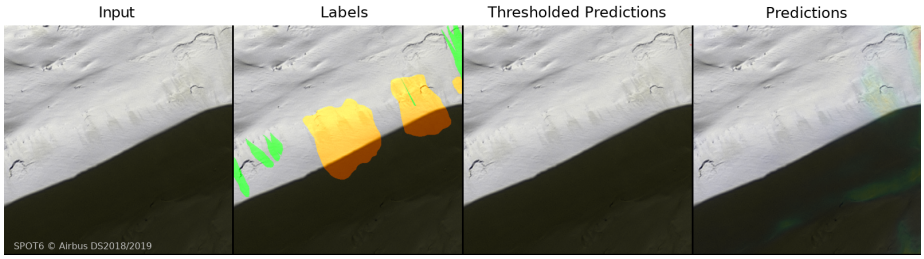


Figure 4.4: An example where it is debatable whether the clearly visible features should be classified as avalanches or not. No avalanche is predicted indicating that such examples are typically not labelled in the dataset.

Looking at the larger area around the town of Realp in the Furkapass region, shown in Figure 4.7, the models quality becomes more apparent. While it can be seen that much more area is predicted than covered by the avalanche labels, particularly in the shaded area, the predictions are mostly uncertain in those areas. Predictions with a high confidence, which appear in red in the figure, are mostly limited to the manual labels. Notice that the tiling effect in the predictions becomes most apparent in shaded regions, which is another indication that predictions there are uncertain. The effect can be reduced by having more overlap and larger patches.

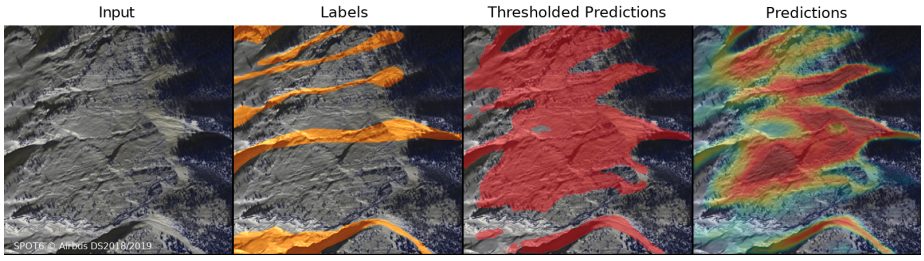


Figure 4.5: Predictions are quite poor, predicting large avalanches than labelled.

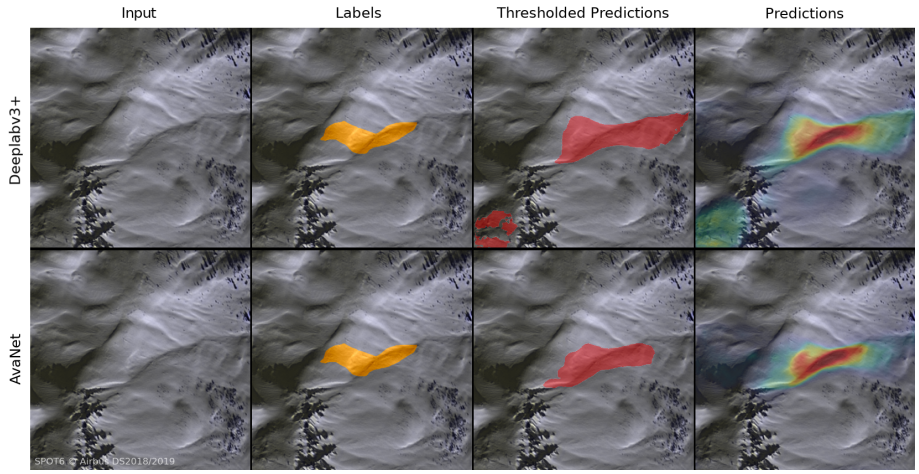


Figure 4.6: Comparison of predictions from Deeplabv3+ (top) and AvaNet (bottom). Predictions on the labelled avalanche are similar. Deeplabv3+ is however unsure about an area in the bottom left of the image where the surface is rougher.

4.3 Generalisation ability

Having only two satellite images available in the dataset provides a great challenge for deep learning models to be able to generalise to new images. In order to test this capability, I train the model only on data from 2018 or 2019 and then test it on images from the test set of the other year. To compare the performance against that when training on both images, I take the difference and average it over both years. The test is done for both Deeplabv3+ and AvaNet. Results are shown in Table 4.5.

