John Isak Furuseth

Automation of landslide detection using Deep Learning

Master's thesis in geotechnology
Supervisor: Ola Fredin
Co-supervisor: Erin Lindsay
June 2022

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Geoscience and Petroleum

John Isak Furuseth

Automation of landslide detection using Deep Learning

Master's thesis in geotechnology
Supervisor: Ola Fredin
Co-supervisor: Erin Lindsay
June 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Geoscience and Petroleum

Abstract

Landslides events frequently affect Norwegian communities. Events that impact roads and other infrastructure are systematically reported based on ground observations; however the resulting database shows spatial bias towards such infrastructure, and is incomplete in other areas. Recent studies show that Deep Learning (DL) can be used to detect and map landslides in satellite images. This study investigates whether there is potential to improve the landslide database in Norway using these methods. It is the first attempt of an automated DL process using Norwegian landslide data.

A DL model available in ArcGIS Pro was tested by varying the training data set scale (national and local) and the combination input bands used for training the classifier. The Jølster landslide inventory (n=120) from the July 30, 2019 summer rainstorm was used for verification of the prediction results.

Preparation of the training data set involved (i) selecting events, (ii) acquiring satellite images, and (iii) verification of the events. Firstly, the national landslide polygon database and other inventories were filtered for post-2017 events greater than 1000 m^2. Excluding landslides from the verification set, 21 events were available. Sentinel-1 and Sentinel-2 satellite images of these landslide events were acquired for each event location using Google Earth Engine (GEE). All landslides were delineated during the verification process, resulting in a final set of 52 labelled landslide polygons to use for the training of national DL models. The DL model was then trained using seven different combinations of input data, including bands from optical and synthetic aperture radar satellites, as well as digital elevation models.

It was found that the input data that was best suited for detecting landslides was an image featuring difference Normalised Difference Vegetation Index (dNDVI). With dNDVI input data it was achieved a precision of 0.30, recall of 0.36, F1 of 0.33 and a Matthews Correlation Coefficient (MCC) of 0.33.

No significant improvement was observed when using a local training sample set compared to a national sample set. Recall values were slightly higher with the models trained on local data set, while precision scores were lower than that of the national model. Precision, F1, MCC and recall values are considerably lower than that of other studies in literature. This is likely due to unrepresentative training samples, low sample size, and unsuitable DL parameters.

These results indicate that there is promising potential to improve landslide data using these methods, however the results were limited by the lack of available training data and not having optimal DL parameters. Further studies could benefit from a bigger sample size and less noise in training samples.

Sammendrag

Jordskred rammer ofte norske lokalsamfunn. Hendelser som treffer veier og annen infrastruktur rapporteres systematisk basert på bakkeobservasjoner; den resulterende databasen viser imidlertid romlig skjevhet mot slik infrastruktur, og er ufullstendig på andre områder. Nyere studier viser at dyplæring (Deep Learning, DL) kan brukes til å oppdage og kartlegge skred i satellittbilder. Denne studien undersøker om det er potensial for å forbedre skreddatabasen i Norge ved hjelp av disse metodene. Det er første forsøk på en automatisert DL-prosess som bruker norske skreddata.

En DL-modell tilgjengelig i ArcGIS Pro ble testet ved å variere treningsdatasettets skala (nasjonalt og lokalt) og ulike kombinasjoner av inndatabånd som brukes til å trene klassifisereren. Skredinventar (n=120) fra sommerstormen i Jølster 30. juli, 2019 ble brukt til å verifisere prediksjonsresultatene.

Forberedelse av treningsdatasettet innebar (i) valg av hendelser, (ii) innhenting av satellittbilder, og (iii) verifisering av hendelsene. Førs ble den nasjonale skredpolygonen database og andre inventar filtrert for hendelser etter 2017 og areal over 1000 m^2. Ved å ekskludere skred fra verifikasjonssettet var 21 hendelser tilgjengelige. Sentinel-1 og Sentinel-2 satellittbilder av disse skredhendelsene ble innhentet for hver lokasjon ved hjelp av Google Earth Engine (GEE). Alle skred ble avgrenset under verifiseringsprosessen, som resulterte i et endelig sett med 52 merkede skredpolygoner til bruk for trening av nasjonale DL-modeller. DL-modellen ble deretter trenet ved hjelp av syv forskjellige kombinasjoner av inngangsdata, inkludert bånd fra optisk- og syntetisk apertur-radarsatellitter, samt digitale høydemodeller.

Det ble funnet at den inndataen som var best egnet for å oppdage skred var et forskjellsbilde av normalisert differanse-vegetasjonsindeks (dNDVI). Med dNDVI-inndataene ble det oppnådd en presisjon på 0,30, recall på 0,36, F1 på 0,33 og en Matthews korrelasjonskoeffisient (MCC) på 0,33.

Ingen avgjørende forbedring ble observert ved bruk av et lokalt treningsdatasett sammenlignet med et nasjonalt treningsdatasett. Recall-verdien var litt høyere med modellene trent på lokale datasett, mens presisjonsskårene var lavere enn de nasjonale modellene. Presisjons-, F1-, MCC- og recall-verdier er betydelig lavere enn samme verdier i andre studier. Dette skyldes sannsynligvis ikke-representative treningsutvalg, lav sample størrelse og uegnede DL-parametere.

Disse resultatene indikerer at det er et lovende potensial for å forbedre skreddata ved å bruke disse metodene, men resultatene var begrenset av mangelen på tilgjengelig treningsdata og ikke å ha optimale DL-parametere. Ytterligere studier kan dra nytte av mer treningsdata og mindre støy i treningdataen.

Preface

This thesis concludes a five years integrated Master's programme in engineering geology at the Norwegian University of Science and Technology (NTNU), the Department of Geoscience and Petroleum (30 ECTS credits). The thesis is written in its entirety by John Isak Furuseth.

The topic of this study was proposed by Erin Lindsay, to complement her PhD research concerning landslide detection for the purpose of improving Landslide Early Warnings. It will be conducted as a part of KLIMA 2050, work package 3: Landslides triggered by hydrometeorological processes (Klima 2050, 2020). KLIMA 2050 is one of many centres for researched-based innovation (SFI) financed by The Research Council of Norway (Norges forskningsråd).

For an insider it might seem foolish to write a master's thesis on deep learning with no prior knowledge of Python—and yes, it is foolish. I have not taken a single course on machine learning, data analysis or advanced programming. All I know is rock, yet, I still made it work! It is easy to sound impressive when talking about Deep Learning. People rarely know what it really means, but they think it is impressive. I have learned that it is, in fact as complicated and impressive as it first seemed. Though the results were not as hoped for, I have learned. I have learned a great deal about management of satellite data and data preparation for deep learning.

I would like to thank main supervisor Ola Fredin for advice, meetings and motivation. You have really helped in making this thesis possible by being a good and understanding man, and providing helpful feedback.

A big thanks goes also to my co-supervisor Erin Lindsay for proposing the topic and for helpful insight in the issue of landslide mapping from remote sensing images. I have learned a lot from you the past month. I am very glad I decided to work on this topic! It has suited me well and you have made it possible by always being accessible when needed.

I would also like to thank Alexandra Jarna for helping giving this thesis the proper direction in the early stage of the project.

Finally, I would like to thank my fellow students in the geotechnology class. Thanks for all the good times in lectures, on field trips, in the study hall, and in soscial gatherings. I am going to miss the times I have had with you, really.

Writing this thesis English was not the easiest decision. I am an advocate for Norwegian language, both written and spoken and believe that Norwegian needs to be used in academia in order to make the research relevant to the Norwegian public. Still, the field of Machine Learning is very much centred around the English language and I am not the one to create a Norwegian vocabulary to describe this field.

Enjoy.

Contents

List of Figures

List of Tables

List of Abbreviations

SWIR	Shortwave infrared, 14
CLC	Corine Land Cover, 17
CMM	Confusion matrix map, 37
CNN	Convolutional Neural Network, 17
DEM	Digital elevation model, xii
DL	Deep Learning, 1
EM	Electromagnetic, 10
ESA	European Space Agency, xii
FN	False negative, 25
FP	False positive, 25
GEE	Google Earth Engine, 2
IDE	Integrated Development Environment, 30
InSAR	Interferometric Synthetic Aperture Radar, 15
IR	Infrared, 10
LDDB	Landslide Detection Database, 30
LGM	Late glacial maximum, xii
LiDAR	Light Detection And Ranging, 11
MCC	Matthews Corrolation Coefficient, 26
ML	Machine Learning, xii
MSI	Multispectral instrument, 13
NDVI	Normalised differential vegetation index, 16
NIR	Near-infrared, 16
NVE	The Norwegian Water Resources and Energy Directorate (Norges vassdrags- og energidirektorat), xii

NVE	The Norwegian Water Resources and Energy Directorate (Norges vassdrags- og energidirektorat), 1
S1	Sentinel-1, xii
S2	Sentinel-2, 12
SAR	Synthetic aperture radar, 12
SFI	The Centres for Research-based Innovation scheme (Sentre for forskningsdrevet innovasjon), v
SWIR	Shortwave infrared, xii
TN	True negative, 25
TP	True positive, 25
VIS	Visible light, 10

Glossary

Temporal	Relating to time as distinguished from space, 14
Aerosol	Tiny particles or droplets in suspension in air, e.g. fog, mist or dust, xii
Cirrus	High altitude clouds (4–6 km a.s.l.), xii
Inferencing	The process of using a trained model to make a prediction, i.e. putting a machine learning model into production, 25
LS_ID	Landslide Identifier. Arbitrary number given in the LDDB to keep track of all the different landslides in the database., 30
Noise	Random fluctuations in pixel brightness, 24
Overfitted model	A model that fits its training data well but is not generalised enough to predict or see samples not encountered during training, defeating the purpose of the model., 24
Transfer Learning	Transfer learning for machine learning is when existing models are reused to solve a new challenge or problem., 24

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Landslide events frequently affect Norwegian communities. Events that impact roads and other infrastructure are systematically reported based on ground observations; however, the Norwegian landslide inventory (www.skredregistrering.no) shows spatial bias towards such infrastructure, and is incomplete in other areas (Jaedicke et al., 2009). The inventory is maintained by the Norwegian Water Resources and Energy Directorate (NVE). Events for the database are collected/registered by road and railway authorities, municipalities, private consultants and public from field observations, historical documents or media (Devoli, 2017).

Given the under-reporting of landslides that do not impact the transport network or other infrastructure, it is wanted to expand the Norwegian national landslide inventory by mapping landslides using images from remote sensing Satellites such as Sentinel-1 and Sentinel-2 (Lindsay et al., 2021). Historically, such mapping has been done by manual mapping from remote sensing images and ground observations (Highland and Bobrowsky, 2008). These are, however, time consuming and labour intensive methods (Guzzetti et al., 2012).

Recent years have shown an increase in studies involving automated landslide mapping using deep learning (DL) methods (Prakash et al., 2020; Prakash et al., 2021; Ghorbanzadeh et al., 2019; Lei et al., 2019; Nava et al., 2021; Herrera Herrera, 2019). Recent studies also indicate that deep learning methods such as convolutional neural networks (CNN) will outperform traditional machine learning algorithms in landslide detecting tasks (Sameen and Pradhan, 2019). Many of these studies feature inventories with earth quake triggered landslide events, with some events triggering several thousands single landslides (Prakash et al., 2021). Such cases are not common in Norway, and therefore it would be of interest to assess how DL models trained on Norwegian landslides perform.

Public access to satellite-based data sets are made available by platforms such as

Google Earth Engine (GEE)[1], USGS Earth Explorer[2] and Copernicus Open Access Hub[3]. These platforms enables viewing and time series analysis of satellite imagery on a global scale (Herrera Herrera, 2019).

By implementing Python Deep Learning packages (Esri, 2021c) directly in the ArcGIS Pro software, Esri has made the process of exploring a (DL) problem more accessible for someone without primary programming knowledge. The user interface allows for easy management of satellite imagery and training of DL models. A range of different computer vision tasks can be performed, including image classification, object detection, semantic segmentation, and instance segmentation (Esri, 2021d). Most interesting for the task of landslide mapping is semantic segmentation and instance segmentation, as those methods allows more precise detection of the boundary of each feature (Esri, 2021d).

1.2 RESEARCH QUESTIONS

Based on the problem introduced in the previous section, the research question is presented:

> ***How can Deep Learning be applied to detect landslides from satellite images?***

To answer the main research question, a set of sub-questions are formulated as follows:

1. How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?
2. What input data will be best suited for detecting landslides?
3. Will training on a national set of landslides compare differently from training only on local training samples?

1.3 DESCRIPTION OF STUDY AREAS

The relevant areas for this study are split into two groups. One group for the Jølster case, where 120 landslides have been thoroughly verified and delineated by Lindsay et al. (2021). This case is limited to one specific area with the same triggering event for all the landslides (Heavy rainfall). Since the landslides in this area are well documented it will serve as a "testing" area for the models that will be trained on samples from the other study areas, as well as the models trained on the local samples.

The other group is for all the landslides that are to be used as training samples for the national deep learning models. This group is made up of 22 different areas in which one or more landslides have been reported to the Norwegian national landslide database (www.skredregistrering.no), and later verified by examining satellite images before and after the reported event dates.

[1]https://earthengine.google.com/
[2]https://earthexplorer.usgs.gov/
[3]https://scihub.copernicus.eu/

1.3.1 Jølster

On the 30th of July, 2019 the municipality of Jølster, In Western-Norway, was hit by a very heavy rainstorm. The town of Vassenden was most notably affected. Many weather stations recorded more than 20 mm of rain in one hour and one station in Haukedal recorded as high as 38 mm in one hour. (Meteorologisk Institutt, 2019) As a consequence, a series of landslides were triggered over the course of five hours. Infrastructures were severely damaged, roads were flooded and closed, and one person died (Grov, 2020). More than 150 people had to be evacuated from the area and several cars were abandoned in the midst of the landslides. It was estimated by insurance companies that the total cost of cleanup and claims for compensation would be at least 50 million NOK (NTB, 2019).

In total 120 recorded landslides was triggered that day (Lindsay et al., 2021). As tragic as the outcome of this event was, the vast landslide count in a relatively small area, by Norwegian standards, does make for a good area to test an landslide detection model. Many landslides in the region are visible in the 10 m resolution images from the Sentinel satellites, which is of great value.

Geologically Jølster is located in the Baltican basement with Caledonian overprint (deformation). The dominant rock type is Granitic to dioritic gneiss, in places augen gneiss, in places migmatitic. Surrounding areas show patches of Caledonian mylonite, augen gneiss and quartz schist, see Figure 1.1 (Geological Survey of Norway, 2021). The topsoil in the area is a result of glacial activity following the final Ice Age. It is dominated by till, in places very thick (See Figure 1.2). The land cover is mostly forest and sparsely vegetated areas (See Figure 1.3).

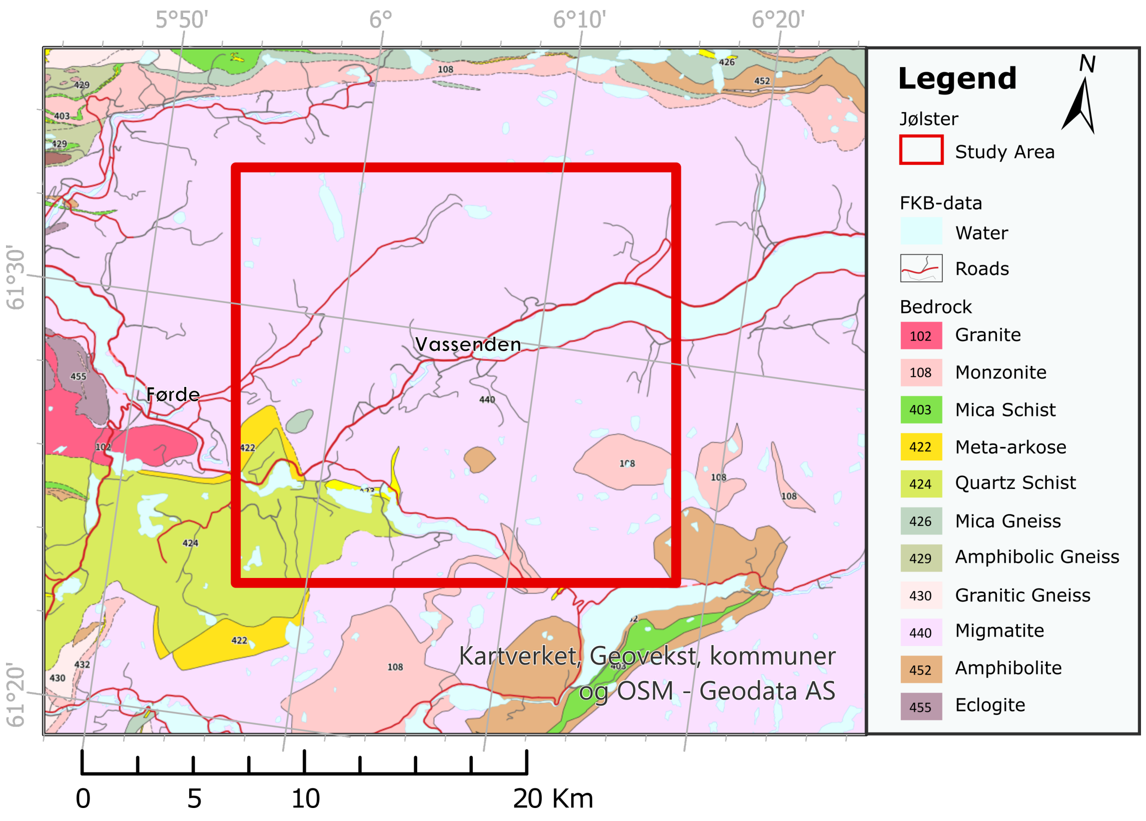


Figure 1.1: *Bedrocks in the Jølster Case Area. From NGU 1:250 000 National bedrock map.*

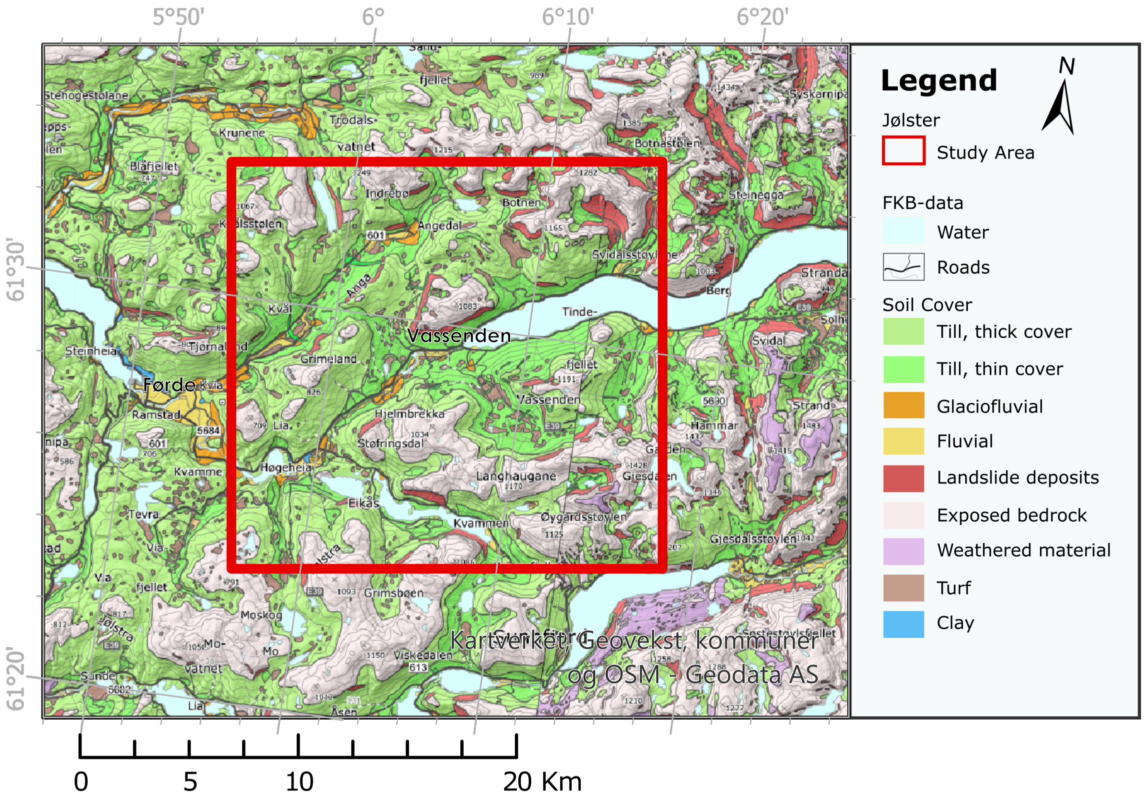


Figure 1.2: *NGU 1:50 000 Quaternary materials map from the study area.*

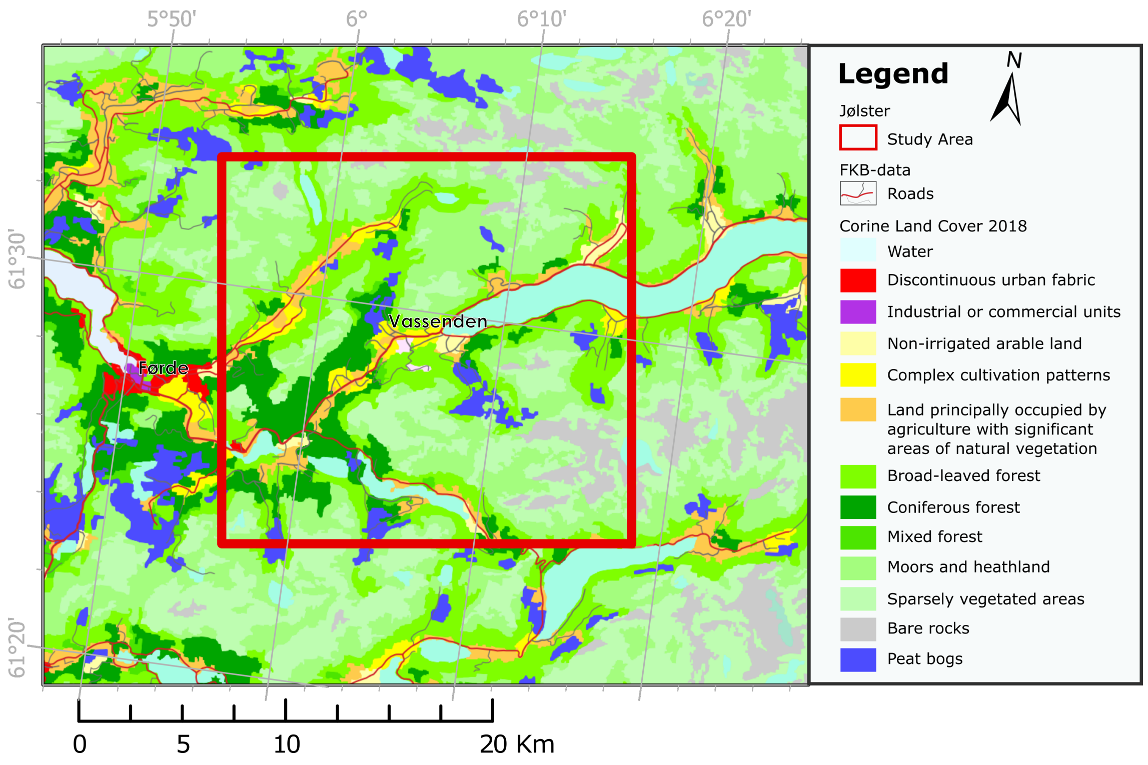


Figure 1.3: *Corine Land Cover for the study area.*

1.3.2 National training areas

To get sufficient training samples to train a DL model at detecting landslides it has been necessary to look at the events that have been registered for all of Norway. The areas where these events have occurred, hereby named *national training areas*, have been selected based on a set of criteria described in the methodology chapter (chapter 3. The training areas have been selected from all over Norway. This does not include Mid-Norway as no landslide met the criteria for selection in this region.