The F1 score of both models is about 7 % lower when tested on images from the other year than they were trained on. Particularly, the recall appears to be affected. This means that avalanches are less likely to be detected while the number of false positives is not affected as much. It can be assumed that the drop in performance would be less than this when tested on new images, since it has then already been trained on two different images.

4.4 Ablation study

To analyse the effect of different design choices and architectures from section 3.3, experiments were performed with model variations trained and tested in the same way.

Firstly, the effect of the backbone is examined on Deeplabv3+ with resnet18, resnet34 and resnet50, see Table 4.6. Resnet50 was found to improve on resnet34 only marginally, increasing the F1 score by 0.4 %. It therefore wasn't considered further. The smaller resnet18 on the other hand has an F1 score 0.9 % lower than resnet34. This is interesting since resnet18 is considerably smaller, with only

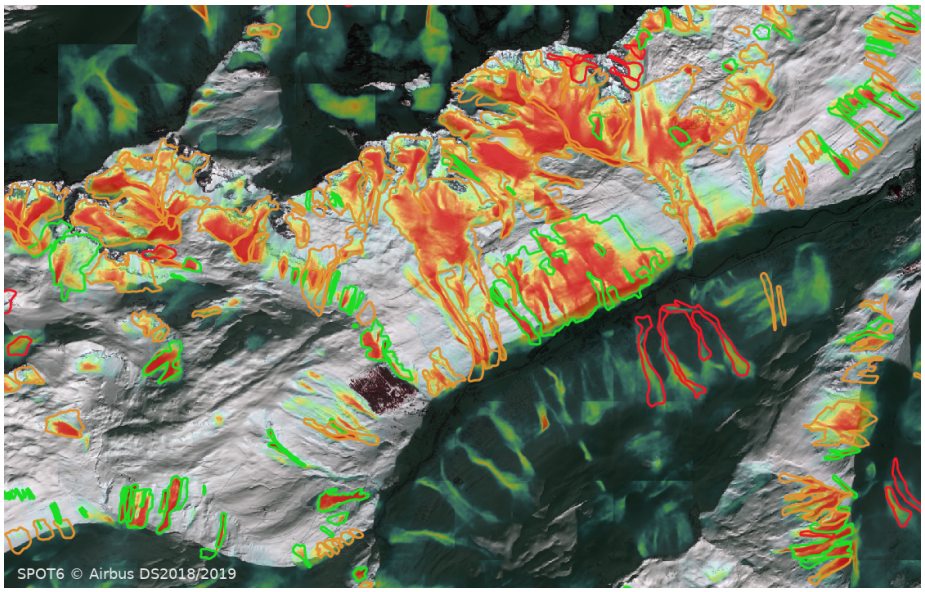


Figure 4.7: Larger patch 12 km across in the Furkapass region with the town Realp at the center. Manual labels are overlayed with the color code from subsection 1.2.2 with outlines only. AvaNet predictions are overlayed with a colormap running from red indicating high certainty, yellow unsure, to blue hues where background is more likely.

Model	Precision	Recall	F1 Score	Acc. 50%	Acc. 80%
Deeplabv3+	63.7	62.8	61.7	66.3	45.9
AvaNet	62.9	66.9	63.1	70.8	50.3
Deeplabv3+ difference	**1.0**	-13.2	-7.6	-15.8	-12.3
AvaNet difference	-3.5	**-8.8**	**-7.0**	**-11.6**	**-7.9**

Table 4.5: Generalisation ability. The top rows show the models when trained on both years as before. The bottom rows correspond to the difference when trained on data from one year but tested on the other, compared to performance when trained on both years. Difference are averaged for tests on 2018 and 2019. Negative value indicate the performance was worse when only trained on data from one year.

Model	Precision	Recall	F1 Score	Acc. 50%	Acc. 80%
Deeplabv3+ resnet18	63.4	61.7	60.8	65.9	43.6
Deeplabv3+ resnet34	63.7	62.8	61.7	66.3	45.9
Deeplabv3+ resnet50	**64.2**	63.7	62.1	67.9	47.4
AvaNet deform. resnet18	62.5	66.0	62.4	69.6	49.0
AvaNet deform. resnet34	62.9	**66.9**	**63.1**	**70.8**	**50.3**

Table 4.6: Models trained with different resnet backbones. For AvaNet, the corresponding deformable version is used as explained in subsection 3.3.2. Metrics are given as percentages and are with respect to the avalanches when evaluated on the test regions.