Not all reported landslides are reported accurately. Some might have reported a land-slide volume that is higher than what is really is. The consequence of this is that the landslide is not visible in the 10 m resolution images from Sentinel. Other landslides might show wrong event date or just an approximate location of the landslide. It is therefore necessary to do a manual verification of the database. This is done by using GEE, also described in the methodology chapter (chapter 3).

1.4 THESIS OUTLINE

This thesis document is structured as follows:

- chapter 2 reviews the theoretical background that was needed to undertake the work in this thesis.
- chapter 3 presents the methods that was used to train a deep learning model to detect landslides from satellite images. This will in part answer the second sub-research question.
- chapter 4 describes the implementation details of the methodology and presents

the results.

- chapter 5 addressees and discusses the results in light of other related works.
- summarises the discussions and answers the main research questions.

It is expected that the reader has some previous knowledge of DL and CNN, as not all important details concerning the DL methods will be explained.

Post and *pre* will always in this study refer to respectively, the time *after* a landslide event and the time *before* a landslide event.

1.5 CONTRIBUTION

The analyses and works completed by the author personally are:

- Checking availability of satellite images from reported landslide events in Norway between 2018 and 2021. This was done with the help of Google Earth Engine and a script written by Erin Lindsay. The time series of the NDVI-value for a given point in the landslide have been the primary factor for deciding weather a landslide is detectable or not.
- Exporting image tiles of landslides and surrounding area: Also done with the help of GEE and script by Erin Lindsay.
- Processing of the image tiles in ArcGIS Pro (Esri, 2021a): Every image tile had to be combined to make one single raster file in order to further process the ... for training data creation.
- Adding and removing bands for training data.
- Exporting training samples and training of Deep Learning models using ArcGIS Pro tools.
- Computing confusion matrix maps for each DL model.

The main contributions of this study are:

- High spatial variability in training data. Many uses training data related to the same triggering event. This approach can be used to search out landslides from a ... and joining them in the same
- Data acquisition using free-of-charge resources. GEE provides a —- for those familiar with coding in javascript(?). The preparation of the data and machine learning is done using commercial software, although

1.6 SCOPE AND LIMITATIONS

The aim of this work is to investigate the possibility of using the Deep Learning Libraries in ArcGIS Pro for automated detection of landslide activity in Norway. This thesis will present a method for acquisition and processing of imagery for use in automated mapping of landslides in Norway. It will also propose the data input that shows best promises to detect landslides.

- Due to the uniqueness of the problem and limitations in time only one type of deep learning algorithm will be explored. This is the Mask RCNN algorithm (He et al., 2017).
- Google Earth Engine is a free tool for academic and research use. However, other uses will require users to sign additional terms and agreements with Google.
- This study aims at providing an method for creating a landslide classifier and not to optimise the DL algorithm for detecting landslides
- The triggering factor for each landslide will not be determined. This study is only for *mapping* landslides.

CHAPTER 2

THEORETICAL BACKGROUND

This chapter is intending to present

The theory in section section 2.1 will be jointly taken from the preliminary works for this thesis done in Furuseth (2022) with some minor alterations.

2.1 REMOTE SENSING FOR EARTH OBSERVATION

Remote sensing is the science — and art — of gathering information about an object, area or phenomenon through a device that is not in direct contact with the object, area or phenomenon under investigation (Lillesand and Kiefer, 1979).

The general way of gathering this information is through instruments mounted on aircrafts or satellites (US Geological Survey, 2021), but as Figure 2.1 from Yang et al. (2013) suggest, remote sensing can also be carried out using boats, floating sensors or ground-based instruments. The choice of vessel depends on the features or objects that are to be monitored.

Remote sensing technology allow us to see much more than we can see when standing on the ground, which is of great help when we want to gather information quickly and for a large area (Yang et al., 2013). This study will have its focus on optical sensors, i.e. cameras.

The first known use of airphotos for geological mapping was in Libya in 1913. But it was not until the 1940's that interpreting and evaluation of the geology from airphotos became a widespread technique (Lillesand and Kiefer, 1979, p. 112). In the same period (1946) small cameras where mounted on American V-2 rockets launched in to space. It became the crude start of *Earth observation* (EO) from space.

As of the end of 2020, there was more than 900 EO-satellites in orbit (Mohanta, 2021). Many of these satellites are used for climate studies and the most common EO satellites are *Earth imaging* satellites (United Nations Office for Outer Space Affairs, 2021). They are a literal 'step out' from aerial photographs in that they captures images of the scenery below them, only from space (Lillesand and Kiefer, 1979).

Laser scanning techniques such as LiDAR (Light Detection And Ranging) is also considered as remote sensing. It is a useful supplement for landslide mapping, in that it

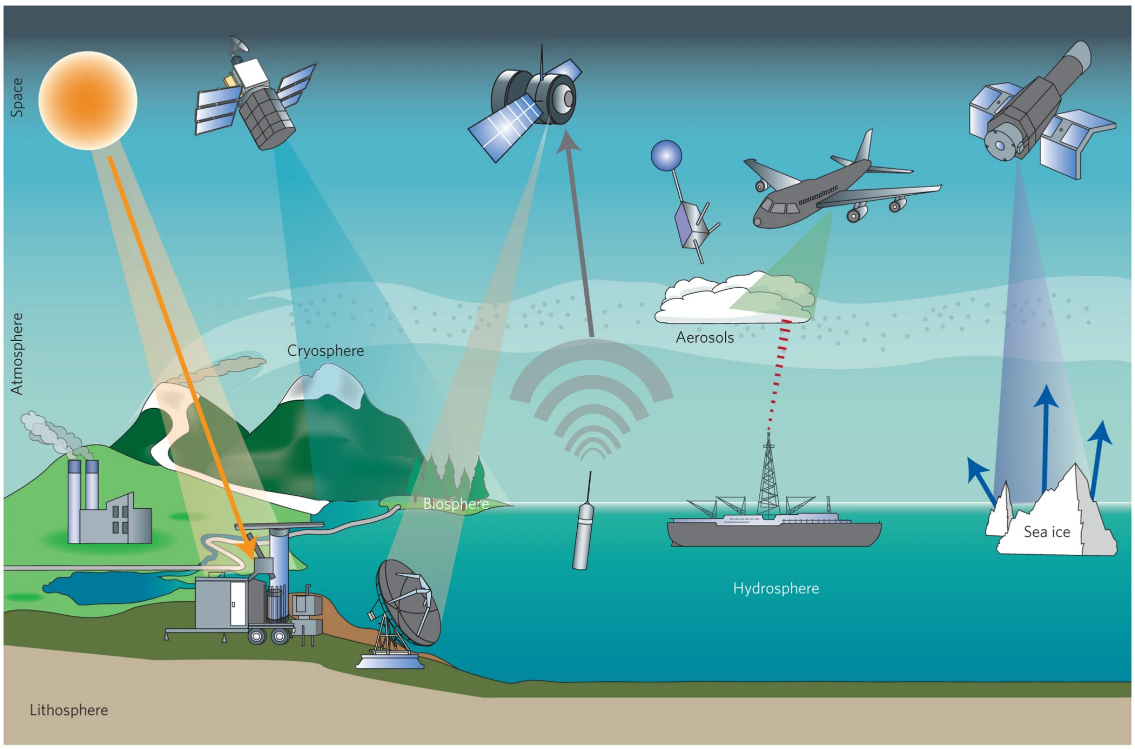


Figure 2.1: *Different remote sensing apparatuses and uses. In climate observation remote sensing is carried out using a variety of platforms, including plane, boat, floating sensors and ground-based instruments. Figure from Yang et al. (2013).*

can be used to create a digital elevation model (DEM) e.g. Ji et al. (2020), but the technique will not be given much further attention in this study.

2.1.1 Electromagnetic energy

The basis for the principle of remote sensing is the measurement of light intensity, i.e. electromagnetic (EM) energy, or radiation. The EM radiation consists of photons, that travels at 300 000 km/s. In remote sensing literature the EM radiation is often given as a wavelength. The EM spectrum ranges from short wavelengths, like gamma and x-rays, to longer wavelengths, like micro and radio waves. In between those ranges we find the *visible light* (VIS) range and the *infrared* (IR) range (Lillesand and Kiefer, 1979). Figure 2.2 shows the components of the electromagnetic spectrum and the relative size of the visible and infrared portions of the spectrum.

The power of remote sensing is that it not only captures light in the visible spectrum, but also in other regions of the electromagnetic spectrum. The visible light and IR range are the most important EM ranges in optical remote sensing as visible light is the easiest for human eyes to interpret and IR energy can tell us a great deal about the presence of vegetation (Lillesand and Kiefer, 1979).

The shortest wavelengths that are practical to measure for remote sensing we find in the ultraviolet (UV) region of the spectrum. Some rocks and minerals emit visible light when illuminated by UV radiation (Canada Centre for remote Sensing, 2019, p. 9).

Although names are given for set regions of the electromagnetic spectrum, e.g. in-

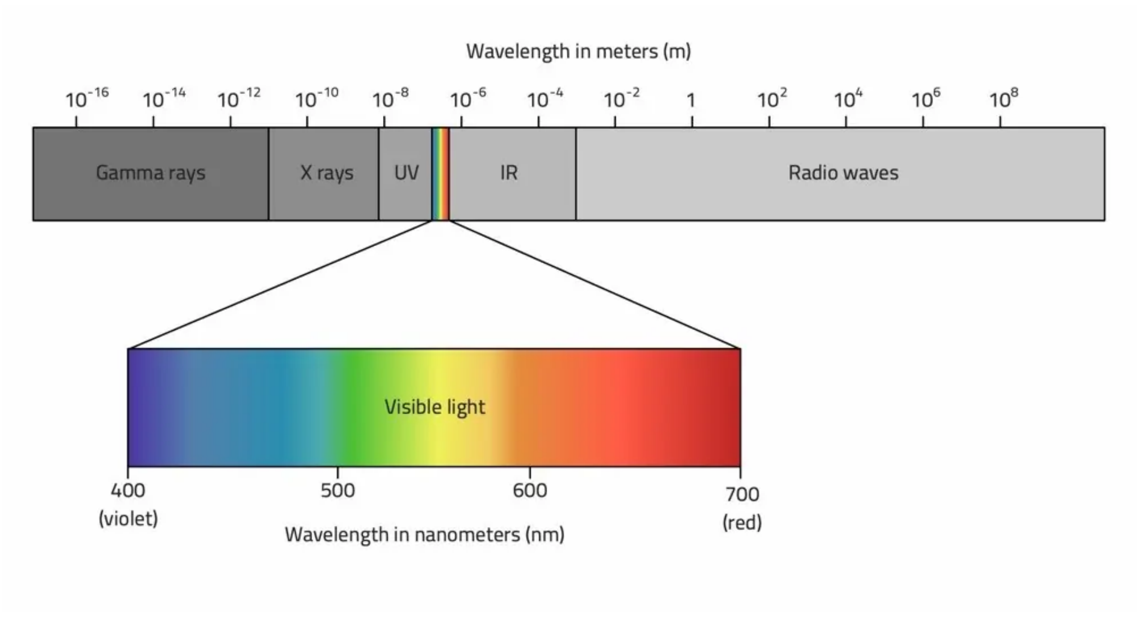


Figure 2.2: *Components of the electromagnetic (EM) spectrum. Figure from Radio2Space (2013).*

frared or ultraviolet, there is no clear-cut dividing line between two spectral regions. The names are assigned out of convenience. (Lillesand and Kiefer, 1979, p. 4).

2.1.2 Passive vs active sensor

There are two types of remote sensing sensors: Passive and active sensors. A **passive sensor** measure the energy that is naturally available, such as the reflected or re-emitted solar energy. All passive sensors rely on energy that is either reflected and/or emitted from earth surface features (Lillesand and Kiefer, 1979, p. 27). The visible wavelengths are reflected from the surface they hit, while the thermal infrared wavelengths are re-emitted (Canada Centre for remote Sensing, 2019).

Passive sensors require the naturally occurring energy to be present to capture any useful energy. The sun is a very efficient and convenient provider of such natural energy. For the reflected solar energy, this can only come about as long as the sun is illuminating the Earth. This means that at nighttime or in polar regions during winter there is no reflected energy available (Canada Centre for remote Sensing, 2019). Infrared energy (heat radiation) can be naturally emitted and can be detected both day and night, given that the emitted energy is large enough to be observed. (Canada Centre for remote Sensing, 2019).

However, an **active sensor** will provide its own energy source for illumination. An example of such a sensor is a LiDAR (Light Detection And Ranging) system, which emits its own laser signal. The sensor is equipped with an instrument that emits radiation directed towards the target surface, which then gets illuminated and the reflected energy can be measured the same way as for a passive sensor (Canada Centre for remote Sensing, 2019).

A common camera is an example of a device that act both a passive and an active sensor. In a bright environment it passively records the reflected light from either the sun or other light sources, but if it is too dark a built in flash will illuminate the subject allowing it to be imaged (Canada Centre for remote Sensing, 2019).

A **SAR** (Synthetic aperture radar) is another example of an active sensor. It is a remote sensing imaging radar capable of providing high-resolution images for a multiple of applications. It can operate regardless of cloud cover and sunlight making it very useful for EO purposes. As opposed to optical sensors, visualising raw data from the SAR sensor does not provide any helpful information from the scene. As can be seen in Figure 2.3, it is necessary to apply a set of filtering operations to visualise the data in an understandably manner (Moreira et al., 2013).

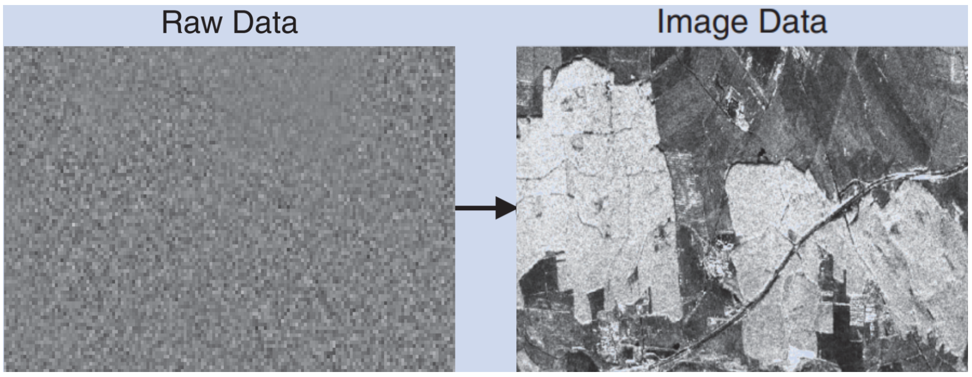


Figure 2.3: *Signal processing of SAR data. SAR data is not straight forward to visualise, The arrow indicates a complex process of range compression, convolution and complex conjugation, resulting in a picture that is readable to the human eye. Modified from Moreira et al. (2013).*

The use of SAR data has proven useful in landslide detection (Nava et al., 2021). Nava et al. (2021) found that SAR data may provide comparable accuracy to classical optical landslide detection with overall accuracies of 99.20 % for optical images and beyond 94 % for SAR data.

However, the results from Lindsay et al. (2021) was not as promising. With only 9 out of 120 landslides detected using SAR images the authors concluded that no definitive trends could be found on why some landslides were detected and som were not.

2.1.3 Sentinel-satellites

Sentinel is the name of the series of EO satellites launched by the European Space Agency (ESA) as a part of the Copernicus program. Copernicus is the European Union's program for Earth observation. The program includes both EO satellites, airborne sensors and ground-based observation stations (Tandberg, 2013). Data from these satellites will be the ones used for the further study following this project report.

The Sentinel-2 mission consist of two polar-orbiting EO satellites, Sentinel-2A (S2A) and Sentinel-2B (S2B). They both have onboard sensors that captures EM radiation that produces images of the Earth. S2A and S2B are placed in the same *sun-synchronous* orbit, phased at 180° to each other. This gives a revisit time – the time between two passages – of only 5 days (ESA, 2020).

A sun-synchronous orbit is an orbit in which the satellites passes over any given point on Earth in the same 'fixed' position relative to the sun. The satellites are thus synchronised with the sun (Lillesand and Kiefer, 1979, p. 532). This means that the S2 satellite will always observe a location as if it was in the same local time. This is important, as the surface illumination on the ground will be nearly the same for images taken days, week, months or years apart. It makes monitoring changes over time easier (Canada Centre for remote Sensing, 2019).

There are many more providers of satellite imagery, e.g. the US's Landsat-program, which is also publicly available – Planet and Worldview are examples of private companies that specialises in satellite imagery. Their services are primarily available through subscription only.

2.1.4 Radiometric resolution

The radiometric resolution represents the ability of a digital sensor to distinguish differences in light intensity or reflectance, and is measured in bits. A greater radiometric resolution means a more accurate depiction. The bit range is typically in the range of 8 to 16 bits, yielding respectively $2^8 = 256$ to $2^{16} = 65\,536$ different values in light intensity that the sensor can distinguish between (ESA, 2021a).

The Multispectral Instrument (MSI) mounted in S2 is 12 bit, making it possible to image light intensity values in the range from 0 (full black) to 4095 (full white). The MSI in S2 has 13 sensors that captures light in different spectral bands that range from the visible (VNIR) and Near Infrared (NIR) to the shortwave Infrared (SWIR) (ESA, 2021a).

2.1.5 Spatial resolution

The spatial resolution is a quality of the remote sensing sensor and refers to the size of the smallest feature on the Earth surface that can possibly be detected. This is naturally dependent on the operating altitude of the sensor. The same sensor will have a better spatial resolution if mounted onboard an aircraft than onboard a satellite (Canada Centre for remote Sensing, 2019). Within satellite sensing, the spatial resolution can range from >100 m to 31 cm (ESA, 2021b) depending on which spectral band and what image provider is used.

Figure 2.4 shows the different spectral bands and their respective spatial resolution collected from the S2 satellites. Spatial resolution is on the y-axis and wavelength is on the x-axis (ESA's Sentinel-2 team, 2015).

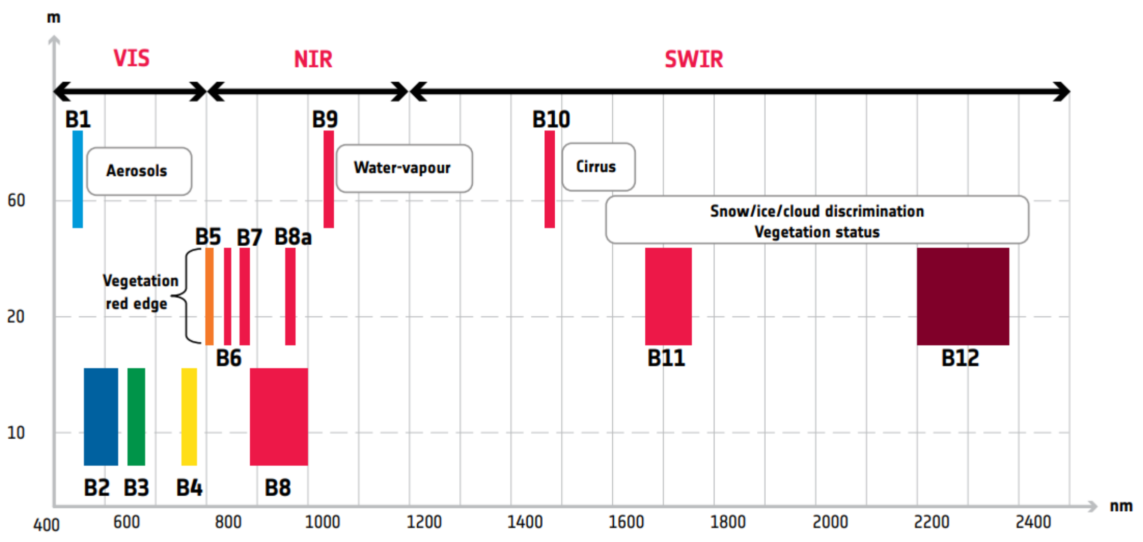


Figure 2.4: *Sentinel-2 spectral bands. Ranging from the visible spectre (VIS) to the Near-InfraRed (NIR) and ShortWave InfraRed (SWIR) spectre. The 60 m resolution bands are not actively used in landslide detection. There is no band in the 900–1300 nm range because ozone, water, carbon dioxide, and other molecules in the atmosphere absorbs light in this range (GISGeography, 2021b). Figure fetched from ESA's Sentinel-2 team (2015).*

The two S2 satellites have roughly the same sensors onboard, but there are some slight deviances as can be shown in Table 2.1.