12 million parameters compared to the 22 million of resnet34. Never the less, it still performs reasonably well indicating that features are relatively simple for the most part and do not require huge networks. Table 4.6 further compares AvaNet. Notice that while the difference between the deformable resnet18 and resnet34 versions of its backbone are less than for Deeplabv3+, the AvaNet resnet18 version outperforms even the resnet50 version of Deeplabv3+. This shows that the architectural changes of AvaNet give it a better learning capacity that is not necessarily achievable through increasing the number of parameters alone.

More test cases regarding other aspects of the models are shown in Table 4.7 and compared to the standard versions of Deeplabv3+ and AvaNet. The first test was to check the performance of Deeplabv3+ without the DEM. Surprisingly, the model without the DEM actually outperforms the one with it. It is not clear why this is the case, but highlights the fact that the model does not learn to use the DEM well, when it is simply added as another channel. This justifies the AvaNet architecture which was created to solve exactly this problem in the first place. Due it integrating the DEM more explicitly it can however not be tested without it.

The decision from section 3.4 to disregard blue and green channels is analysed on AvaNet. A test with all channels shows a slight decrease in performance. While the decrease itself is negligible, it does confirm that the blue and green channels are redundant for avalanche mapping as was assumed in section 3.4.

Finally, the architectural changes of AvaNet are analysed by replacing parts with those from Deeplabv3+. The two models are however very similar in terms of performance and therefore the conclusions drawn are limited. Table 4.7 does however indicate that the deformable backbone is responsible for most of the improvement. When trained with the standard resnet34 backbone, AvaNets F1 score drops by 0.9 % and is closer to that of Deeplabv3+. Simply replacing the decoder with that of Deeplabv3+ appears to have less of an effect. Furthermore, the novel flow layers used in the decoder, which was explained in subsection 3.3.2, do not seem to add anything but rather inhibit learning. When trained without the flow layer the model actually performs slightly better.

Model	Precision	Recall	F1 Score
Deeplabv3+	63.7	62.8	61.7
Deeplabv3+ – without DEM	64.7	62.2	62.4
AvaNet	62.9	66.9	63.1
AvaNet – all BGR-NIR channels	64.0	65.9	62.8
AvaNet – resnet34 backbone	66.1	62.3	62.2
AvaNet – deeplabv3+ decoder	61.3	67.7	62.8
AvaNet – no flow layers	63.4	66.8	63.3

Table 4.7: Ablation study. Models are tested with different input and model variations. Metrics are given as percentages and are with respect to the avalanches when evaluated on the test regions.

4.4.1 Deformable convolution

Being responsible for a large part of the improvement of AvaNet over Deeplabv3+, the deformable backbone is analysed further. The only change made to the backbone, was to make the first convolution of each residual block deformable, with offsets computed by a smaller OffsetNet. Therefore, I examine the deformed kernel as determined by OffsetNet.

Since the offsets only depend on the DEM, I visualise them by showing how the convolutional kernel is deformed on top of an image of the DEM. Four examples from the last resnet layer are shown in Figure 4.8. Although hard to interpret, there are a few observations that can be made. Firstly, the deformation appears to be limited in size, reaching a maximum displacement of around one pixel in the examples. Furthermore, the deformations do not appear to represent any typical transformation such as a rotation. Although the deformation can appear somewhat random, kernel element tend to stretch along the DEM gradient direction to some extent. Also notice, that the center pixel of the top two images shows a similar gradient direction and accordingly, the kernels are deformed similarly. This hints that offsets and corresponding deformation are mainly determined by the terrain aspect.

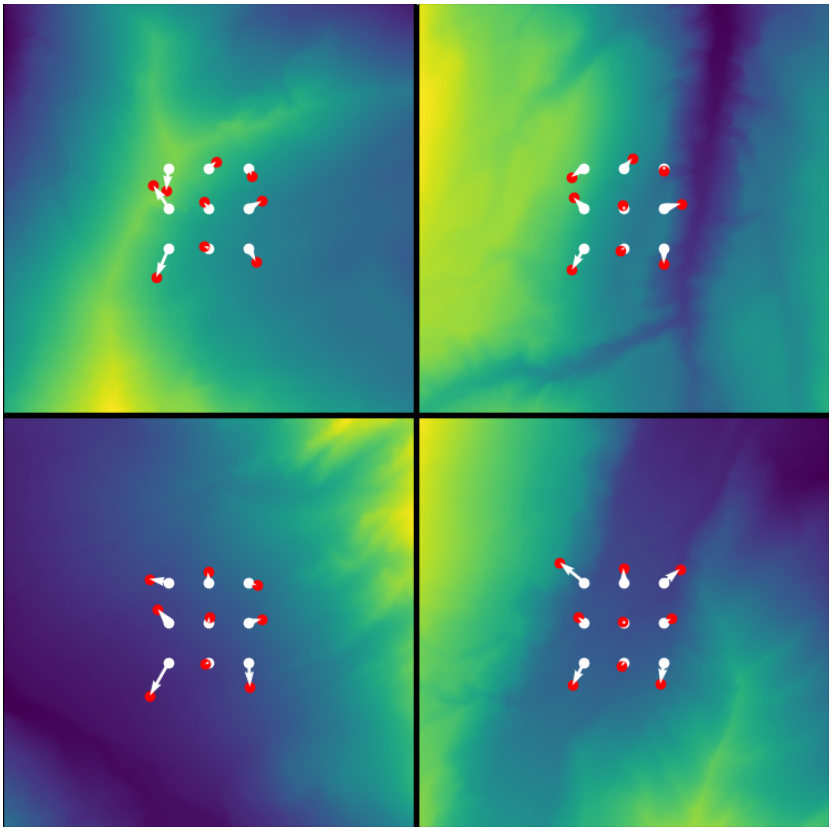


Figure 4.8: The deformed kernel is specific to the third resnet layer and is only shown for the center pixel of the patch. Its size has been exaggerated to be more easily interpretable but corresponds to a 3x3 kernel with a dilation of 1. The background image corresponds to the DEM with lighter yellow hues being higher. The red dots show the final position of kernel elements, as displaced from the standard shape by the offsets visualised with white arrows.

Conclusion and Outlook

5.1 Conclusion

This work tackled the task of automatic avalanche mapping from VHR optical satellite imagery. While related work has been successful using SAR images, optical images could provide higher resolution mappings including release zones and small avalanches. Existing work using optical images has however been more limited and still showed room for improvement. This thesis therefore attempted to bring the method using VHR optical satellite images up to speed with those using SAR images.

The task was determined to be a semantic segmentation problem which has widely been researched in the computer vision field. Successful deep learning methods from the domain could directly be applied to that of avalanche segmentation. Deeplabv3+ was taken as a baseline but appeared not to be learning the necessary features of the terrain from the DEM. The model was therefore extended to integrate the DEM more explicitly resulting in a model I call AvaNet. Its main feature consists of a deformable backbone that adapts the convolutional kernel to the terrain described by the DEM.

Both models were trained and tested on a dataset from the SLF, with 25 000 manually labelled avalanches in 1.5 m resolution optical SPOT-6 satellite imagery. An F1 score of 63 % is achieved on the test set by AvaNet which is comparable to the performance of related work using SAR images. While Deeplabv3+ performs slightly worse, results from the two architectures are still very similar. Both are able to map avalanches to a reasonably good degree considering the uncertainties in the manual labels they are trained on.

A thorough analysis into the results showed that predictions are best on sunny slopes and with larger avalanches. While most smaller avalanches are also detected, the precision of their mapped outline is considerably worse than for larger avalanches. The main challenges are however shadow areas and those were avalanches reach deep into the valleys through forested areas. These specific cases still need more attention.

Overall, it can however be concluded that VHR optical satellite images can be used for large scale mapping of avalanches. While some avalanches will remain undetectable due to the nature of the images, the majority of avalanche activity

can be detected from a single VHR optical satellite image.

5.2 Future work

This work still needs to be improved on further to become more reliable however. Specifically, those parts that remain challenging for my model need to be focused on: slopes in the shade, forested areas and more precise predictions on small avalanches.