Although not a physical property, the **temporal resolution** is used to specify how long it takes for the same area to be covered by the remote sensing sensor. The term *revisit time* is a synonym for temporal resolution (Canada Centre for remote Sensing, 2019).

Table 2.1: *Spectral Bands for Sentinel-2A and 2B. From ESA (2021a).*

			S2A		S2B	
Band	Description	Res. [m]	Central Wavelength [nm]	Bandwidth [nm]	Central Wavelength [nm]	Bandwidth [nm]
B1	Aerosols	60	443.9	21	442.3	21
B2	Blue	10	496.6	66	492.1	66
B3	Green	10	560	36	559	36
B4	Red	10	664.5	31	665	31
B5	Red Edge 1	20	703.9	15	703.8	16
B6	Red Edge 2	20	740.2	15	739.1	15
B7	Red Edge 3	20	782.5	20	779.7	20
B8	NIR	10	835.1	106	833	106
B8A	Red Edge 4	20	864.8	21	864	94
B9	Water vapor	60	945	20	943.2	21
B10	Cirrus	60	1373.5	31	1376.9	30
B11	SWIR 1	20	1613.7	91	1610.4	94
B12	SWIR 2	20	2202.4	175	2185.7	185

2.2 TRAINING DATA RESOURCES [CONSIDER RENAME HEADING]

In this section a few relevant resources related to the input data for landslide detection will be presented. These are Digital Elevation Model (DEM) and NDVI (Normalised Differential Vegetation Index). I addition, the Corine Land Cover (CLC) inventory will be presented. It does not contribute to the Deep Learning models, but provides relevant information on the land cover surrounding the landslides.

2.2.1 Digital Elevation Model

A digital elevation model (DEM is, for the purpose of this study, simply a georeferenced raster graphic where the value of each pixel corresponds to the elevation at that geographic point. From the DEM *slope inclination values* and *hillshade* maps can be derived using ArcGIS Pro or any other GIS software. The slope map can be used as an input layer for the training data as well as the DEM, and hillshade maps are great for visualisations. Figure 2.5 shows examples DEM and the two derivatives sloe and hillshade. Slope is the inclination at a given pixel an is given as an value from 0 to 90°.

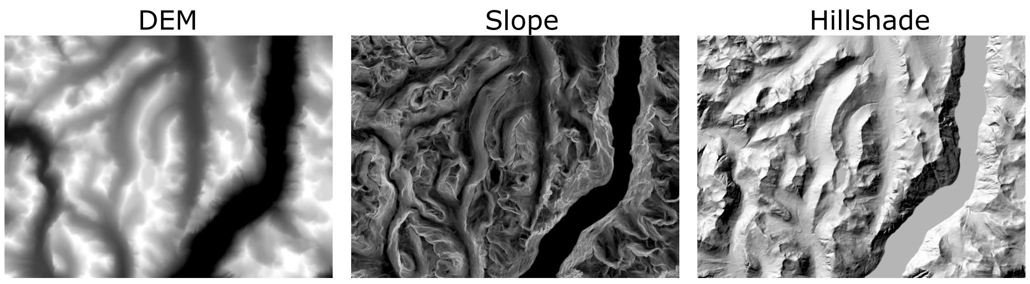


Figure 2.5: *Examples of a digital elevation model (DEM), slope map, and hillshade map. All from the same area. Created using ArcGIS Pro and 10 m DEM from hoydedata.no*

The data is normally captured with airborne sensors, from plane or helicopter. The traditional methods for DEM creation utilises photogrammetry in addition to field surveys (Ismail and Jaafar, 2013). Recent techniques includes Interferometric Synthetic Aperture Radar (InSAR) and LiDAR (Geological Survey of Norway, 2015). Elevation models for all of Norway are public and can be accessed from hoydedata.no. 1 m, 10 m, and 50 m resolution DEMs can be downloaded to a local computer freely for analysis.

2.2.2 RGB and NIR

Each colour is assigned to its separate channel, or band. Red is in the Sentinel-2 MSI assigned to the fourth band (B4), green to the third band, and blue to the second band. When displayed at a computer screen these three bands are displayed in their respective pixels on the screen, making the image appear natural. Near infrared (NIR) is in the eight band (B8).

2.2.3 Sentinel-1 Radar Data

SAR is described in section 2.1.2 and when referred to as training data it will be named VV an/or VH. VV and VH refer to the polarisation of the signal received (Moreira et al., 2013). Explaining the theory behind this is rather technical and beyond the scope of this study.

2.2.4 NDVI (Normalised Differential Vegetation Index)

NDVI is a commonly used dimensionless remote sensing index used to detect vegetation and indicate its health and vitality (GISGeography, 2021a). Identifying where vegetation is, and more important, isn't can be an indication of where a landslide has taken place (Lindsay et al., 2021).

Our eyes can see light that is in the visible part of the EM-spectrum, from red to green to blue. The chlorophyll in healthy vegetation absorbs more of the red and blue light and reflect more of the green light, which is why healthy plants appear green to us. But healthy vegetation also reflects wavelengths in the near-infrared (NIR) part of the spectrum, which our eyes can not see, but an remote sensor can (GISGeography, 2021a).

NDVI (Equation 2.1) is calculated by subtracting the value of the red band (B4) from the value of the near-infrared band (B8) and then dividing by the sum of those bands (GISGeography, 2021a). See table 2.1 for details on band names and wavelengths.

$$\mathsf{NDVI} = \frac{\mathsf{NIR} - \mathsf{RED}}{\mathsf{NIR} + \mathsf{RED}} \tag{2.1}$$

This calculation will generate a value between -1 and +1. The value will be close to +1 if the reflectance from the NIR-band is high while the reflectance from the red band is low. This indicates healthy vegetation. Conversely, the value will be close to -1 if the reflectance from the NIR-band is low while the red reflectance is high. This is an indication of no present vegetation (GISGeography, 2021a).

2.2.5 Change in NDVI (dNDVI)

By taking the NDVI values from one image at a specific date and subtracting them from another NDVI image taken at different date we get a difference NDVI image (dNDVI). This allows for detection of changes in vegetation between different dates. In the case of landslide detection it is common to refer to the time before a landslide event as *pre* and the time after a landslide event as *post*. From this dNDVI can be calculated as shown in Equation 2.2 (Lindsay et al., 2021):

$$\mathsf{dNDVI} = \mathsf{NDVI}_{post} - \mathsf{NDVI}_{pre} \tag{2.2}$$

Figure 2.6 shows how one such calculation will perform. The image to the left show the NDVI values from an image taken before the 2019 landslide event in Jølster. Bright pixels indicate healthy vegetation, while dark pixels indicate unhealthy or the absence of vegetation. The image in the middle shows the same NDVI calculation only done an image taken after the landslide event. Some landslides can clearly be seen along the lake in the bottom of the image. The image to the left show the calculated dNDVI values, and the landslides pop out more clearly as features that show low NDVI values in both pre- and post-images are cancelled out. It is therefore crucial that images taken before and after each landslide event are available in order to generate dNDVI images.

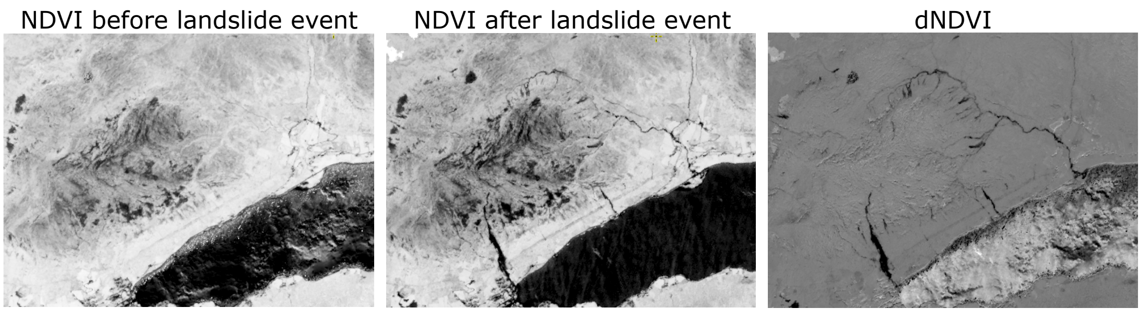


Figure 2.6: *NDVI images from the Jølster 2019 landslide event. Generated from GEE script provided at the courtesy of Erin Lindsay (script provided in section A.2)*

2.2.6 Corine Land Cover (CLC)

The Corine Land Cover (CLC) inventory can provide relevant information of the land cover in which landslides occur. CLC is a database of land cover and its changes, land use, vegetation state, water cycle and earth surface energy variables for European countries. The database is jointly implemented by the European Environment Agency (EEA) and the European Commission DG Joint Research Centre (JRC) (Büttner et al., 2021). Connecting landslides occurrences to the CLC inventory will make it possible to say if the landslide training samples show a bias towards a particular type of land cover.

2.3 CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional neural network (CNN) (Fukushima, 1980; LeCun et al., 1998; Krizhevsky et al., 2012) is a type of deep learning technique that has proven to be very successful in extracting information from images. It has even outperformed other conventional learning methods (Prakash et al., 2020). The concept of CNN dates back to the 1970s. However, the modern subject of CNNs was established in 1998 by LeCun et al. (1998) (Nilelsen, 2019, Deep Learning). It has since revolutionised the field of computer vision (Camps-Valls et al., 2021).

2.3.1 Neural Networks

To understand what a CNN is we first need to understand the concept of a *neural network*. In simple terms, a neural network is a technique for building a computer program that, in ways, resemble the way we think our human brain work, so that it can learn from data (Smilkov and Carter, 2022). The first approach at making a machine learn from external stimuli in a similar way we humans learn can be dated back to the 1950s and the *Perceptron* model, realised by Frank Rosenblatt (Rosenblatt, 1958). The Perceptron was designed as "A probabilistic model for information storage and organisation in the brain" (Rosenblatt, 1958), and represented the state of the art in neural networks until the mid-80s. It has since formed the basis of the complex neural networks of today (Ibañes, 2016).

The basis for a neural network are what is referred to as artificial neurons, or nodes—the terms are used interchangeably. The nodes are mathematical functions that takes an input, that is processed, and returns an output. Neural networks are comprised of several node layers, containing one or several nodes; one input layer, one or more hidden layers, and an output layer. Each node in one layer is connected to the nodes in the next layer, with associated weight and bias for each connection. A node can be seen as its own linear regression model, where input data, weights, and bias are put together to make one output. The output of one node is the input of the next node (IBM Cloud Education, 2020b).

To understand this further we will study a simple neural network that takes four inputs x_{11}, x_{12}, x_{21} and x_{22}, see Figure 2.7. This network has three layers; one layer with the four input neurons, one layer with two neurons that does some calculations using the weights w and the final layer contains the output neuron.

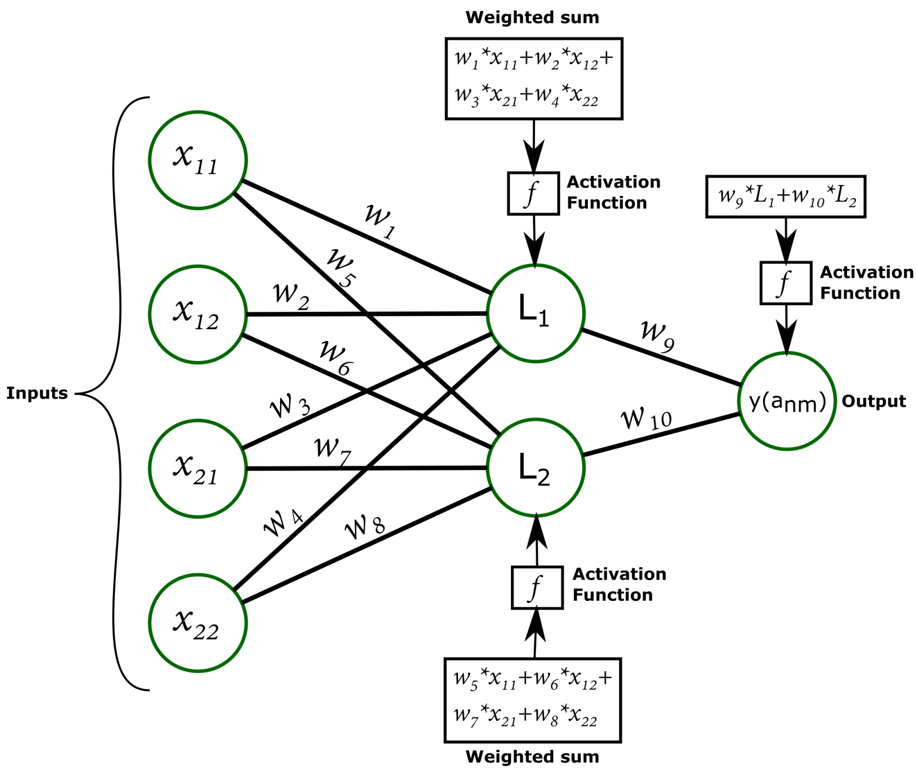


Figure 2.7: *Illustration of a very simple neural network with four input nodes, two calculation nodes and one output node. The activation function f for this illustration will be an arbitrary function that outputs a number that fits to the presented task.*

Consider a very simple and small image of 2x2 pixels. Figure 2.8 show such an image where two pixels are black, forming a diagonal feature. Each pixel has a value corresponding to the colour intensity (-1 for black and +1 for white). These values can be used as inputs to a neural network that has been trained at recognising diagonal features in 2x2 images. Figure 2.9 shows what the calculations might look like. Red lines indicate a negative weight, grey line indicate a weight of zero and black lines indicate a positive weight. The activation function is a function that at every node in the neural network takes in the value x computed from preceding nodes and weights. x can in theory be any number on the number line. The activation function then 'squishes'[1] the value of x down to fit in the range $[0, 1]$. Its purpose is to avoid extreme values in the calculations (Arnx, 2019). By studying the figure it can be understood how the final output layer gets the value of 1 and the 2x2 image is classified as "diagonal line". The activation function 'squishes' the number 2 down to 1.

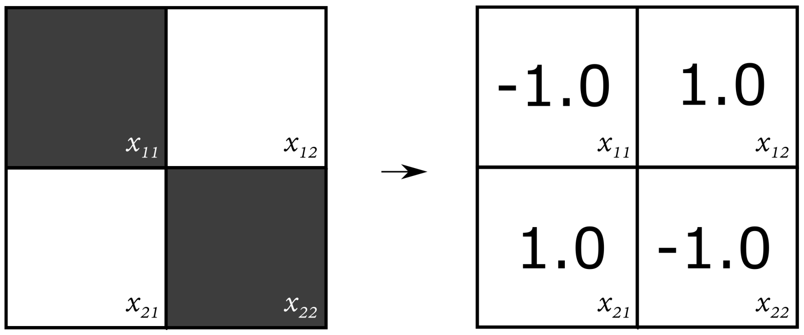


Figure 2.8: *Left: Image of size 2x2 pixels of a black diagonal feature. Each pixel in the image has a numerical value, in this case the value is a number between -1 and +1, which is illustrated on the right. A black pixel yields a value of -1 and white pixel a value of +1.*

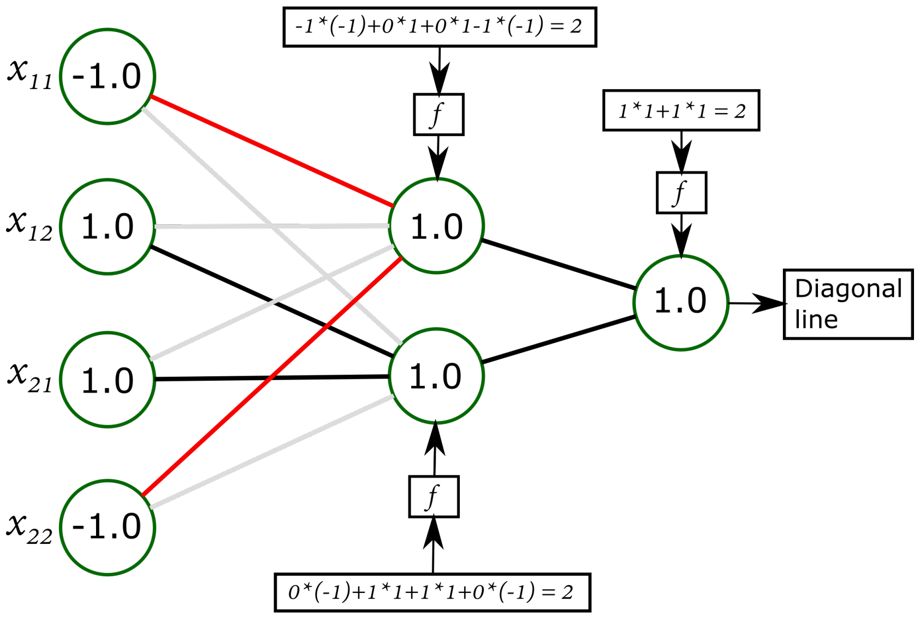


Figure 2.9: *Theoretical result from applying an image of a diagonal feature to a model trained at detecting diagonal features.*

One type of activation function is the sigmoid function, shown in Figure 2.10. Other activation functions exists with different limits, but the aim is still to limit the value of the neuron (Arnx, 2019).

What happens then, if we try to classify something that is not a diagonal feature? Figure 2.11 shows a 2x2 image of a black horizontal feature. By using the values of

[1] https://www.youtube.com/watch?v=aircAruvnKk

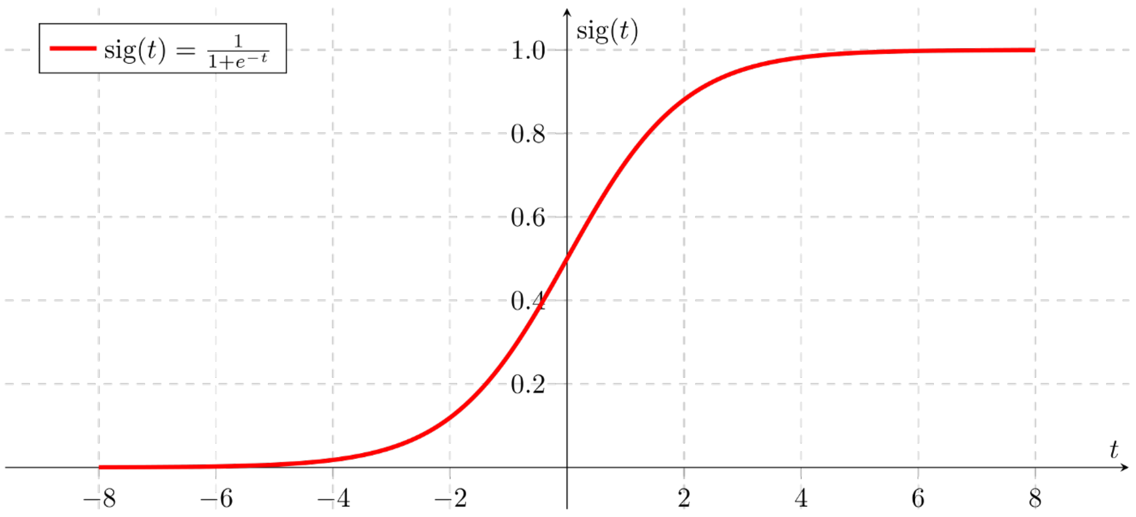


Figure 2.10: *Generic Sigmoid function that is often used as an activation function for neural networks. The function takes in any value t and outputs a number between 0 and 1. Figure fetched from Arnx (2019).*

each pixel as input to our model the output value will be different from the diagonal feature. Figure 2.12 show the same neural network, only with different inputs, and now the value in the final output layer is 0.7 and not 1.0. This feature is likely not a diagonal line.

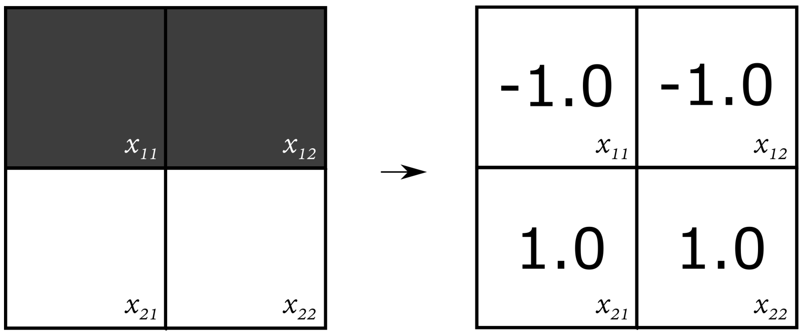


Figure 2.11: ***Left:*** *2x2 pixel image of a black horizontal feature.* ***Right:*** *The pixels' corresponding numeric values.*

This simple network has some flaws and is not trained well enough and will only classify diagonal features that start in the top left pixel, but it still helps in conveying the idea behind a neural network. Neural networks and deep learning tend confusingly to be used interchangeably, but they are not the same. The "deep" in deep learning simply refer to the number of layers in the neural network. A neural network that consist of more than three layers would be considered a deep learning network (IBM Cloud Education, 2020b).

2.3.2 Convolutional Neural Networks

A CNN is to an extent the same as a neural network except that CNNs show superior performance in image recognition compared to other neural networks. A CNN contains what is called a *convolutional layer*. In essence, the convolutional layer identifies different parts of a feature in an image. It does this by moving a small digital filter of

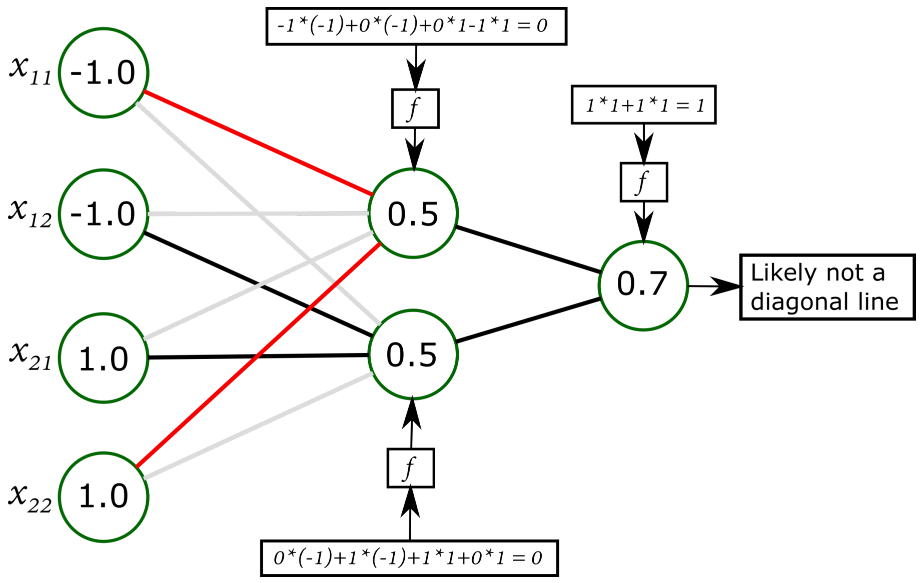


Figure 2.12: *Theoretical result from applying an image of a horizontal feature to a model trained at detecting diagonal features. Weights with a value of 1 are represented by black lines, weights with a value of 0 are represented by grey lines, and -1 by red lines. In reality, these weights will not be exact whole numbers. This illustration does not show the bias that is also added at each node.*

typically a 3x3 matrix across the image performing several *convolutions* (IBM Cloud Education, 2020a). A convolution is a mathematical term expressed as "an integral that expresses the amount of overlap of one function g as it is shifted over another function f" (Weisstein, 2022).