Regarding detailed prediction on smaller avalanches, this is likely a problem of the network architecture. Both models tested perform downsampling operations to increase context and reduce computation. Predictions are later upsampled bilinearly. Although skip connections from higher resolution features are used in the decoder, the upsampling process could probably be improved to achieve a better accuracy in small details. This probably needs to be done in combination with a loss that puts more emphasis on those details, such that they are not negligible compared to the mistakes on large avalanches.

To combat the problems regarding forested areas and slopes in the shade, I believe better data is needed for training. For forested areas it might simply be a matter of choosing more samples specifically in forested areas. For shaded slopes on the other hand, it may actually require more accurate labels. As already highlighted in [9], shaded areas are challenging to map avalanches in even for people. Ground truth labels that are obtained through other means than mapping in optical satellite images may be necessary here.

Alternatively, it may be better to combine optical mapping methods with those using SAR. This would not only help detection in shaded areas but also bring the advantages of both sensors together. Since both SAR and optical sensors have been shown to be effective for avalanche mapping even though they have different advantages, combining them could be the next big step.

Appendix

A.1 Pretrained weights

Although one might expect the task of avalanche mapping from satellite images to focus on different features than in standard semantic segmentation tasks, empirical tests showed using pretrained weights had a big effect on the models performance. In fact, the model only converged to reasonable results when using pretrained weights. Figure A.1 shows the difference of convergence during training. Notice that the version without pretrained weights only converges to an F1 score of 50%. While the data imbalance means that this is actually slightly better than random predictions, performance and convergence are still inferior to when using pretrained weights. It was therefore important to retain pretrained weights when customising architectures.

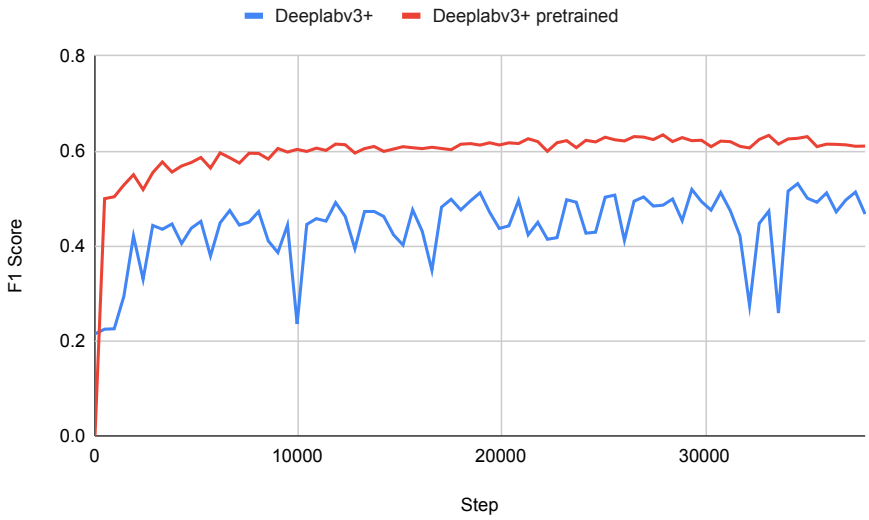


Figure A.1: Effect of pretrained weights on the F1 score when training Deeplabv3+.

A.2 Instance Segmentation

One of the major problems during the development of an avalanche segmentation network was that it was focusing to much on texture without understanding how an avalanche works. This was particularly apparent in rather flat areas where the snow texture looked similar to that of an avalanche resulting in wrong predictions. Any trained person would understand that an avalanche starts in steep terrain and runs out into flatter terrain.

In attempt to tackle this issues and better teach the network to understand what makes avalanche, it was suggested to try instance segmentation. This is a more sophisticated version of semantic segmentation that can tell apart different instances of an object. The intuition behind this was that avalanches are often close and even overlapping in the dataset. Such conglomerates may inhibit the network from properly understanding avalanches as it disturbs some patterns such as that of avalanches starting in steep terrain and stopping in flat areas.

To test this hypothesis, I decided to use the Mask R-CNN framework [10] which was the state of the art in 2017. This has become a well established standard for instance segmentation with many new methods building upon this. The standard Mask R-CNN is already available in the standard torchvision library so I used this.

Having adapted the dataset to include bounding boxes and class labels for each instance, I trained the new model. Although I did get the model to converge, training was 4x slower than for semantic segmentation and therefore did not allow for as much experimentation. The results were also unsatisfactory. An example is shown in Figure A.2. It can be seen that while the semantic segmentation is not too bad, the bounding boxes representing the instances are very cluttered and do not represent the ground truth very well.

Since, the segmentation itself was alright, although bounding boxes and instances were not, I decided to test a newer SOTA instance segmentation framework. The problem with Mask R-CNN is that it has many parameters to tune, especially with respect to the object detection step. CenterMask [16] promised to improve upon this by using the anchor-free, single stage detector FCOS [24], along with other SOTA features. Unfortunately however, it did not make any significant improvement over Mask R-CNN.

There are various reasons that can explain the sub-optimal performance of instance segmentation. For one, avalanche segmentation is a difficult task and telling different instances apart is often impossible even for experts. Additionally, due to the nature of how Mask R-CNN and its variants work, the input images are also distorted in the 'RoIAlign' step. Newer methods like [20] that avoid this are still in development. At this point however, I had to accept that instance segmentation had not helped in better understanding avalanches. While it could be tuned and improved, it only provided more difficulties and therefore made no sense to pursue this path. I therefore returned to semantic segmentation.

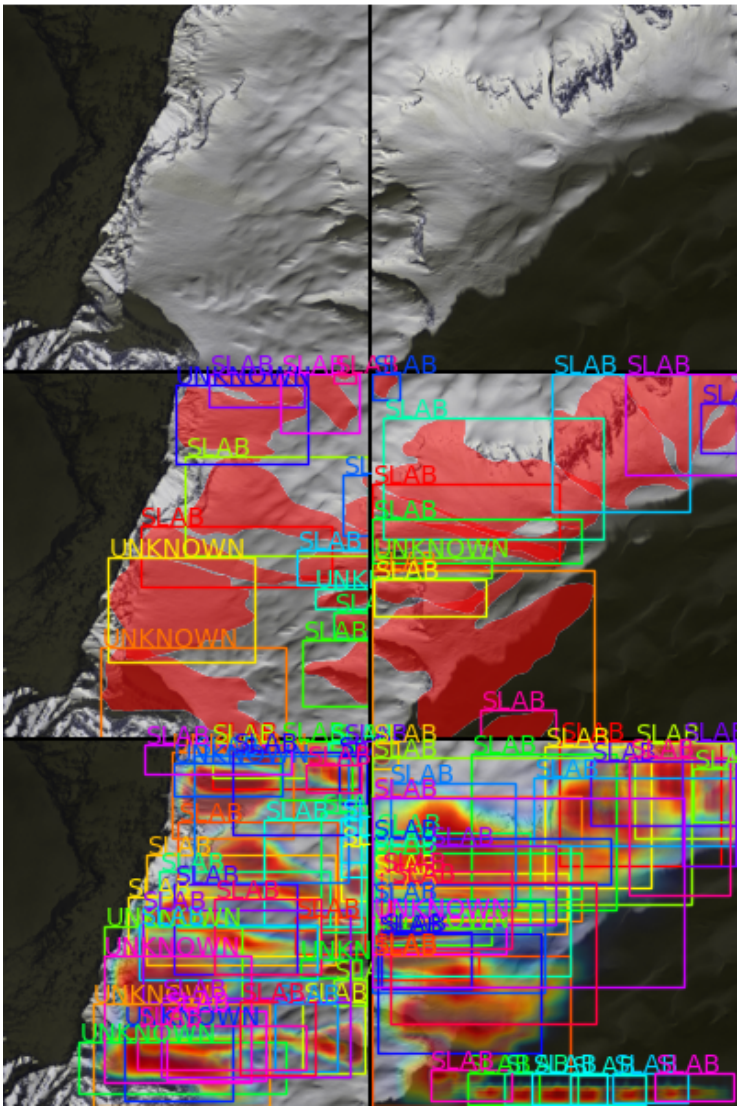


Figure A.2: Instance Segmentation results from Mask R-CNN. Two samples are shown with the input image (top), ground truth labels (middle) and predictions (bottom). Segmentation predictions are most certain where the color is red and less so as the hue changes to blue. The color of the bounding boxes is only to separate instances.

A.3 Evaluation on ground truth data

Around the area of Davos, ground truth data was available from other sources used for the validation of the manual mappings as described in 1.2.3. Since this data was available, I also evaluated with respect this ground truth, although the results are to be interpreted with caution since they are only evaluated on a very small area.