These features are then sent on to the next layers for more calculations. Eventually it ends up in the final layer, where classification is performed based on the features extracted through the previous layers and their different filters (IBM Cloud Education, 2020a).

2.3.3 Applications of Deep Learning for computer vision

Within DL for computer vision there a number of applications. This section is summarised from Esris page on applications of deep learning. The most common ones are the following (all images are from Esri (2021d)):

- **Image classification:** This type of DL algorithm will assign a label to each image. In the example in Figure 2.13 the image on the left might be labelled *crowd* and the image on the right might be labelled *cat* (Esri, 2021d).

Figure 2.13: *Example of image classification in deep learning (Esri, 2021d).*

- **Object detection:** This type of DL algorithm involves locating and describing a bounding box for a feature in an image. This might be locating a plane from a remote sensing image or detect different pets in an image as shown in Figure 2.14. This is useful for locating specific objects in remote sensing images and mark them on map (Esri, 2021d).

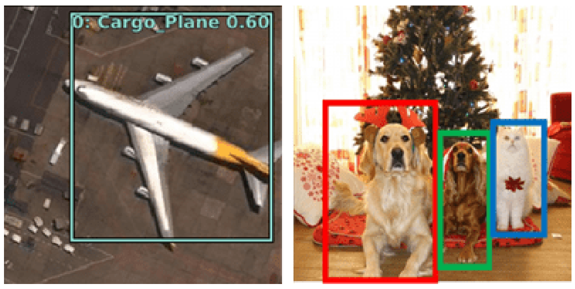


Figure 2.14: *Example of object detection in deep learning (Esri, 2021d).*

- **Semantic segmentation (Pixel classification)** This type of DL algorithm classifies each pixel in an image as belonging to a class. In the example to the left in Figure 2.15 road pixels are classified as belonging to a separate class from non-road pixels. On the right, pixels belonging to a cat are classified as *cat*. Other pixels in the image are given different classes depending on what the model has been trained to recognised (Esri, 2021d).

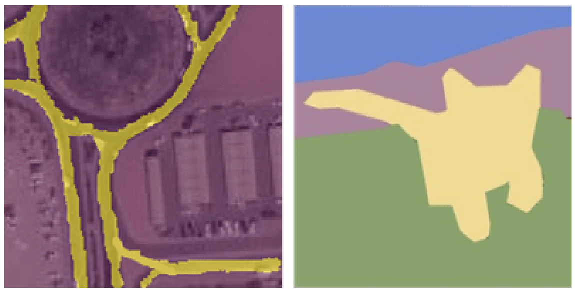


Figure 2.15: *Example of semantic segmentation (pixel classification) in deep learning (Esri, 2021d).*

- ***Instance segmentation*, or *object segmentation:***
  Instance segmentation, also known as object segmentation, detects and draws the boundary of each desired object. This makes it a more precise object detection method. For example, in the remote sensing image on the left in Figure 2.16, each house is recognised as its own object. In addition, the exact outline of the

roof shape is drawn. In the image on the right, the distinct shapes of cars are distinguished from the ground (Esri, 2021d).

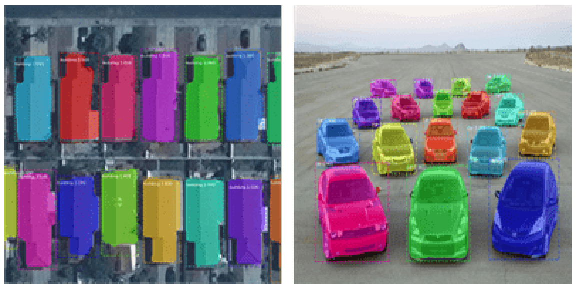


Figure 2.16: *Example of instance segmentation in deep learning (Esri, 2021d).*

- **Image translation** Image translation takes one possible representation or style of an image and translate it to another. Examples might be noise reduction or super-resolution as shown in the Figure 2.17. The low-resolution image on the right is translated to an image with higher resolution using a super-resolution model (Esri, 2021d). Another task might be to generate a thematic map from satellite images (Ingale et al., 2021).

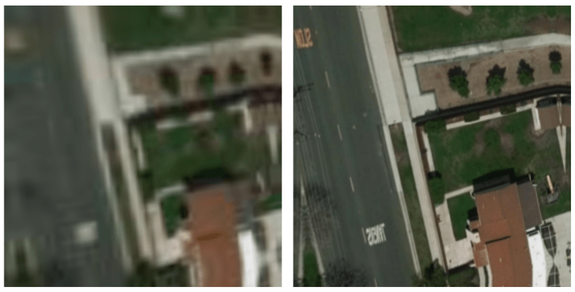


Figure 2.17: *Example of image translation in deep learning (Esri, 2021d).*

- **Change detection** This type of DL algorithm can look at two images from the same place but at different times and detect changes in features of interest between them. The image to the left shows housing development, the middle image shows the same development five years later, and the image on the right shows a logical change map where new homes have been detected and shown in white (Esri, 2021d).

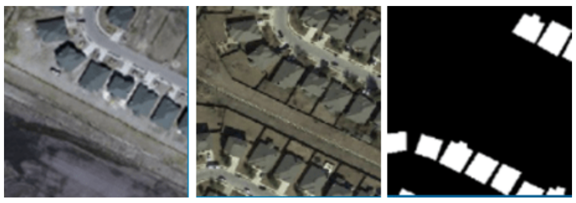


Figure 2.18: *Example of change detection in deep learning (Esri, 2021d).*

2.3.4 Transfer Learning

Computer vision is a field that has been researched on a large scale (Camps-Valls et al., 2021). A number of models have been trained on large data sets to make them tasks and made public. Anyone can use these generic models to train on their own data sets. This is called Transfer Learning. Transfer Learning for machine learning is, according to Seldon.io (2021), "...when existing models are reused to solve a new challenge or problem." It is a technique used while training models and not a type of machine learning algorithm. The original model requires a high level of generalisation to be able to fit to the new unseen data. Transfer learning allows for faster training times and saves computer resources. Models do not have to be trained from scratch, as knowledge from one model can be transferred to e new model. This kind of knowledge might be parts of the model that identifies the edge of an object or parts that identifies different textures in an image (Seldon.io, 2021).

2.3.5 Overfitting

Overfitting is according to IBM Cloud Education (2021) "...when a statistical model fits exactly against its training data." This is an unwanted situation as an overfitted model is not generalised enough to predict and classify data it has not been trained on. This prevents the model from being used as intended. This can occur either if the model has been trained for too long on the sample data or when the model is too complex. It can start to see the 'noise,' or other irrelevant information as important parts of the data set (IBM Cloud Education, 2021).

Underfitting is another undesired phenomena related to the training time or model complexity. This occurs conversely when the model has not been trained long enough on a data set or the data , and a meaningful relationship between the input and output variables can not be found (IBM Cloud Education, 2021). This is illustrated to the left in Figure 2.19 where the trained model (red line) does a poor work of fitting the training data set. To the right a case of overfitting is illustrated, and in the middle, an optimal trained model is shown.

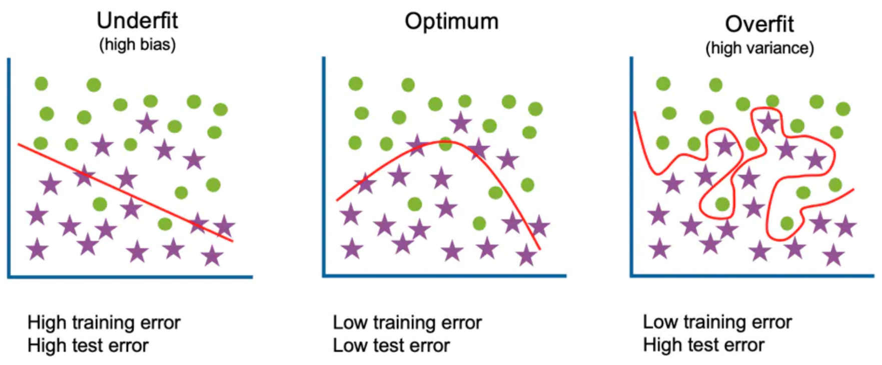


Figure 2.19: *Illustration from IBM Cloud Education (2021)*

There are a number of techniques to avoid overfitting according to IBM Cloud Education (2021). The most relevant of them are:

- **Early Stopping:** By stopping the training early, before the model starts to learn the noise and irrelevant information, overfitting can be reduced.
- **Train with more data:** The accuracy of the model can be increased by introducing more training data. This is more effective when the added data is relevant

and not too noisy. Otherwise, the complexity of the model will increase and it will overfit.

- **Data augmentation:** Generate new data from the training data by adding noise or distort the images slightly, either by rotation or by stretching the images.
- **Dimensionality reduction:** More data means more complexity and thus, greater tendencies to overfitting. Reducing the size of the data set, i.e. the number of attributes, features or input variables, will prevent the model from overfitting (Pramoditha, 2021).

2.3.6 Model Inferencing

Once a model has been trained it needs to be applied to a problem we want to investigate. It is said that the model is inferenced (Esri, 2021b). In ArcGIS Pro there are mainly two parameters that can be set when inferencing a model: *Padding* and *batch*. The padding refers to a band of extra pixels with that is added outside of the image when inferencing. The padding is added to use the pixels in the corners of an image tile in more convolutions. The padding should be adjusted to the tile size of the images that the model has been trained on.

2.3.7 Metrics for model performance

In the case of object detection where only *one* class of object is accounted for two possible outcomes can take place when applying a DL model: either the model predicts the object class to exist in a particular place in the image (positive prediction), or it does not make a prediction of the object in a particular place in the image (negative prediction) (Google Developers, 2022). In the case of landslide prediction *Landslide* is a positive class and *Non-landslide* is a negative class. The different outcomes of a classifier is presented below (Google Developers, 2022):

- A True Positive (TP or T_p) is when the model correctly predicts the positive class we are looking for (landslide).
- A True Negative (TN or T_n) is when the model correctly predicts the negative class (non-landslide)
- A False Positive (FP or F_p) is when the model wrongly predicts the *negative* class to be positive (non-landslide predicted to be landslide).
- A False Negative (FN or F_n) is when the model wrongly predicts the *positive* class to be negative (landslide predicted to be non-landslide).

Metrics to evaluate the DL model can be derived from these four outcomes. In the following metrics also used in Prakash et al. (2020) will be presented.

Accuracy is defined as the ratio between the number of correct predictions and the the total number of predictions (Equation 2.3):

$$\mathit{Accuracy} = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \tag{2.3}$$

Since the data sets in object detection often are imbalanced, this metric is not very convenient. The positive class is often the minority and the negative class the majority. The metric take into account the True Negatives. For a given image the object (positive feature) will often be outnumbered by the non-object (negative features). E.g. if the model predicts only negative values in a image of a landslide, TN will be far greater than TP, resulting in a high score for accuracy even though the landslide was not detected.

Precision gives a value of how many of the predicted positives are actual positives and is the ratio of correct positive predictions to all the positive predictions, including the incorrect predictions (Equation 2.4):

$$\textit{Precision} = \frac{T_p}{T_p + F_p} \tag{2.4}$$

Recall (probability of detection) is defined as the ratio between the correct positive predictions and all the actual positives (Equation 2.5):

$$\textit{Recall} = \frac{T_p}{T_p + F_n} \tag{2.5}$$

F1

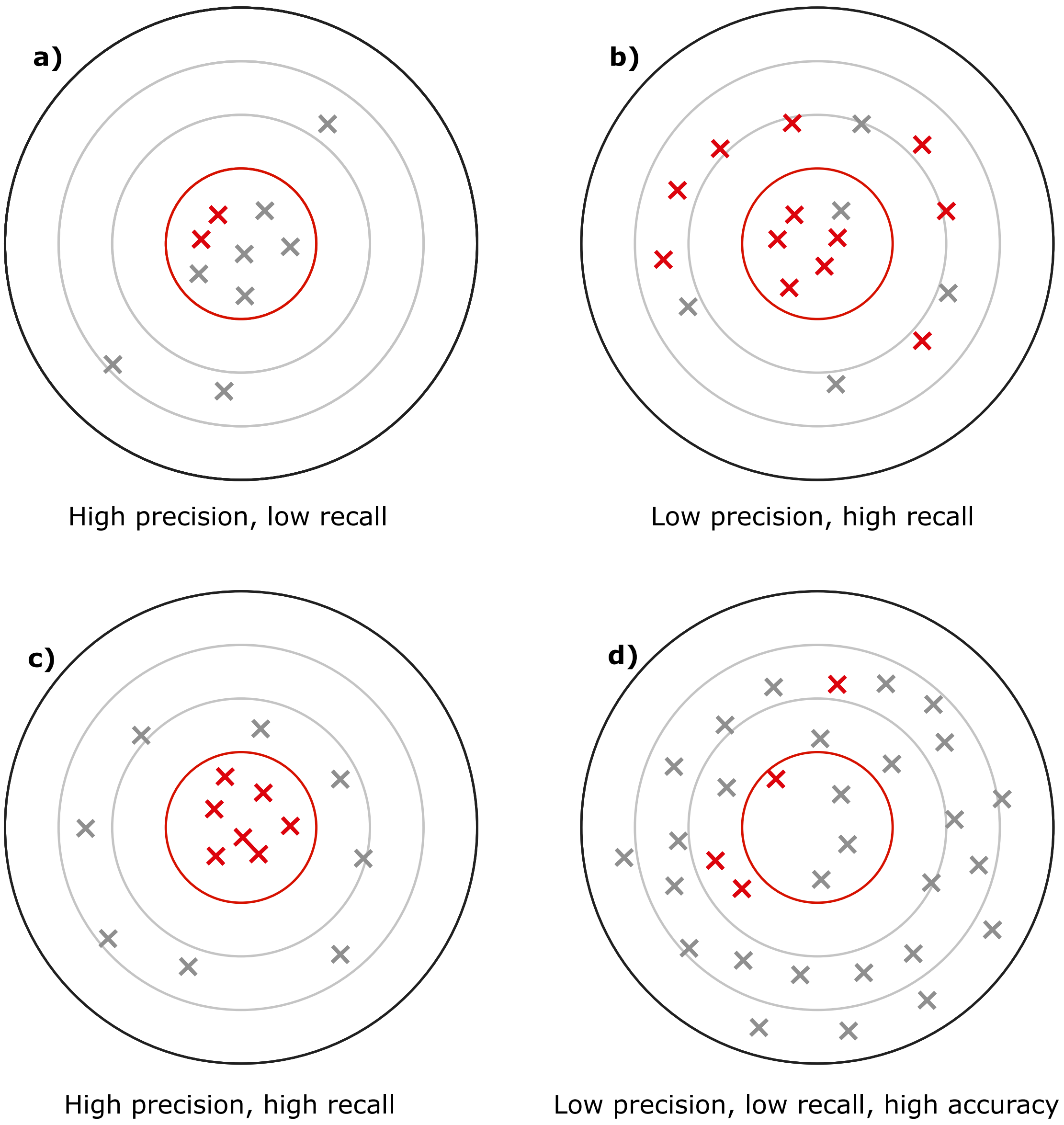
$$\textit{F1} = \frac{2 * (\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}} = \frac{2 * T_p}{2 * T_p + F_p + F_n} \tag{2.6}$$

Matthews corrolation coefficient (MCC)(Equation 2.7), named after Brian Matthews as it was first used in Matthews (1975) (Baldi et al., 2000), is a coefficient that is commonly used to assess the corrolation between two variables. If two variables are independent of each other, then their MCC is 0.

$$\textit{MCC} = \frac{T_p * T_n - F_p * F_n}{\sqrt{(T_p + F_p)(T_p + F_n)(T_n + F_p)(T_n + F_n)}} \tag{2.7}$$

The priority in civil protection purposes, such as landslide detection, is to minimise the number of missed detections (FN) and secondary to limit false detections (FP) (Brunetti et al., 2018), thus precision and recall needs to be maximised.

Figure 2.20 illustrates some predictions a DL model might make. Let the inside of the red circle be the actual positive class and the red crosses be the positive prediction. And let the outside of the red circle be the actual negative class and the grey crosses be the negative predictions. Then the TP are red crosses inside the circle, FP are red crosses outside the circle, TN are grey crosses outside the circle, and FN are grey crosses inside the circle. The figure also indicates what relative precision and recall one would expect in each case a)—d).


a)
High precision, low recall
b)
Low precision, high recall
c)
High precision, high recall
d)
Low precision, low recall, high accuracy

	✕ **Positive** prediction	✕ **Negative** prediction
Inside red circle	True Positive	False Negative
Outside red circle	True Positive	True Negative

Figure 2.20: *Illustration of some scenarios from a CNN model and its relative evaluation metrics.* ***a)*** *show a model that has high precision, but low recall value. This model is very conservative in its predictions.* ***b)*** *model that has low precision, but high recall. This model is excessive in its predictions.* ***c)*** *"perfect" model that makes only correct predictions and consequently the precision and recall values are both high. Figure* ***d)*** *show how accuracy is not a very useful metric in object detection, as most of a model's prediction will be True Negatives. This makes accuracy higher than precision and recall as those metrics do not consider TN.*

CHAPTER 3

METHODOLOGY

In this chapter the methodology to address the research questions presented in section 1.2 will be presented. Specifically we here look at the first sub-research question, “How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?”

3.1 VERIFICATION OF REPORTED LANDSLIDES

In order to get the desired images of the landslides to use as training data a number of steps must be gone through. First step is to filter through a database of reported landslides[1]. The database organises all the reported observations that are reported by users through the online service regObs. RegObs is an inventory service, consisting of observation of rockfalls, floods and landslides, as well as snow condition and snow avalanches, and ice cover conditions. When registering an event the user is asked to specify the date of occurrence/observation, a coordinate reference, pictures of the observation and, if available, a link to other sources, e.g. a news report (Fjeld, 2018).

The inventory is quite extensive, but not all reported landslides events will be suitable as training data, so the desired landslides has to be filtered out based on a set of requirements. Figure 3.1 show what parameters the filtering is based on. The filtering requirements are:

1. **Landslide must be larger than 1000 m^2:** Smaller landslides are difficult to detect from 10 m resolution satellite images.

2. **Landslide has to be debris slide:** The database of reported events contains events from all kinds of landslides and avalanches. This includes small rockfalls and small rotational slides that have little influence on the reflectance in satellite images. Clay slides are in this instance accepted as a type of debris slide.

3. **Landslide must be after 2017:** Sentinel-2 data is only available after March 2017. Under some circumstances it might be necessary to use images from the year before a landslide event to create pre-event composite images. For this

[1] www.skredregistrering.no

reason only landslides after 2017 are selected as to ensure that post-images will be available for all landslides.

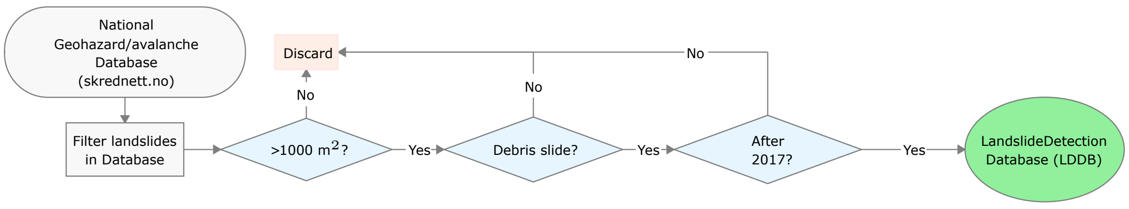

National Geohazard/avalanche Database (skrednett.no)
Filter landslides in Database
Discard
No
>1000 m^2?
Yes
No
Debris slide?
Yes
No
After 2017?
Yes
LandslideDetection Database (LDDB)

Figure 3.1: *Decision tree for filtering for potential landslides to use in the Landslide Detection Database (LDDB).*

The landslides in the database of reported landslides that meet the requirements are moved to a separate Landslide Detection Database (LDDB). This database will be brought on to the next step that is the verification of the LDDB. Figure 3.2 show the process for verification

Landslide identifier

Each landslide in the LDDB is assigned an arbitrary number to keep track of all the different landslides in the database. This number is named LS_ID or *landslide identifier* and will be used frequently to refer to the landslides in the training data set.

3.2 IMAGE PRE-PROCESSING

It is very time consuming to acquire and pre-process the relevant images for the purpose of landslide detection. But it is quite important that it is done in a appropriate manner so that the quality and relevance of the data is satisfactory. It is also important that the processing steps are made as uncomplicated as possible so that it is easy to replicate the process, should it be necessary to do things over. The method for image acquisition and pre-processing presented in section 3.2.1 is to a large extent based on the work by Lindsay et al. (2022). Particular the methods involving Google Earth Engine GEE is mostly based on Lindsay's work.

3.2.1 Google Earth Engine (GEE)

GEE is cloud-based platform that allows for viewing and analysis of satellite imagery on a large scale. The possible analysis includes change detection, mapping of trends and quantification of differences on the Earth's surface (Gandhi, 2021).

An extensive library of remote sensing data are stored in the Google Earth Engine cloud and can be accessed and manipulated by the user using a code editor that is an Integrated Development Environment IDE for Earth Engine Javascript API. The API contains many pre-made functions and libraries that allows for time saving pre-processing (Gandhi, 2021).