The ground truth data is given in the form of points, each with an attribute determining if there was an avalanche there, if it was old, or if it was unclear. To use this data, I run the avalanche prediction on a 512x512 pixel patch around each point. Avalanche count as detected if the average predicted probability in a 10x10 pixel patch around the point is higher than the 0.5 threshold.

The results are were very similar for both AvaNet and Deeplab and are therefore only shown for AvaNet in Table A.1. The accuracy on the ground truth data is lower than on the labels, as is to be expected since the labels themselves are only have a POD of 74 %. Interestingly however, the ground truth accuracy is higher in 2018 although the label accuracy is almost 10 % lower then. Also note that while most the difference in predictions is from wrong predictions of my model, the automatic mapping does actually also detect some avalanches that have been missed in the manual mappings.

Test Region	Val. points	Label Acc. [%]	GT Acc. [%]	Diff. Correct	Diff. Wrong
Davos 2018	534	73.8	54.1	22	66
Davos 2019	197	80.2	45.5	2	31

Table A.1: Accuracy of predictions of AvaNet when compared to ground truth data, obtained through other means than mapping in optical satellite imagery. The total number of validation points is shown, the accuracy of predictions with respect to the manual labels as well as the ground truth (GT) data. Further, the number of samples classified differently to the manual labels are shown for predictions that were correct and wrong according to the ground truth data. This excludes unknown cases and old avalanches.

A.4 More qualitative results

This section shows some more qualitative results in Figures A.3 to A.6.

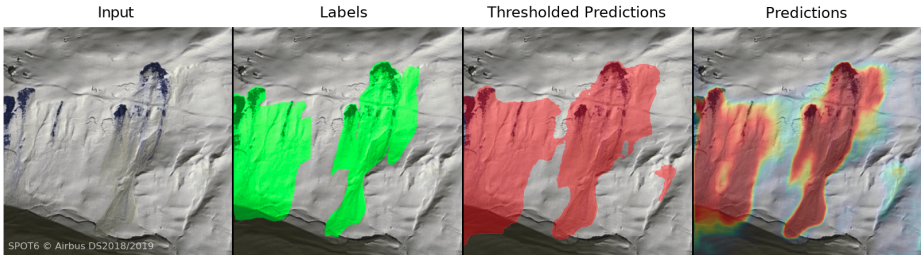


Figure A.3: Although glide avalanches are a less common type of avalanche in the dataset (making up 11 % in 2018 [2]), they are predicted well in this case.

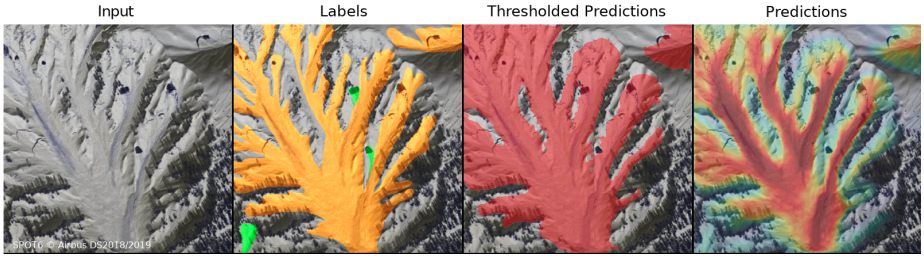


Figure A.4: Detailed prediction even in complex examples, especially in the raw predictions.

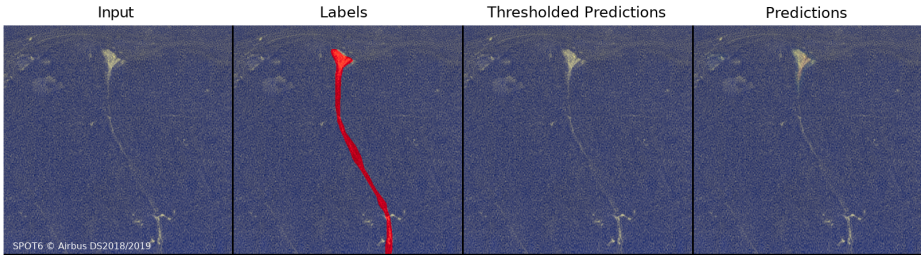


Figure A.5: A typical example the model struggles with is long debris tracks through steep forest. Often context is missing because the avalanches travel a long way into the valley. Furthermore, not many similar examples exist to train on.