By uploading a shape file of the verified LDDB, image tiles of 2500x2500 meters for each LS_ID can be downloaded. Scripts for acquiring image tiles are provided at the courtesy of Erin Lindsay (see appendix A). The image tiles are downloaded as three sets. One set with pre-event images, one set with post-event images, and one set

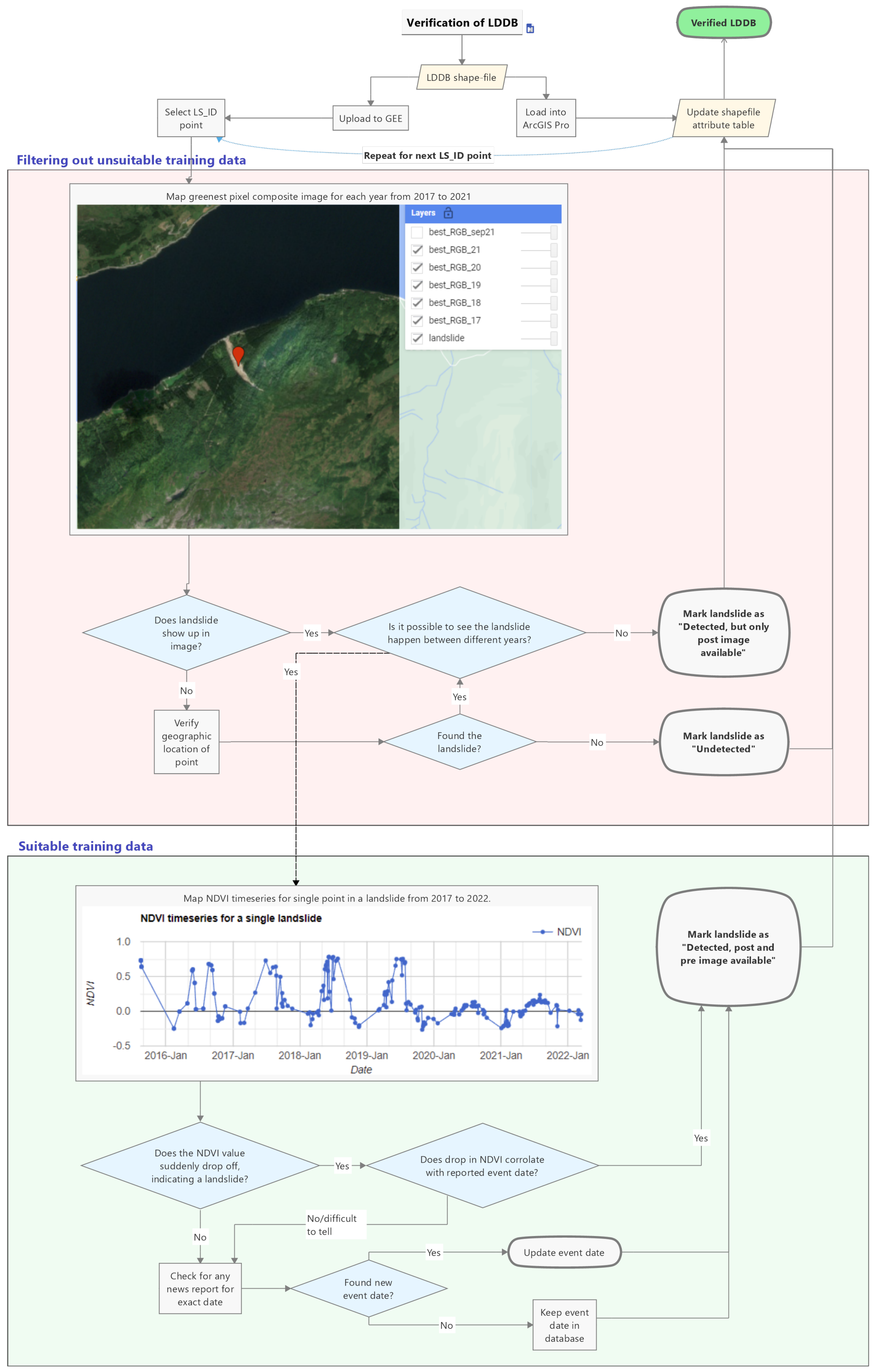

Verification of LDDB
Verified LDDB
LDDB shape-file
Upload to GEE
Load into ArcGIS Pro
Select LS_ID point
Update shapefile attribute table
Repeat for next LS_ID point
Filtering out unsuitable training data
Map greenest pixel composite image for each year from 2017 to 2021
Layers
best_RGB_sep21
best_RGB_21
best_RGB_20
best_RGB_19
best_RGB_18
best_RGB_17
landslide
Does landslide show up in image?
Yes
Is it possible to see the landslide happen between different years?
No
Mark landslide as "Detected, but only post image available"
Yes
No
Yes
Verify geographic location of point
Found the landslide?
No
Mark landslide as "Undetected"
Suitable training data
Map NDVI timeseries for single point in a landslide from 2017 to 2022.
NDVI timeseries for a single landslide
NDVI
1.0
0.5
0.0
-0.5
2016-Jan
2017-Jan
2018-Jan
2019-Jan
2020-Jan
2021-Jan
2022-Jan
Date
Mark landslide as "Detected, post and pre image available"
Does the NDVI value suddenly drop off, indicating a landslide?
Yes
Does drop in NDVI corrolate with reported event date?
Yes
No/difficult to tell
No
Yes
Update event date
Check for any news report for exact date
Found new event date?
No
Keep event date in database

Figure 3.2: *Process for verifying Landslide Detection Database.*

with difference-images ($post - pre$). In each set bands featuring S1-radar data (SAR), S2 optical images images and DEM are downloaded.

Cloud filtering

Using optical data for mapping landslides can have serious limitations in the presence of cloud cover (Nava et al., 2021). It is therefore necessary to reduce the influence of cloud cover on the images that are acquired.

To get cloud free images it is in GEE possible to select the images with the least cloud cover. As described in section 2.1.3 the Sentinel satellites have a revisit time of 5 days. This means that for a 1-month period an average of 6 images are taken of the same place. If the sky is clear on one or more of those revisits, it is possible to select the images that have the least amount of cloud cover and use those for landslide detection. This assumes that at least one image in the 1-month period is cloud free, if not, you get images containing clouds (Erin Lindsay 2022, personal communication, 21 June). The "degree of cloudiness" or *Cloud Percentage* for an image is an attribute determined by an algorithm that is applied to all the Sentinel-2 data that is made publicly accessible. This algorithm is created by the European Space Agency (ESA).

3.2.2 ArcGIS Pro

The use of ArGIS Pro forms the foundation of this thesis' topic. ArcGIS Pro enables the use of advanced deep learning algorithms for GIS and remote sensing applications (Esri, 2021b). The author will claim that an intuitive user interface make the process of applying complex DL algorithms to a remote sensing problem feasible for someone without a background in computer science. Figure 3.3 shows the workflow for processing the image tiles downloaded from GEE, so that proper training data can be generated. The most important parts of the workflow will be presented in the text, refer to Figure 3.3 for the full workflow.

Merging of image tiles

Once all the separate image tiles corresponding to each landslide in the verified LDDB have been downloaded from GEE, they need to be combined to the same raster file. This will make it easier to extract the different bands and combine them to create training data with different input data. This is done to answer the second research question: *What input data will be best suited for detecting landslides?*.

For some few cases it might be that two or more landslides in the same area have occurred at different times. If image tiles of these landslides are included in the same merging sequence a described above, one tile might overlap the other and only the landslide of the topmost tile will be included in the training data set. This is most critical if difference images such as dNDVI are used. If such a case happens it is necessary to clip one or more image tiles so that no landslide is obscured by another tile. This can be done using the *Clip* raster function.

Model Types

In Table 3.1 the different model types are listed. Along the model type is also what input data goes into each type. The input data is simply added as a raster band to a composite raster. This is what will be used to determine what input data will be best suited to detect landslides.

Table 3.1: *Input data to each model type. Each colour in RGB requires its own separate band.*

Model Type	Input Data
1a	B1: dNDVI
1b	B1: dNDVI B2: DEM
1c	B1: dNDVI B2: Slope
1d	B1: dNDVI B2: DEM B3: Slope
2	B1: dVV B2: dVH B3: dNDVI B4–B6: RGB_post B7: NIR_post B8: DEM
3	B1: dNDVI B2–B4: RGB_post B5: NIR_post B6: Slope
4	B1: VV_pre B2: VH_pre B3: VV_post B4: VH_post B5: Slope

Projection

It is important to change all the raster data sets too the same projection. This will ensure that the pixel size for all the tiles are the same, as it was discovered that pixels were elongated in the wrong projection. For this study the ETRS 1989 UTM Zone 33N projection has been selected. The training data set presented in chapter 4 is scattered around all of Norway, from Hordaland to Finnmark. The data set is therefore covered by the UTM zones 32, 33, 34, and 35 (Mæhlum, 2021). Since it is confusing and troublesome to deal with different projections in the same project UTM 33N has been selected as the only projection for this study. This will also ensure that the image pixel aspect ratio remains 1:1.

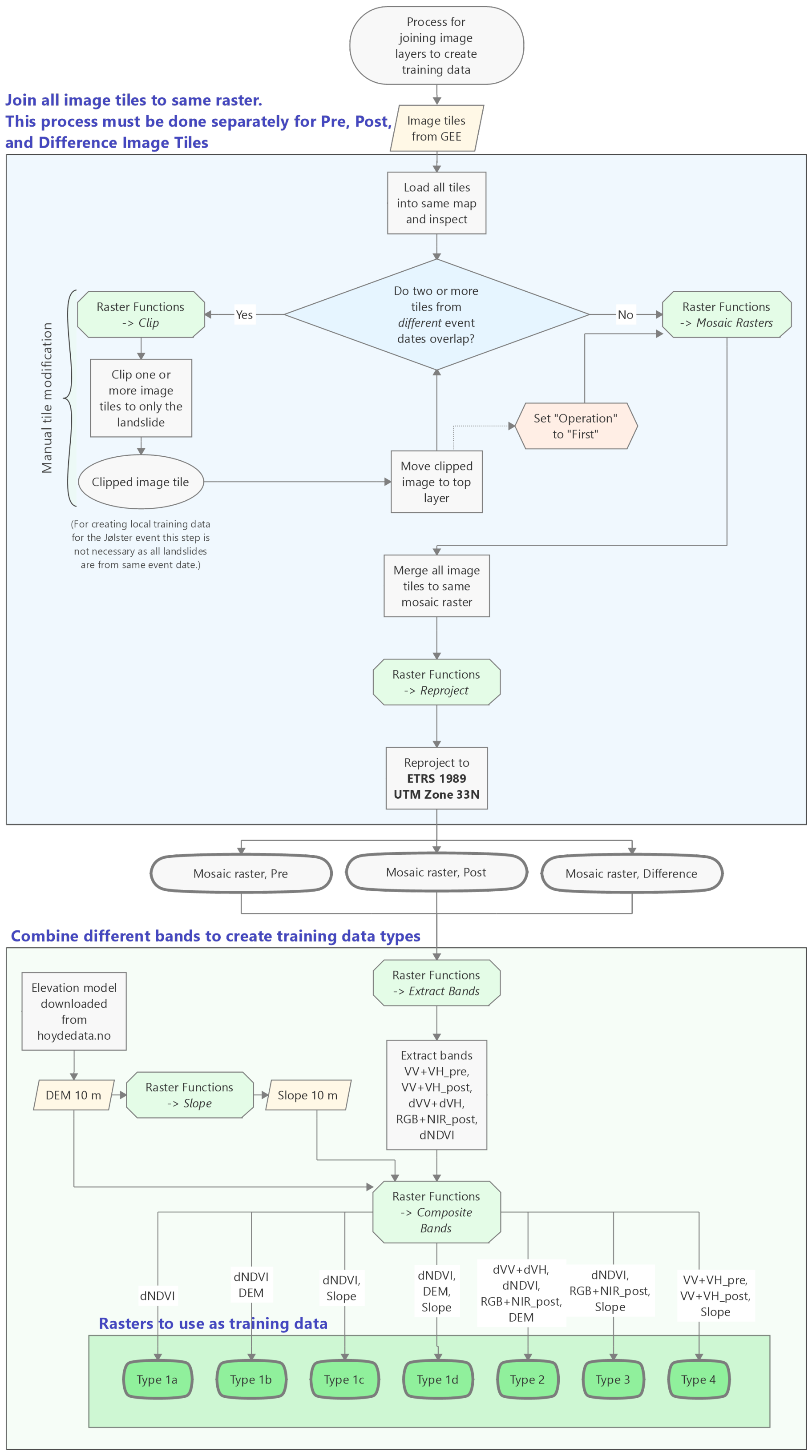

Process for joining image layers to create training data
Join all image tiles to same raster.
This process must be done separately for Pre, Post, and Difference Image Tiles
Image tiles from GEE
Load all tiles into same map and inspect
Do two or more tiles from *different* event dates overlap?
Yes
No
Raster Functions -> *Clip*
Raster Functions -> *Mosaic Rasters*
Manual tile modification
Clip one or more image tiles to only the landslide
Clipped image tile
Set "Operation" to "First"
Move clipped image to top layer
(For creating local training data for the Jølster event this step is not necessary as all landslides are from same event date.)
Merge all image tiles to same mosaic raster
Raster Functions -> *Reproject*
Reproject to **ETRS 1989 UTM Zone 33N**
Mosaic raster, Pre
Mosaic raster, Post
Mosaic raster, Difference
Combine different bands to create training data types
Raster Functions -> *Extract Bands*
Elevation model downloaded from hoydedata.no
DEM 10 m
Raster Functions -> *Slope*
Slope 10 m
Extract bands VV+VH_pre, VV+VH_post, dVV+dVH, RGB+NIR_post, dNDVI
Raster Functions -> *Composite Bands*
dNDVI
dNDVI DEM
dNDVI, Slope
dNDVI, DEM, Slope
dVV+dVH, dNDVI, RGB+NIR_post, DEM
dNDVI, RGB+NIR_post, Slope
VV+VH_pre, VV+VH_post, Slope
Rasters to use as training data
Type 1a
Type 1b
Type 1c
Type 1d
Type 2
Type 3
Type 4

Figure 3.3: *Process for processing downloaded image tiles. The "Raster Functions" refer to the tools to use in ArcGIS Pro. Green boxes represent different tools in ArcGIS Pro, blue boxes represent decisions, yellow input data, light red input parameters, and grey boxes processes.*

3.3 Deep Learning

Refer to Figure 3.4 for the workflow for creating image samples to use as training data for the Deep Learning geoprocessing tools in ArcGIS Pro.

One important note is that the 'backbone model'-setting in the 'Train Deep Learning Model' tool sets the model to use for transfer learning as described in section 2.3.4. The backbone model used in the training of the models in this thesis is ResNet50 (He et al., 2015), one of the most popular and most successful deep learning models so far (Shakhadri, 2021).

A process that will not be described here is the inferring of the models. Due to time constraints, the appropriate flowcharts have not been created, but the reader is encouraged to look to the Esri-tutorial in https://learn.arcgis.com/en/projects/classify-mangroves-using-deep-learning/ for the process of applying a classifier on new data. The output from model inferring is a shape file with georeferenced polygons of predicted landslides. Along with the polygons are confidence score telling how certain the model is on its prediction on a scale from 0–100 %.

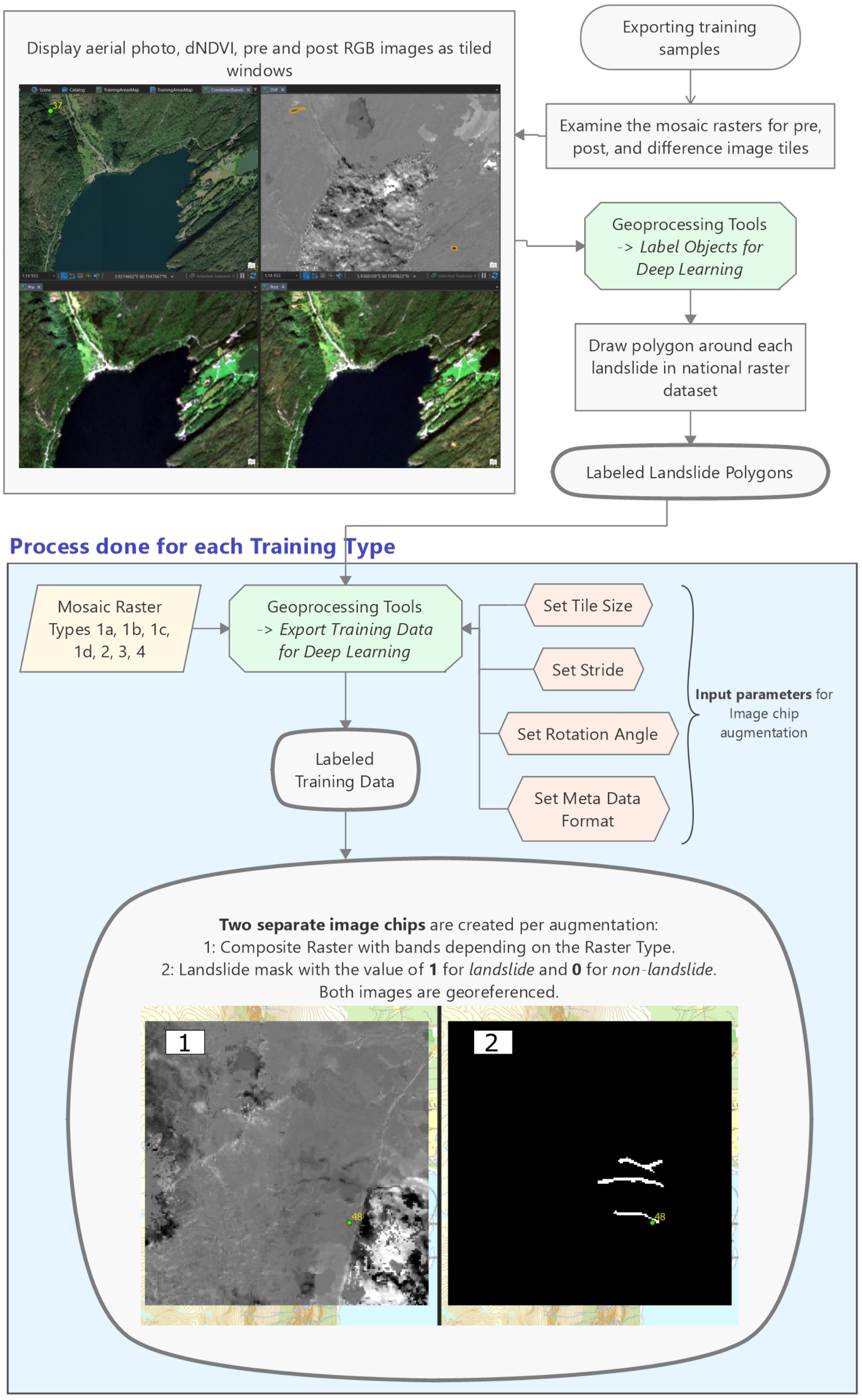

Exporting training samples
Examine the mosaic rasters for pre, post, and difference image tiles
Display aerial photo, dNDVI, pre and post RGB images as tiled windows
Geoprocessing Tools -> *Label Objects for Deep Learning*
Draw polygon around each landslide in national raster dataset
Labeled Landslide Polygons
Process done for each Training Type
Mosaic Raster Types 1a, 1b, 1c, 1d, 2, 3, 4
Geoprocessing Tools -> *Export Training Data for Deep Learning*
Set Tile Size
Set Stride
Set Rotation Angle
Set Meta Data Format
Input parameters for Image chip augmentation
Labeled Training Data
Two separate image chips are created per augmentation:
1: Composite Raster with bands depending on the Raster Type.
2: Landslide mask with the value of **1** for *landslide* and **0** for *non-landslide*.
Both images are georeferenced.
1
2

Figure 3.4: *Process for exporting training samples to use in deep learning in ArcGIS Pro.*

3.3.1 Confusion matrix map (CMM)

A confusion matrix map (CMM) is a means of visually displaying the predictions made by a classifier. All the possible outcomes for the predictions (TN, FN, FP, TP) described in section 2.3.7, will be visualised.

For calculating a CMM it is first necessary to convert the detected landslide polygons and the ground truth polygons to raster. The raster cells contained within a polygon are given the value of 1 and all other cells are given the value 0. By adding the raster of predicted landslides two times to the raster of ground truth landslides, each cell will have a value of either 0, 1, 2, or 3; 0 = TN, 1 = FN, 2 = FP, and 3 = TP. This technique was learned from Erin Lindsay on 5 April 2022, and eventually implemented in ArcGIS Pro by the author, using the geoprocessing tools *Polygons to Raster*, *Reclassify*, and *Raster Calculator*. They were assembled in the same workflow using the ModelBuilder for easy replication (Figure 3.5).

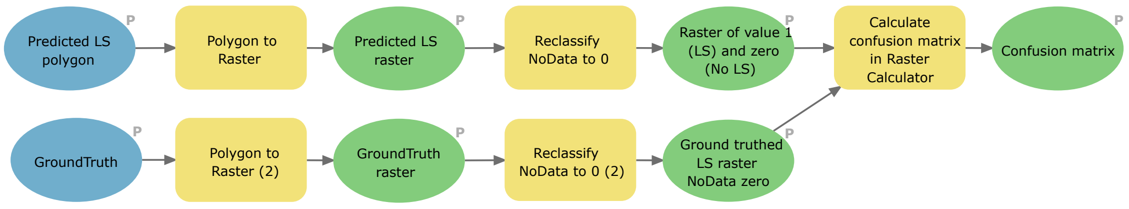

Predicted LS polygon
Polygon to Raster
Predicted LS raster
Reclassify NoData to 0
Raster of value 1 (LS) and zero (No LS)
Calculate confusion matrix in Raster Calculator
Confusion matrix
GroundTruth
Polygon to Raster (2)
GroundTruth raster
Reclassify NoData to 0 (2)
Ground truthed LS raster NoData zero
P

Figure 3.5: *ArcGIS Pro ModelBuilder chart for creating confusion matrix maps (CMM).*

CHAPTER 4

RESULTS

The results presented here will help answer the main research question *How can Deep Learning be applied to detect landslides from satellite images?* Each sub-research question will be addressed in its separate section (section 4.1, section 4.2 and section 4.3.

All the models are trained on the same hyperparameters. The only variable that is altered from model to model is the input data. The models are trained using Mask R-CNN[1] object detection model type (He et al., 2017). The results are split into three sets or cases:

A. Models trained on national data set (presented in section 4.2)

B. Models trained on local data set (presented in section 4.3)

C. Models trained on local data set but verified against the data it has been trained on (presented in section 4.3)

Models in B and C are the same DL models, but inferred on different landslide samples.

The local data set consist of 56 out of the 120 mapped landslides in Jølster. The area containing the landslides used for the local data set is represented in red tint in the CMMs in the figures in section 4.3. The area containing the 64 landslides the model has not been trained on is represented in blue tint in the same figures.

Models trained on local training samples have been trained on the same parameters as the models trained on the national data set, except for the *rotation angle* set when exporting the training samples for the local data set. The rotation was set to 30° for the national data set, while it was set to 90° for the local data set. This means that the local training samples are augmented with a rotation of 90° per stride, i.e. four images are created per stride. This means that the models trained on the local data set have been trained on *fewer* image samples.

[1] https://github.com/matterport/Mask_RCNN

4.1 WORKFLOW

This section aims at presenting the answer to the first sub-research question: *"How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?"* The answer to this question is described in the methodology chapter (chapter 3).

The workflow has been established by a means of trial and error in the exploratory works leading up to the final models and results presented in section 4.2 and section 4.3.

4.2 INPUT DATA

This section aims at presenting the answer to the second sub-research question: *"What input data will be best suited for detecting landslides?"* It will first present the results from the verification of reported landslides. This is the first step in getting the input data needed. Then the geologic distribution of the training samples will be presented to assess possible bias toward specific categories. Next, the parameter values selected for the DL process will be explained and presented. Finally, confusion matrix maps (CMM) and performance metric score for each of the seven models trained on the training types introduced in the methodology chapter (Table 3.1).

4.2.1 Verification of reported landslides

By following the filtering criterion presented in section 3.1 (landslides after 2017, area larger than 1000 m^2, and classified as debris slides), a total number of **74** landslides events were selected and moved to the Landslide Detection Database (LDDB) for further verification. Following the method for manual verification of the LDDB presented in Figure 3.2, these 74 events have been deemed *detected*, *detected, but only for post image events* and *not detected*. The verification was done using GEE by filtering for satellite images before and after the reported event date for each landslide. In addition a dNDVI timeseries was generated for a point within the landslide for possible detection of the date when NDVI dropped.

See Table B.1 in appendix B for the full table containing the verification The results from the verification is summarised in the map in Figure 4.1. A full page map can be found in appendix C. This in part help answer the first sub-research question "*How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?*"

Some interesting remarks from the manual verification process:

1. Even though some reported landslides are visible in the satellite image, many of the landslides follow older landslide scars. This means that there is little to no difference in pre- and post-images.
2. One reported landslide was not correctly localised in the LDDB (LS_ID 41). But through some minutes of looking at photographs from a news article and searching in satellite images taken after the event, the exact location of the landslide

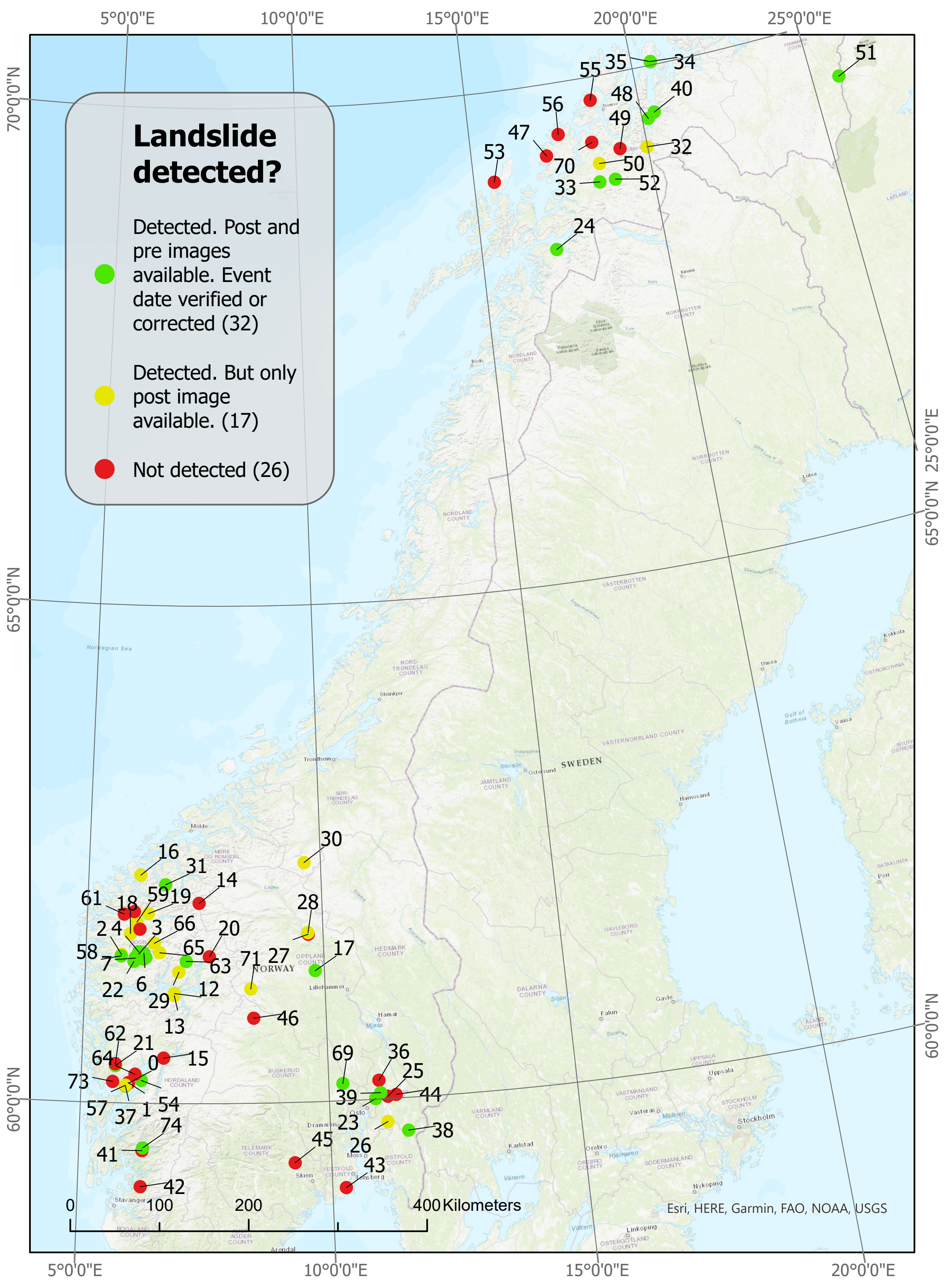

Landslide detected?
● Detected. Post and pre images available. Event date verified or corrected (32)
● Detected. But only post image available. (17)
● Not detected (26)
5°0'0"E
10°0'0"E
15°0'0"E
20°0'0"E
25°0'0"E
70°0'0"N
65°0'0"N
60°0'0"N
NORWAY
SWEDEN
0 100 200 400 Kilometers
Esri, HERE, Garmin, FAO, NOAA, USGS

Figure 4.1: *Map of examined events. Numbers represent the landslide identifier (LS_ID). The green dots represents the location of events that will be used as training samples for Deep Learning. The yellow events are not suited for difference images, i.e. dNDVI is not possible to calculate. Red events were not possible to detect from S2-images. When LS_IDs are mentioned refer to this figure for the geographic location of the corresponding landslide.*

was eventually found. A new point was put in place of the old one and given the ID 74.

3. It can be difficult to pinpoint the event date if the landslide happened outside the summer season. Generally lower green vegetation makes the difference in NDVI much smaller than compared to summer events.

4. Additional landslides were discovered by LS_IDs 74, 48 and 37. These are not the specific landslides that were reported to regObs, but are landslides that happened close by and was detected in the dNVDI image.

5. Many landslides were not possible to detect in the satellite images, despite being reported as larger than 1000 m^2. See Figure 4.2 and Figure 4.3 for an example of such an event (LS_ID 47). The volume of the landslide covering the road was reported to be <1000 m^3.

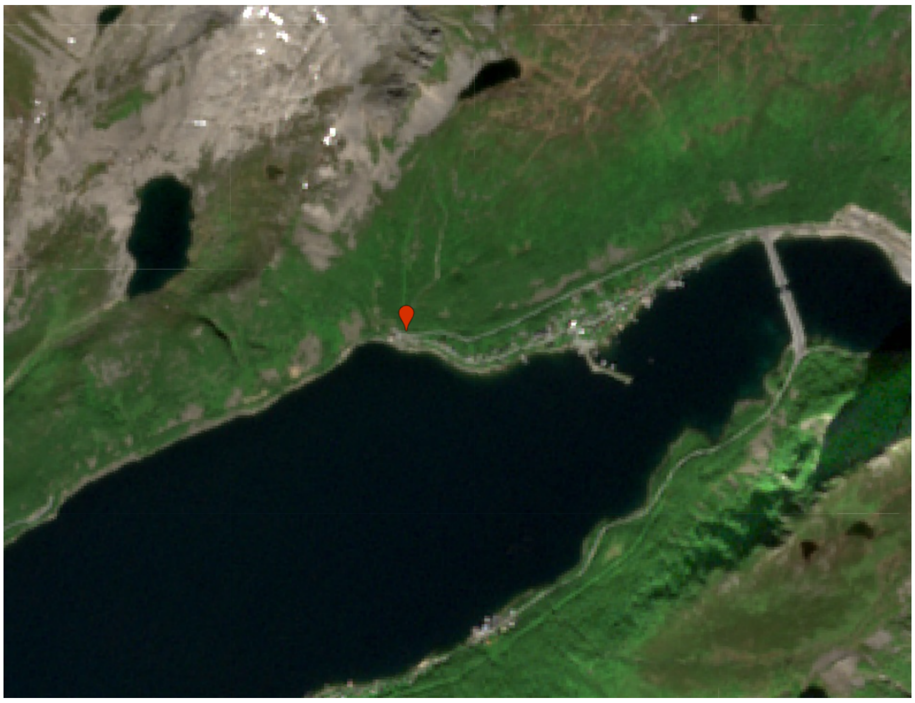


Figure 4.2: *This reported landslide (LS_ID 47), located near the red point, was **not detected** even though it was reported that the volume of the debris covering the road was <1000 m^3.*

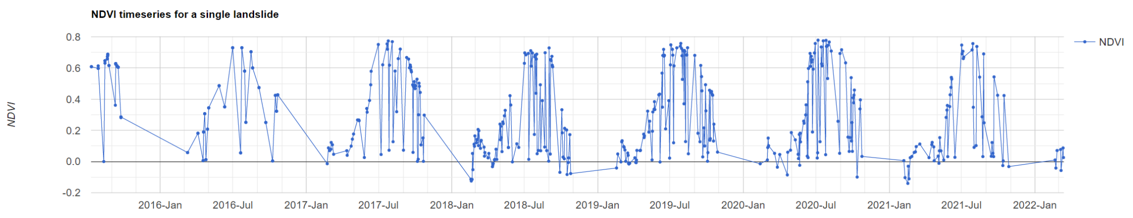


Figure 4.3: *NDVI timeseries for a reported landslide (LS_ID 47) that was not detected in satellite images. NDVI graph show a cyclic behaviour due to different seasons, but no drop in value that would indicate a landslide.*

Figure 4.4 show the manually detected and delineated landslides around LS_ID 54 in

Jondal, Vestland fylke. 23 landslides have been delineated, but there might be more, as they are difficult to differentiate from one another in dNDVI. This event happened during the same rainstorm event as that of Jølster 2019, but 140 km further south. No news report seems to document the event, which is interesting given the number of landslides. No landslide appears to cross any mayor roads, which might explain the absence of media coverage.

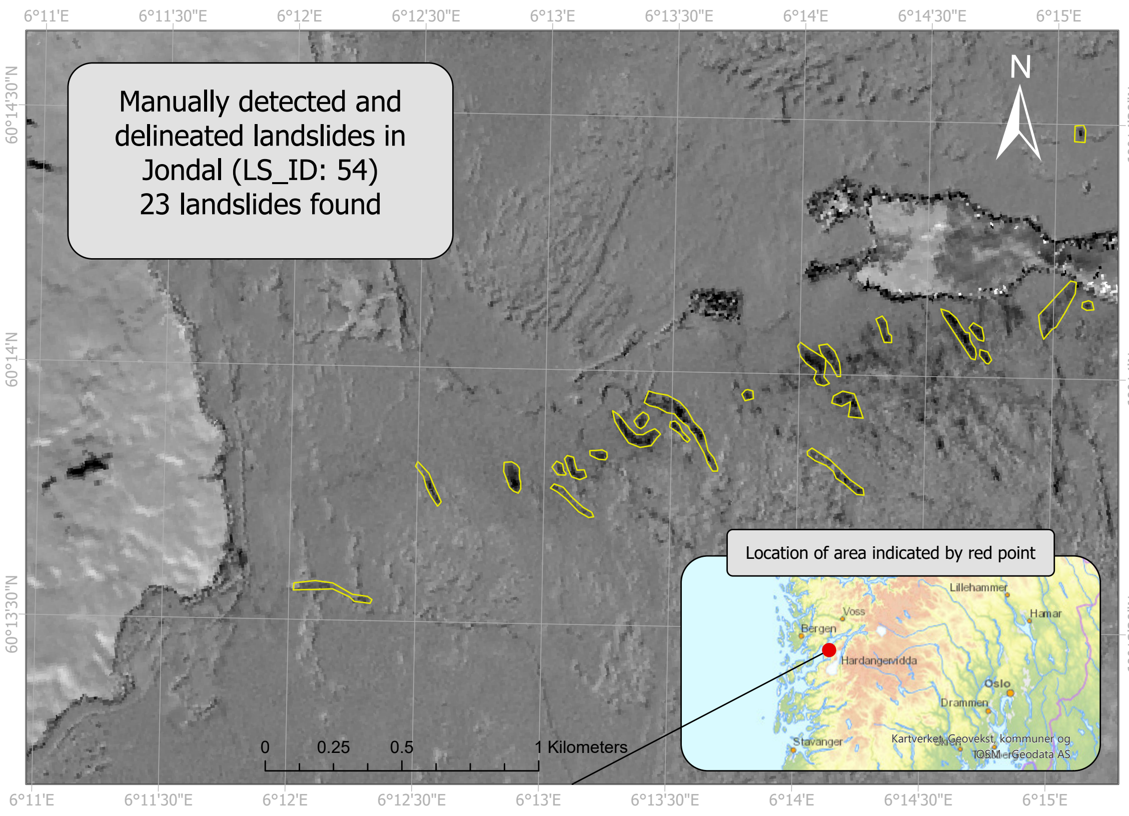


Figure 4.4: *Map of manually detected and delineated landslides around LS_ID 54. Basemap is calculated dNDVI from greenest pixel composite images.*

After all the models had been trained and CMMs was made it was discovered that five of the downloaded post event image tiles were lacking important data in the RGB channels. Either the landslide was obscured by clouds (LS_ID 24,31, and 74) or RGB was lacking completely (LS_ID 33 and 63). Figure 4.5 shows one of the landslides that was obscured by clouds (LS_ID 24). The post-image in the lower right show only clouds and no landslide, while the pre-event image in the lower left is cloud-free. However, the dNDVI image (upper right) was still created and show the landslide scar, even if the post-image was cloudy.

Figure 4.6 shows another example of an RGB-image that has lost some data. The part of the post-image (lower right) containing the landslide is cropped out and only a black nan (not a number) values are present. Like the images with obscured landslides, the dNDVI image (upper right) was still created even if the post-RGB image tile was missing data. It was not clearly understood at the time of writing why some image tiles like these were incomplete.

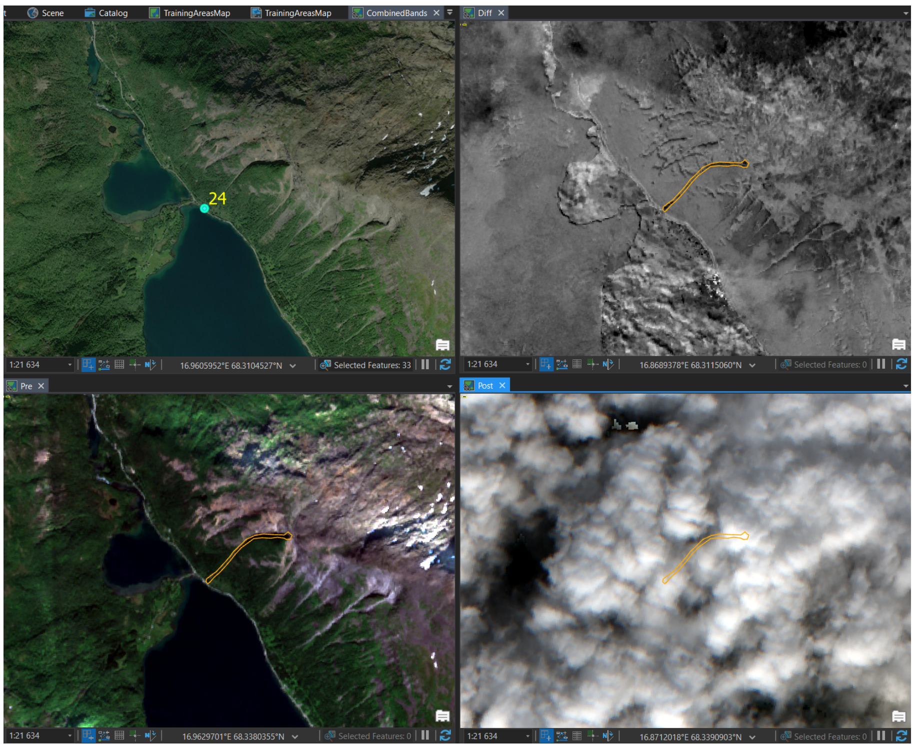


Figure 4.5: *Downloaded image tiles from LS_ID 24.* ***Top left:*** *Aerial image (World Imagery, Esri).* ***Top right:*** *dNDVI.* ***Bottom left:*** *Pre-event RGB image.* ***Bottom right:*** *Post-event RGB image with unsuccessful cloud filtering.*

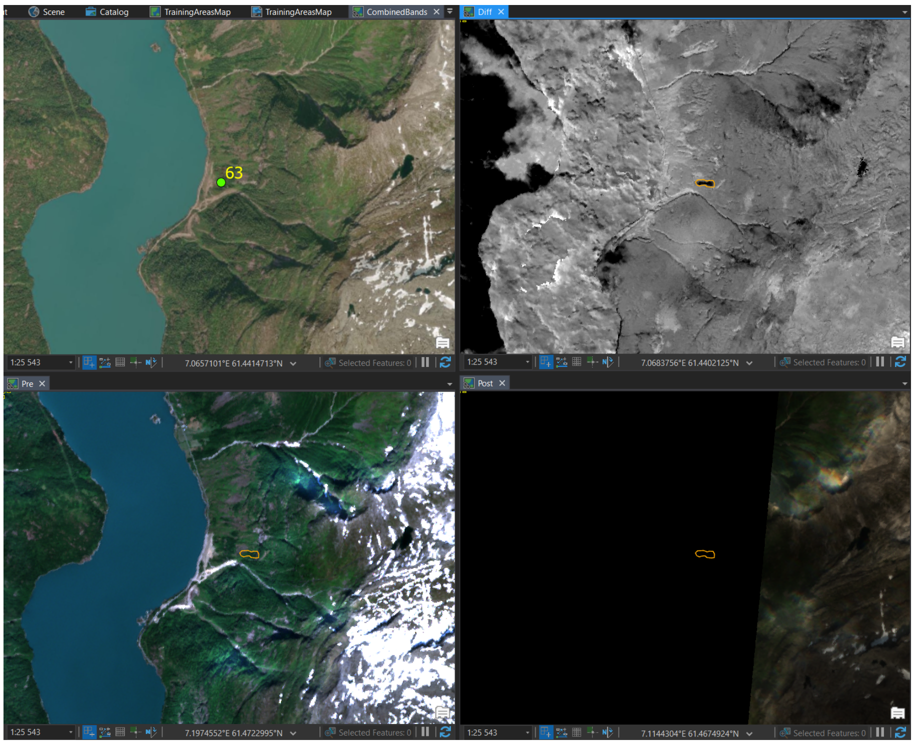


Figure 4.6: *Downloaded image tiles from LS_ID 63.* ***Top left:*** *Aerial image (World Imagery, Esri).* ***Top right:*** *dNDVI.* ***Bottom left:*** *Pre-event RGB image.* ***Bottom right:*** *Post-event RGB image where the part with the landslide is clipped out.*

4.2.2 Geologic distribution of training samples

Figures 4.7—4.10 show the distribution of respectively underlying bedrock, quaternary geology, Corine Land Cover, and landslide type for the landslides in the national data set. The bedrock and quaternary geology are obtained from NGU's N250 geologic map and Quaternary Geology County map (fylkeskart). The land cover is obtained from Corine Land Cover (CLC) 2018 inventory (see section 2.2.6). It is important to note that these are regional maps with limited precision and accuracy. They are not necessarily meant for mapping of relatively small features such as landslides.

In places where the landslide cover multiple types of geologic units or land cover types the landslide will be given multiple classes, so the total number of registered entries will differ from the number of landslides in the national data set. Generally, it can be said that the distribution in geology type is fairly evenly distributed, while the most common land cover is a forest type.

Figure 4.7 shows that slate is the most common underlying bedrock where landslides occur, with 5 events registered, while the less common rocks mylonite, migmatite, limestone, gabbro, and eclogite only have 1 registered event each.