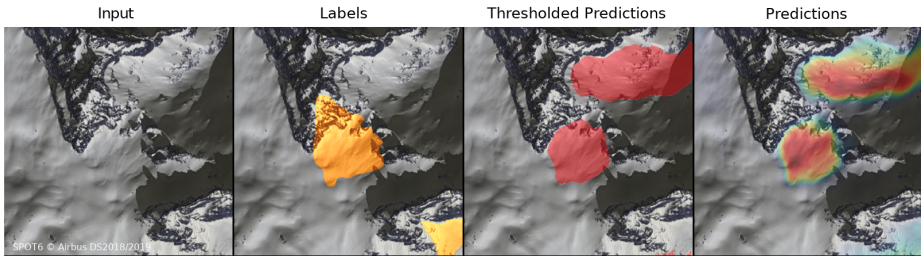


Figure A.6: The model prediction is fairly confident about a second avalanche that is not mapped in the labels. Maybe it was missed during the manual mapping?

Bibliography

[1] Filippo Bianchi, Jakob Grahn, Markus Eckerstorfer, Eirik Malnes, and Hannah Vickers. Snow avalanche segmentation in SAR images with Fully Convolutional Neural Networks. *IEEE*, 2020.

[2] Yves Bühler, Elisabeth Hafner, Benjamin Zweifel, Mathias Zesiger, and Holger Heisig. Where are the avalanches? Rapid SPOT6 satellite data acquisition to map an extreme avalanche period over the Swiss Alps. *The Cryosphere*, 2019.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *ECCV*, 2018.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, M. Enzweiler, Rodrigo Benenson, Uwe Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] Jifeng Dai, Haozhi Qi, Y. Xiong, Y. Li, Guodong Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017.

[6] Markus Eckerstorfer, Hannah Vickers, Eirik Malnes, and Jakob Grahn. Near-Real Time Automatic Snow Avalanche Activity Monitoring System Using Sentinel-1 SAR Data in Norway. *MDPI*, 2019.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. `http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html`.

[8] How to limit RAM usage of python GDAL VRT reading. `https://stackoverflow.com/a/65039417/14180970`.

[9] E. D. Hafner, F. Techel, S. Leinss, and Y. Bühler. Mapping avalanches with satellites - evaluation of performance and completeness. *The Cryosphere Discuss*, 2020.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[11] E. Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: better training with larger batches. *ArXiv*, abs/1901.09335, 2019.

[12] E. Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8126–8135, 2020.

[13] N. Keskar and R. Socher. Improving generalization performance by switching from adam to sgd. *ArXiv*, abs/1712.07628, 2017.

[14] Siri Larsen, Arnt Salberg, and Rune Solberg. Automatic avalanche mapping using texture classification of optical satellite imagery. *EARSeL*, 2013.

[15] M. Lato, R. Frauenfelder, and Y. Bühler. Automated detection of snow avalanche deposits: segmentation and classification of optical remote sensing imagery. *Natural Hazards and Earth System Sciences*, 2012.

[16] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13903–13912, 2020.

[17] Tsung-Yi Lin, Priyal Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:318–327, 2020.

[18] Joe McGlinchy, Brian Johnson, Brian Muller, Maxwell Joseph, and Jeremy Diaz. Application of UNet Fully Convolutional Neural Network to Impervious Surface Segmentation in Urban Environment from High Resolution Satellite Imagery. *IGARSS*, 2019.

[19] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image Segmentation Using Deep Learning: A Survey. *arXiv*, 2020.

[20] Lu Qi, X. Zhang, Y. Chen, Yukang Chen, J. Sun, and Jiaya Jia. Pointins: Point-based instance segmentation. *ArXiv*, abs/2003.06148, 2020.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *MICCAI*, 2015.

[22] Saumya Sinha, Sophie Giffard-Roisin, Fatima Karbou, Michaël Deschatres, Anna Karas, Nicolas Eckert, Cécile Coléou, and Claire Monteleoni. Can Avalanche Deposits be Effectively Detected by Deep Learning on Sentinel-1 Satellite SAR Images? *Climate Informatics*, 2019.

[23] Avalanche Fatalities in Switzerland. `https://www.slf.ch/en/avalanches/destructive-avalanches-and-avalanche-accidents/long-term-statistics.html`.

[24] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9626–9635, 2019.