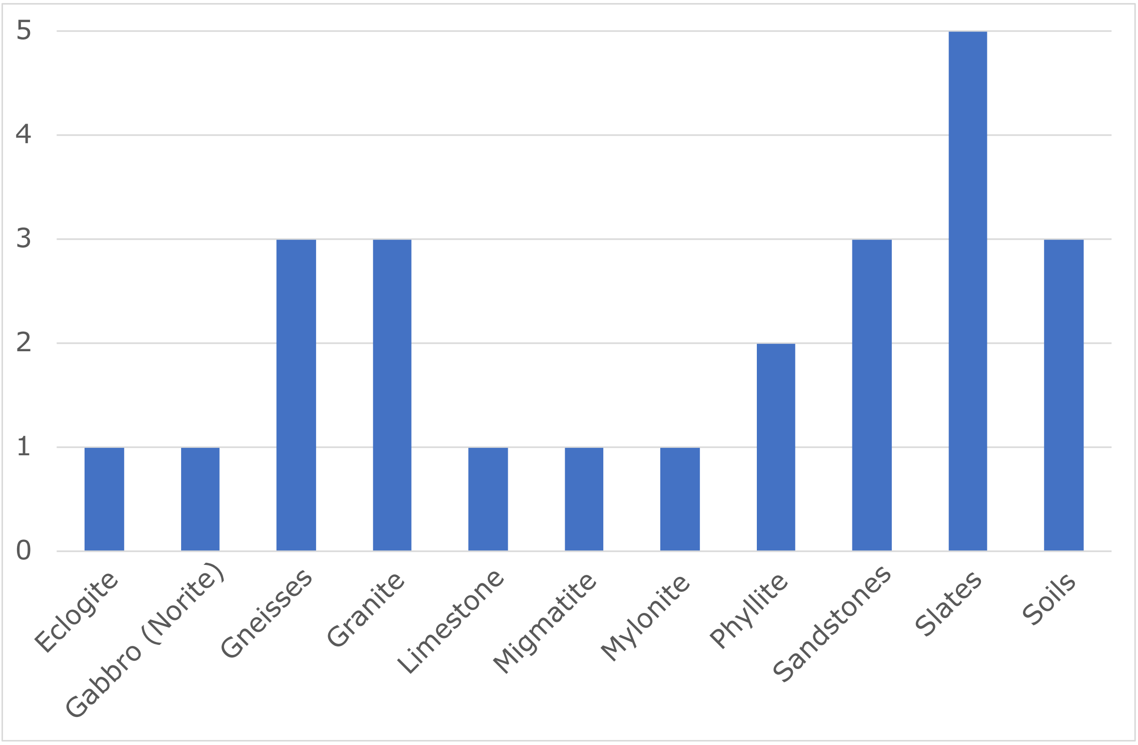


Figure 4.7: *Distribution of underlying bedrock for national landslide polygons.*

Figure 4.8 shows that the most common soil material for landslides to occur in are landslide deposits and till.

Figure 4.9 shows that the landslides events in the national data set are heavily biased towards events occurring in forests. A few events are registered in agricultural areas or low vegetation and sparsely vegetated areas.

Figure 4.10 shows that the most common type of landslide in the national data set is debris flows, which is the type of landslide that has been filtered for (see section 3.1).

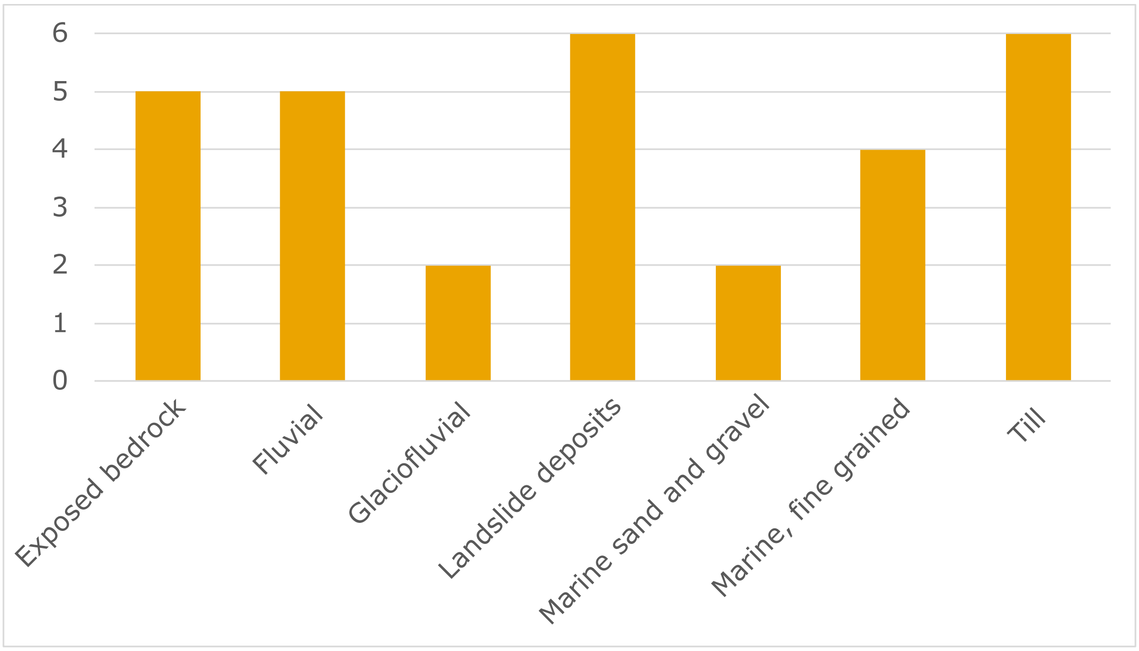


Figure 4.8: *Distribution of quaternary geology for national landslide polygons.*

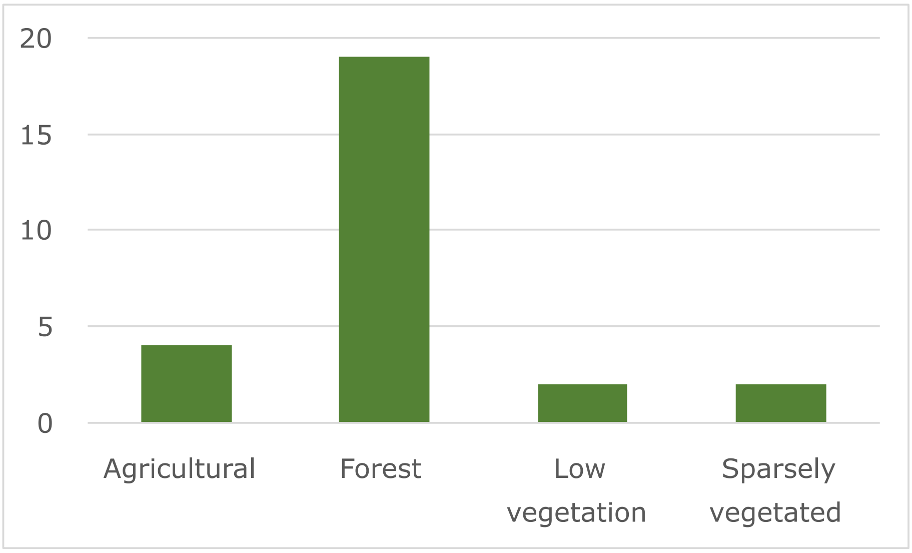


Figure 4.9: *Distribution of Corine Land Cover (CLC) for national landslide polygons.*

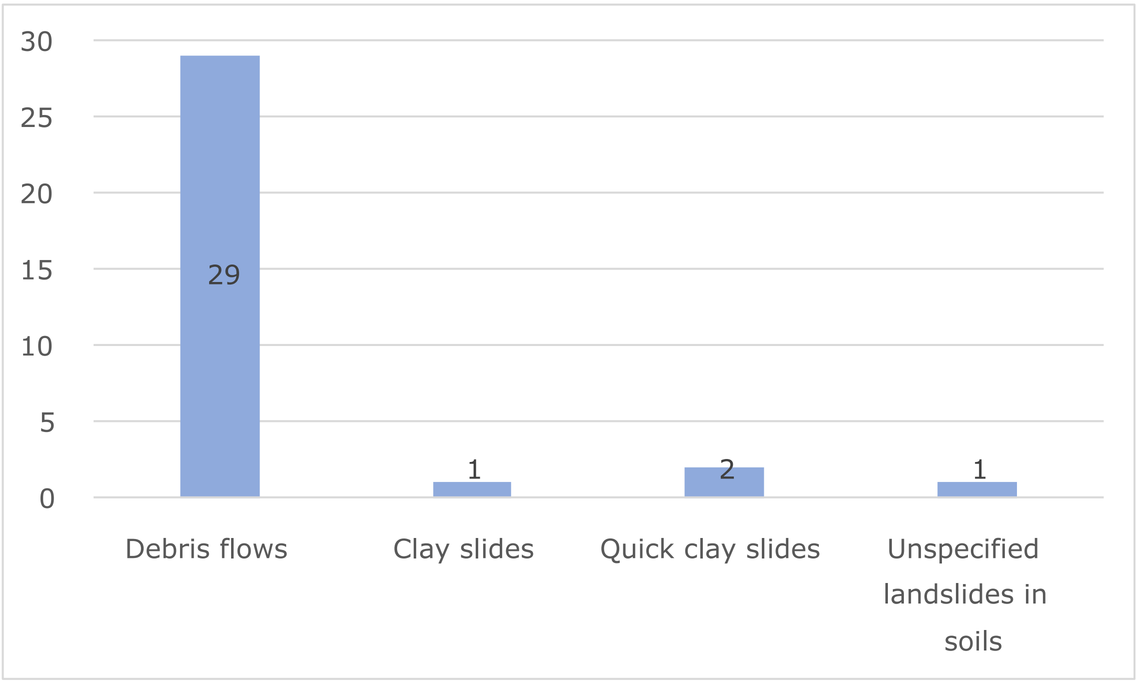


Figure 4.10: *Distribution of landslide types for national landslide polygons.*

4.2.3 DL parameters

All models have been trained on image samples with the parameters shown in Table 4.1:

The export parameters for national training samples are presented in Table 4.1. The parameters are prompted by the "Export Training Data for Deep Learning" geoprocessing tool in ArcGIS Pro.

Table 4.1: *Export parameters for national training samples exported using the Geoprocessing tool "Export Training Data for Deep Learning"*

Export parameters	Value	Unit
Pixel dimension	10	Meters
Chip size	180	Pixels
Stride length	70	Pixels
Rotation angle	30	Degrees
Metadata format	RCNN Masks	-

In Table 4.2 the *hyperparameters* for the model training are shown. *Hyperparameters* are parameters that control the learning process, but are not part of the model itself. They remain the same when training ends (Nyuytiymbiy, 2020). They have been chosen by trial and error to get models that are trained to the point where an extra epoch will not make the model perform better. In the following each parameters will be described:

Table 4.2: *Hyperparameters set for the training of the national models using the Geoprocessing tool "Train Deep Learning Model."*

Hyperparameters	Value
Epochs	75
Batch	8
Backbone model	ResNet-50
Validation	20 %
Freeze backbone model	True

Epochs are the number of iterations that the model goes through a training sample set. The model adjusts its weights (section 2.3) only after each epoch, so for the model to continuously improve its performance it needs several epochs.

Batch represents the number of image samples that the model will be trained on at the same time. This number is dependant on the size of the graphical processing unit (GPU) and the data size of the training samples. Batch number must always be equal or lower than the epoch value.

Validation is the percentage of training samples that will be set aside by the algorithm to use as validation samples when training the model. The model will only "see" these samples after each epoch of training. This allows the algorithm to know how well the model is performing against unseen data, so that it can adjust the weights (section 2.3) to minimise the loss function (improving the model).

The **freeze backbone model** option will, if selected, force the main layers of the backbone model (Transfer learning, see section 2.3.4) to stay the same. This saves computing time, but may make the model less accurate (Sagar, 2019).

4.2.4 Landslide detection

Model Type 1a: Trained on dNDVI images only and is the model with overall best score. It predicts many landslides correctly, and it found almost the exact shape of the big landslide south of Jølstravatnet. However, it still does not predict many of the small clusters of landslides north of Vassenden (middle of the map). It is also excessive in its predictions with many false positives (blue). It received the highest MCC, recall and F1 score of all the models, and the precision is only slightly lower than that of model Type 3.

Model Type 1b: Trained on dNDVI images and a digital elevation model (DEM). This model makes fewer predictions than the Type 1a model, so the number of false positives and true positives are lower, reflected in the lower performance metric scores. Some of the false positives are even bodies of water.

Model Type 1c: Trained on dNDVI images and slope information. This is the model with overall worst score. It makes only one correct prediction (far left next to a cluster of false positives) and makes generally few false predictions.

Model Type 1d: Trained on dNDVI images, DEM, and slope information. It performs slightly better than model Type 1c, as reflected in slightly higher performance metric scores, but the difference is only 0.01 for the MCC score and 0.02 for the precision.

Recall and F1 values remain the same. It makes correct predictions in two small areas and is nowhere near in detecting a full landslide.

Model Type 2: Trained on dVV, dVH, dNDVI images, RGB-post, NIR-post, and DEM. This model makes better predictions than Type 1b, 1c, and 1d models, but still very few landslides are actually detected and the number of false positives are greater than true positives. Thus, the performance metric scores are fairly low.

Model Type 3: Trained on dNDVI, RGB-post, NIR-post, and slope. The model has very few false negatives compared to the rest of the models, which is also indicated by the highest precision score (0.31) of all models.

Model Type 4: Trained on VV-pre, VH-pre, VV-post, VH-post, and slope. This model performs similar to Type 3 model, but the metric scores are overall lower. It makes few predictions, but since the number of false negatives are fairly low, the precision score (0.26) is higher than the other metrics for this model.

4.3 NATIONAL VS. LOCAL DATA SET

4.3.1 DL parameters

Table 4.3: *Export parameters for local training samples exported using the Geoprocessing tool "Export Training Data for Deep Learning"*

Export parameters	Value	Unit
Pixel dimension	10	Meters
Chip size	180	Pixels
Stride length	70	Pixels
Rotation angle	90	Degrees
Metadata format	RCNN Masks	-

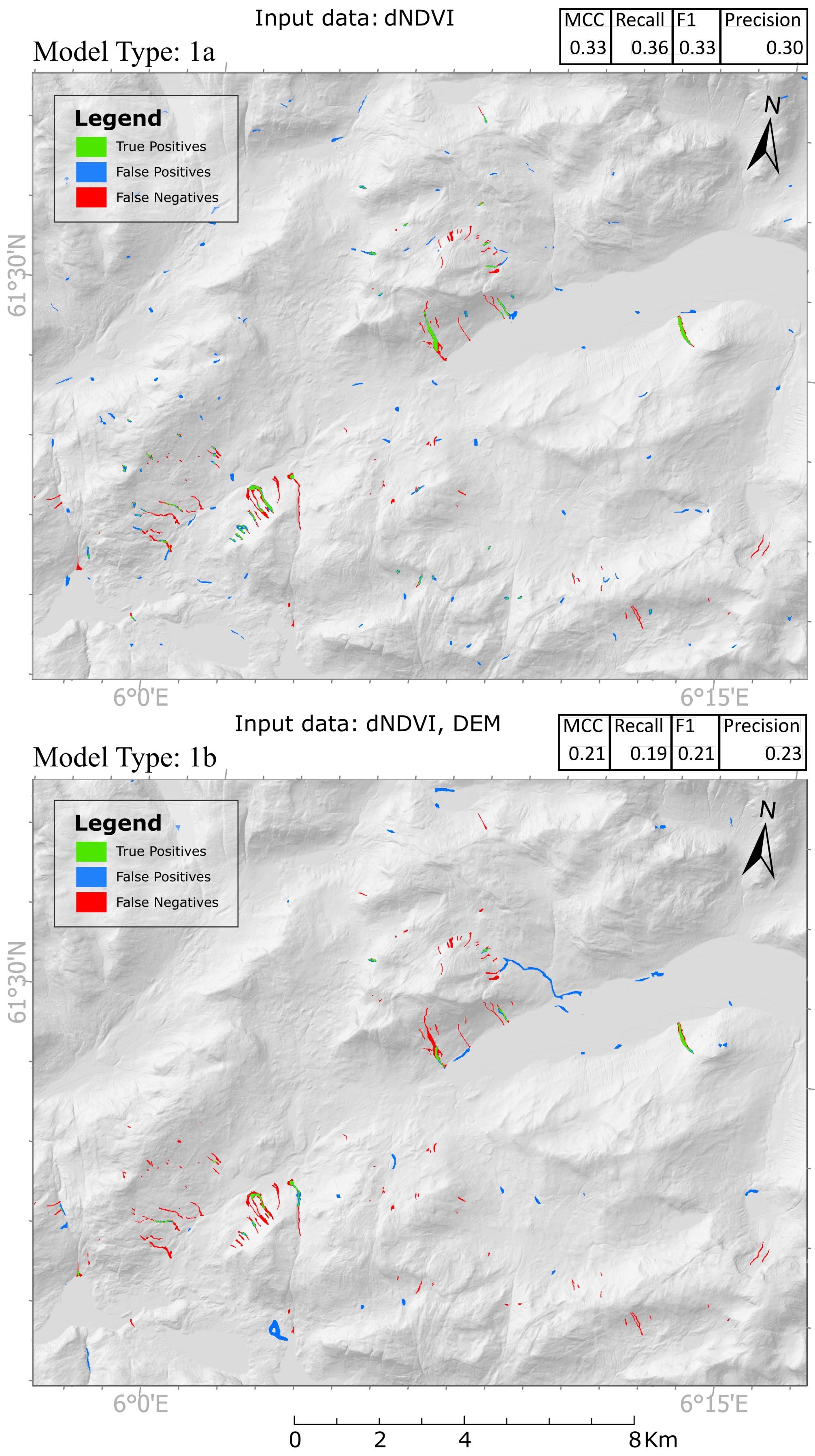
Table 4.4: *Hyperparameters set for the training of the local models using the Geoprocessing tool "Train Deep Learning Model."*

Hyperparameters	Value
Epochs	75
Batch	8
Backbone model	ResNet-50
Validation	20 %
Freeze backbone model	True

4.3.2 Landslide detection

Presented in this section are the resulting Confusion Matrix Maps (CMM) from inferencing of the models trained on the local data set on the Jølster area.

Model Type 1a: Trained on dNDVI images and a digital elevation model (DEM). This model makes the most correct predictions of all the locally trained models. This is reflected in the highest recall value of 0.39. It also outperforms the Type 1a model trained on the national data set. The validation of the model shows that it also makes excessive predictions (many false positives), so the precision score is lower.



MCC	Recall	F1	Precision
0.33	0.36	0.33	0.30

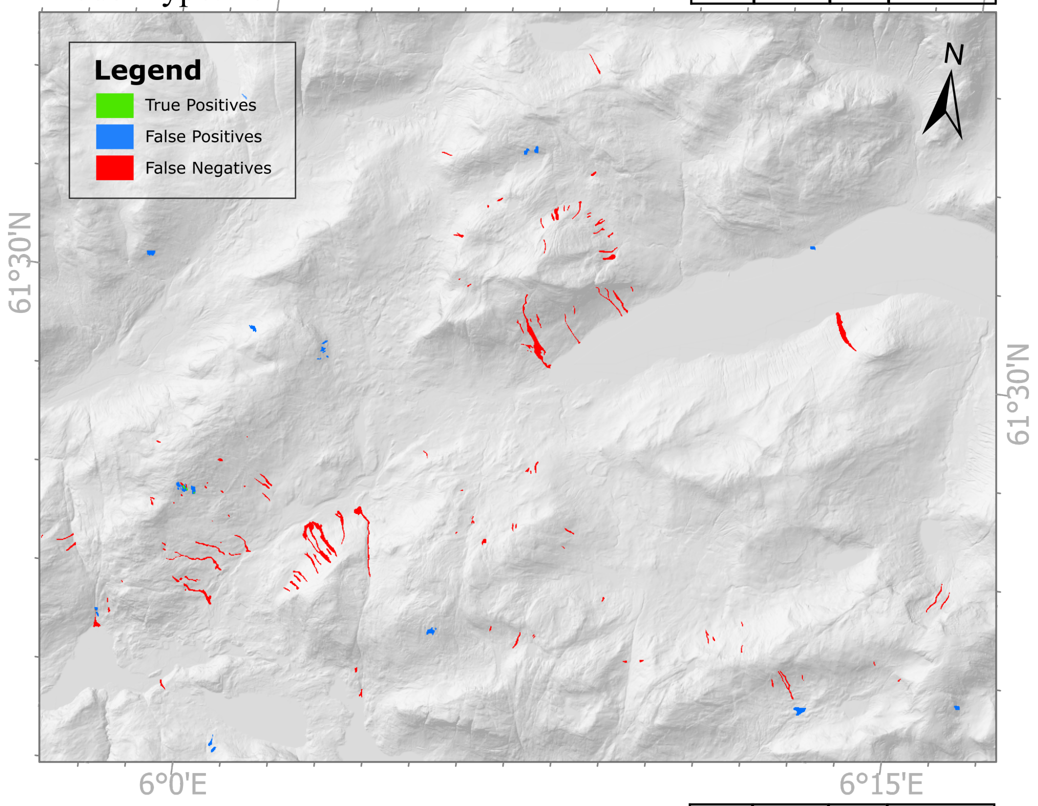
MCC	Recall	F1	Precision
0.21	0.19	0.21	0.23

Figure 4.11: *Confusion Matrix Maps and performance metrics from models 1a and 1b trained on **national** data set.*

Model Type: 1c

Input data: dNDVI, slope

MCC	Recall	F1	Precision
0.01	0.01	0.01	0.03



Model Type: 1d

Input data: dNDVI, DEM, slope

MCC	Recall	F1	Precision
0.02	0.01	0.01	0.05

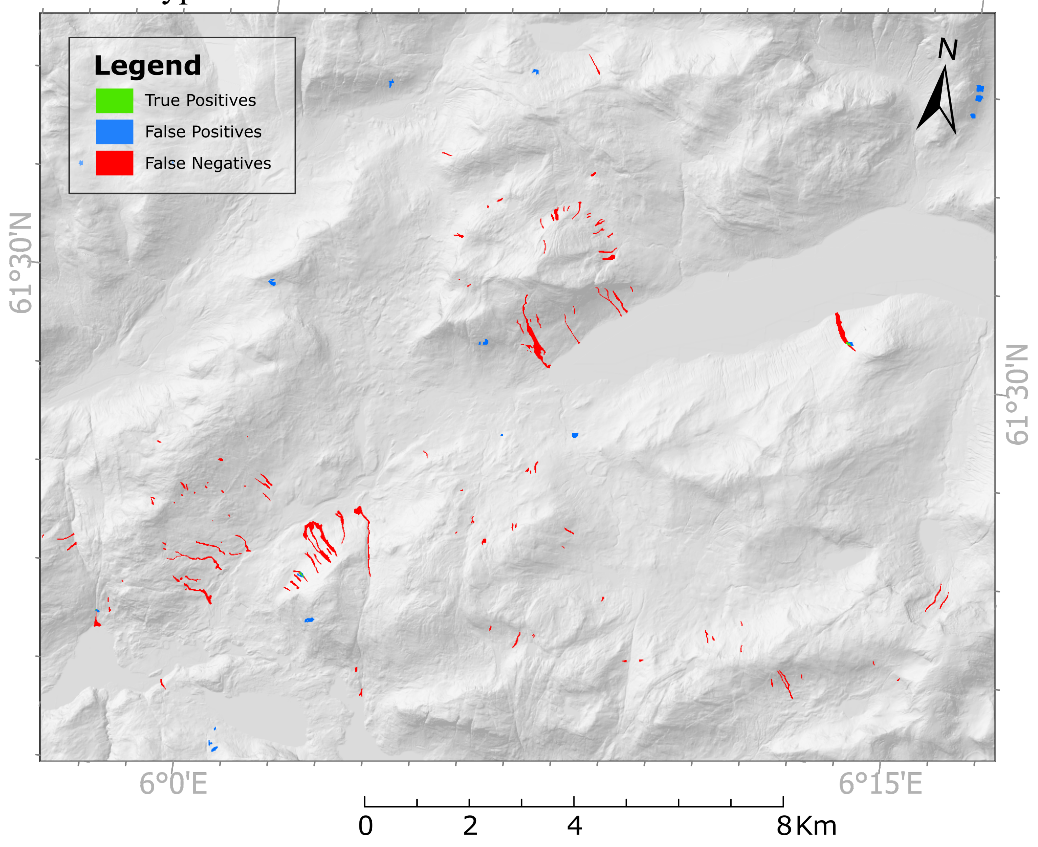
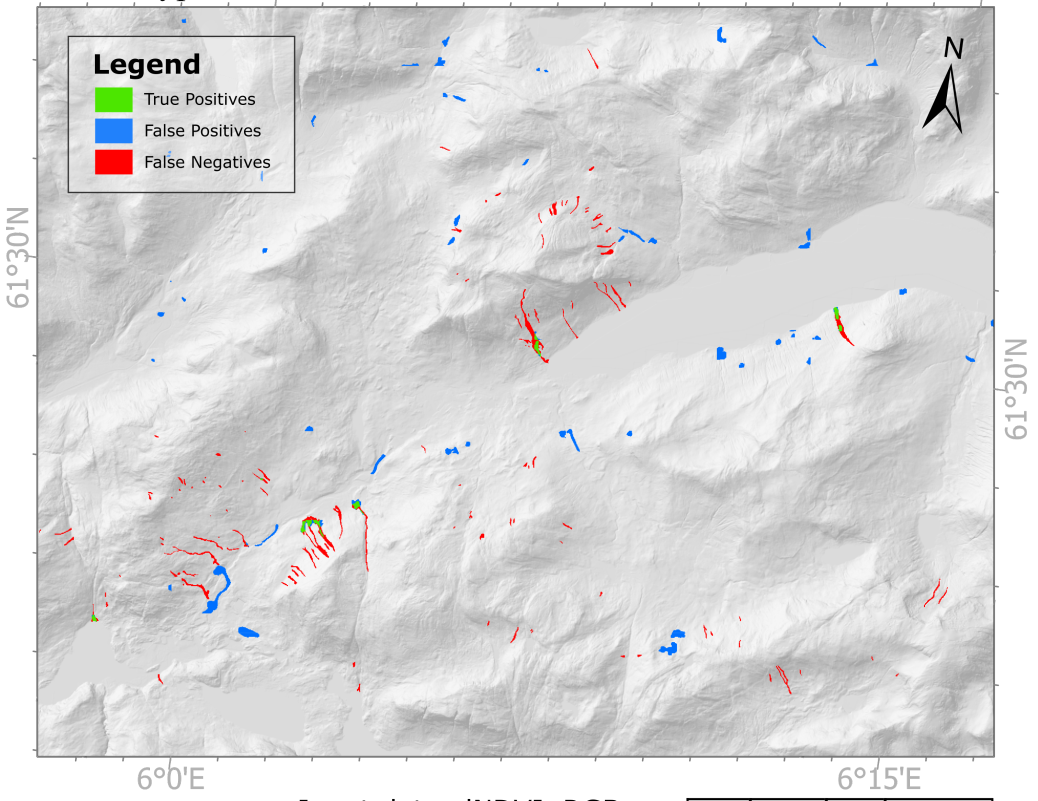


Figure 4.12: *Confusion Matrix Maps and performance metrics from models 1c and 1d trained on **national** data set.*

Model Type: 2

Input data: dVV, dVH,dNDVI, RGB+NIR_post, DEM

MCC	Recall	F1	Precision
0.11	0.12	0.12	0.11



Model Type: 3

Input data: dNDVI, RGB+ NIR_post, slope

MCC	Recall	F1	Precision
0.12	0.05	0.08	0.31

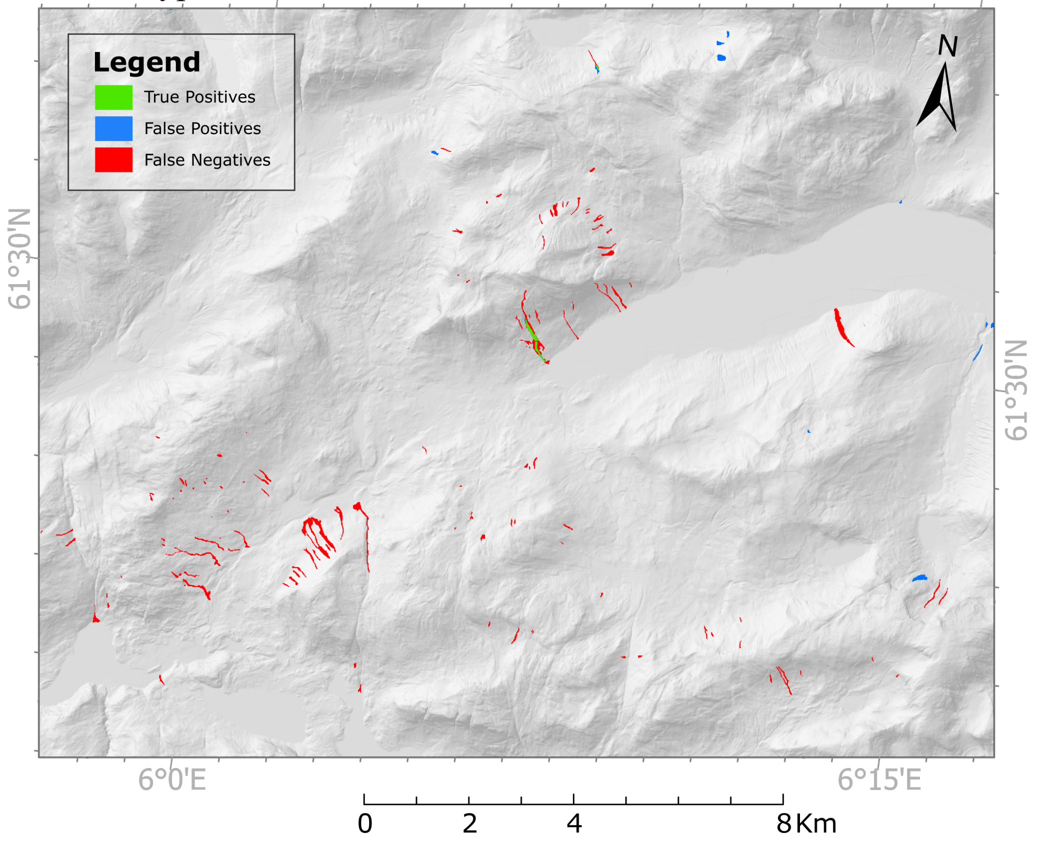
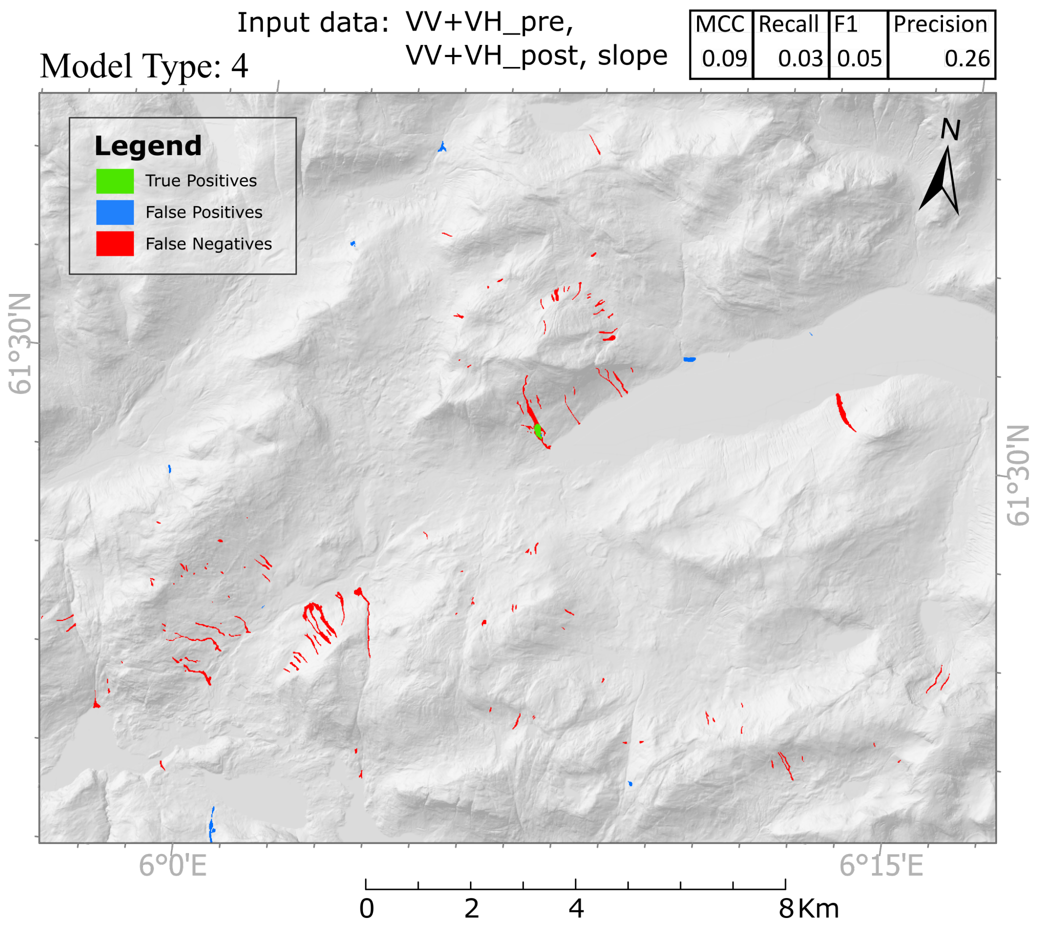


Figure 4.13: *Confusion Matrix Maps and performance metrics from models 2 and 3 trained on **national** data set.*



MCC	Recall	F1	Precision
0.09	0.03	0.05	0.26

Figure 4.14: *Confusion Matrix Maps and performance metrics from model 4 trained on **national** data set.*

Model Type 1b: Trained on dNDVI images and a digital elevation model (DEM). This model makes slightly fewer predictions than Type 1a model, as shown by higher precision, but lower recall.

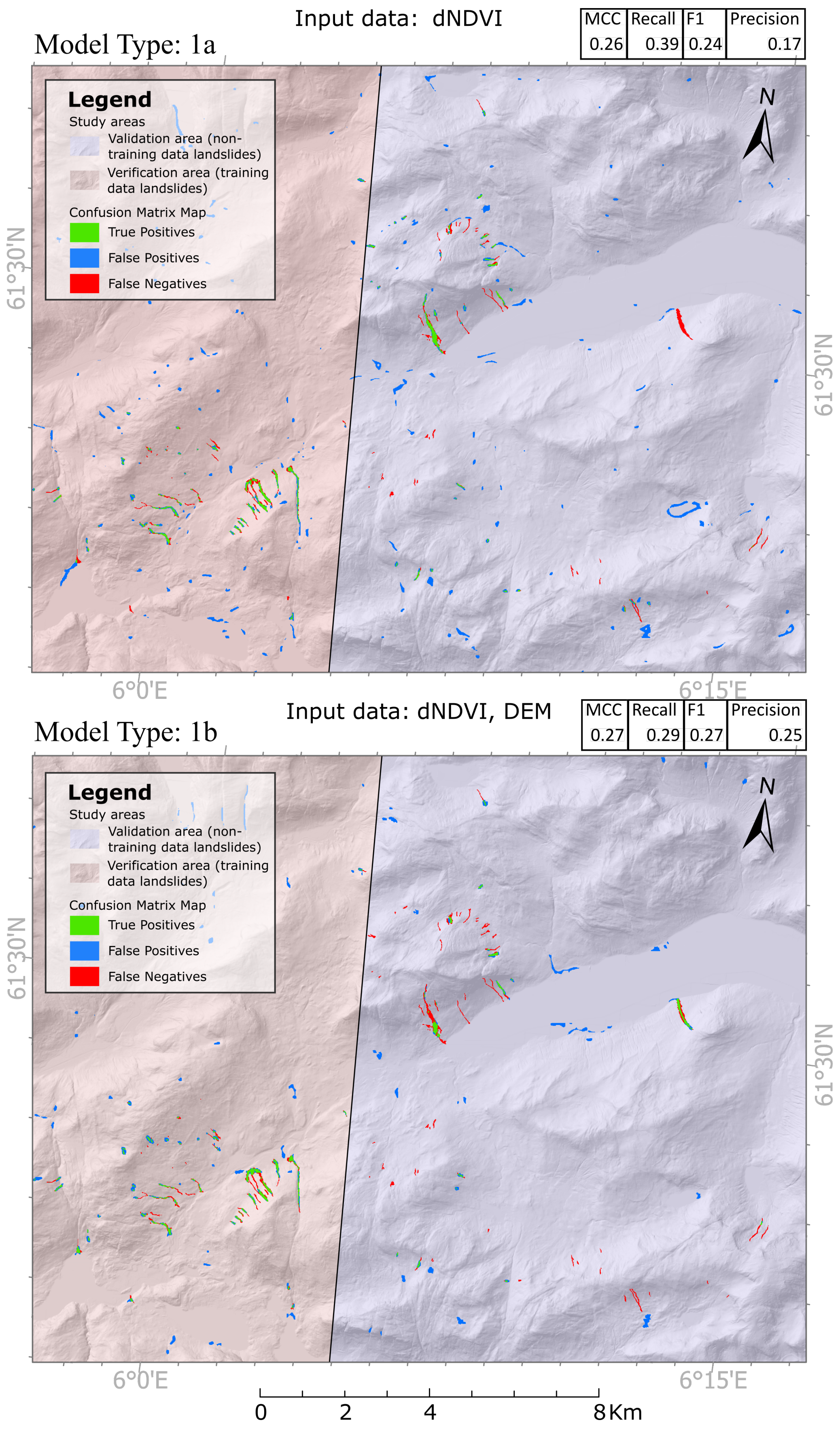
Model Type 1c: Trained on dNDVI images and slope information. This model makes only one correct prediction and many more false predictions, yielding quite low performance scores.

Model Type 1d: Trained on dNDVI images, DEM, and slope information. This model performs slightly better than model Type 1c, with almost the full area of one landslide detected. Still, it makes few correct predictions, so scores are low.

Model Type 2: Trained on dVV, dVH, dNDVI images, RGB-post, NIR-post, and DEM. One landslide is correctly predicted and several false positive are made by this model. Overall very low performance metric scores.

Model Type 3: Trained on dNDVI, RGB-post, NIR-post, and slope. Slightly more correct predictions than Type 2 model, but still they are only of the two largest landslides in the area.

Model Type 4: Trained on VV-pre, VH-pre, VV-post, VH-post, and slope. This is the locally trained model with overall lowest performance metric scores. It makes a few predictions, but almost all are false negatives, so scores are low.



Model Type: 1a — Input data: dNDVI

MCC	Recall	F1	Precision
0.26	0.39	0.24	0.17

Model Type: 1b — Input data: dNDVI, DEM

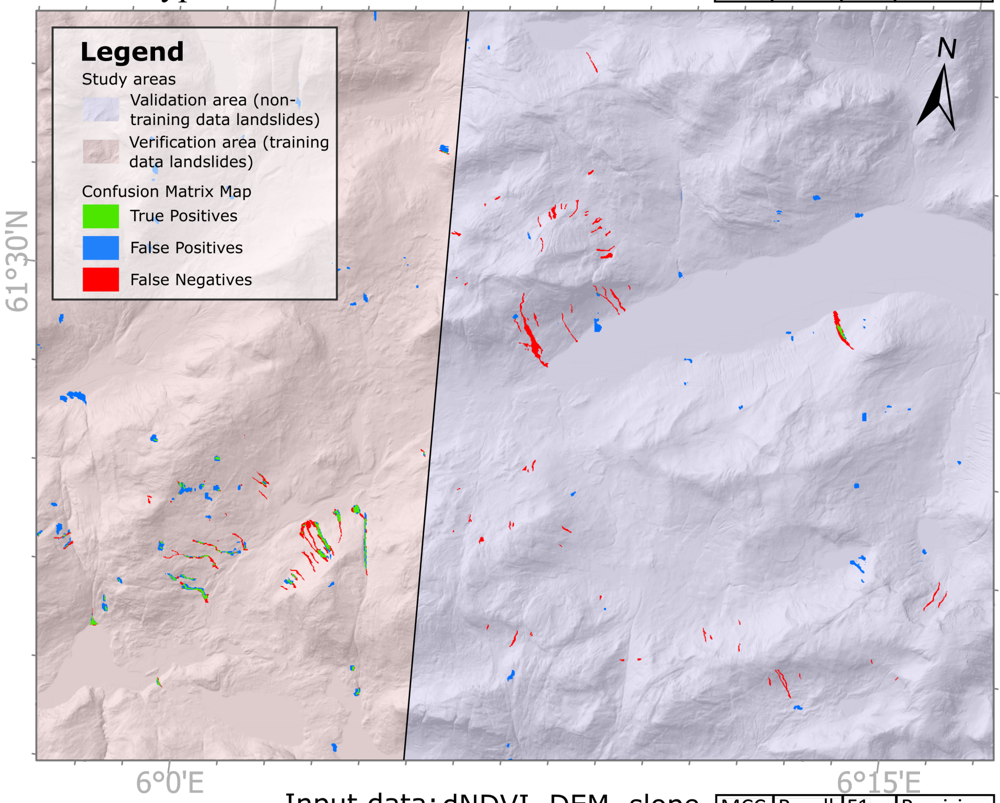
MCC	Recall	F1	Precision
0.27	0.29	0.27	0.25

Figure 4.15: *Confusion Matrix Maps and performance metrics from models 1a and 1b trained on **local** data set.*

Model Type: 1c

Input data: dNDVI, slope

MCC	Recall	F1	Precision
0.04	0.03	0.04	0.07



Model Type: 1d

Input data: dNDVI, DEM, slope

MCC	Recall	F1	Precision
0.17	0.15	0.17	0.21

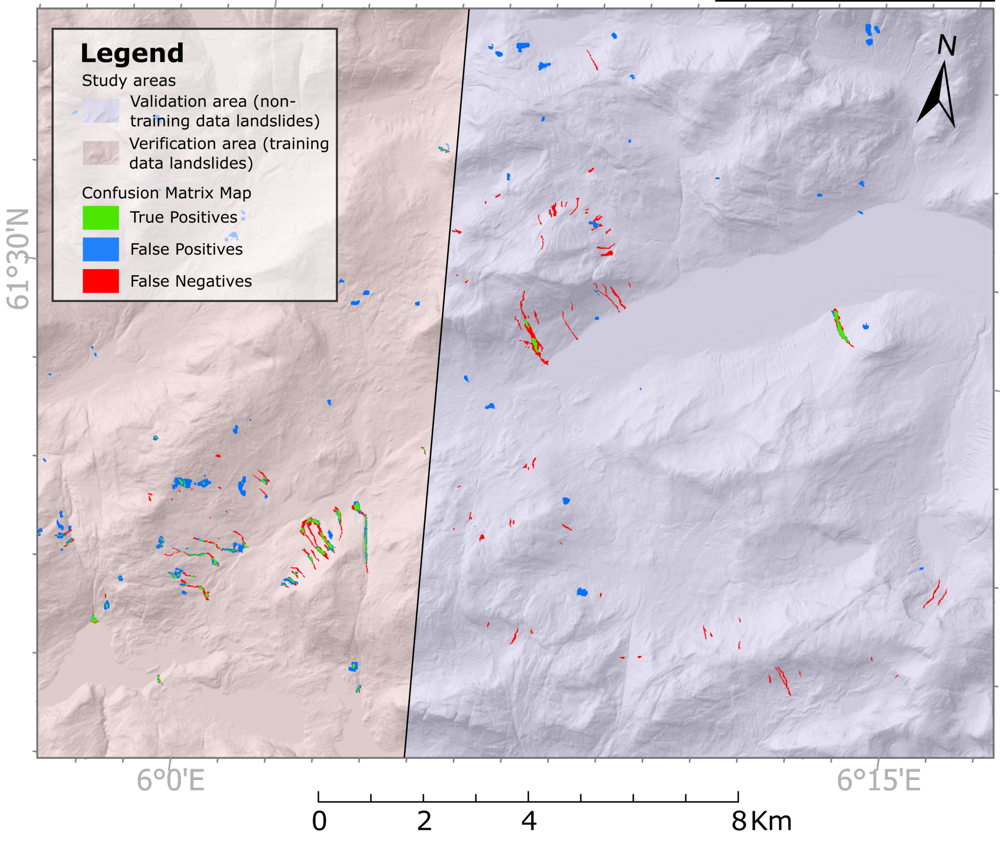
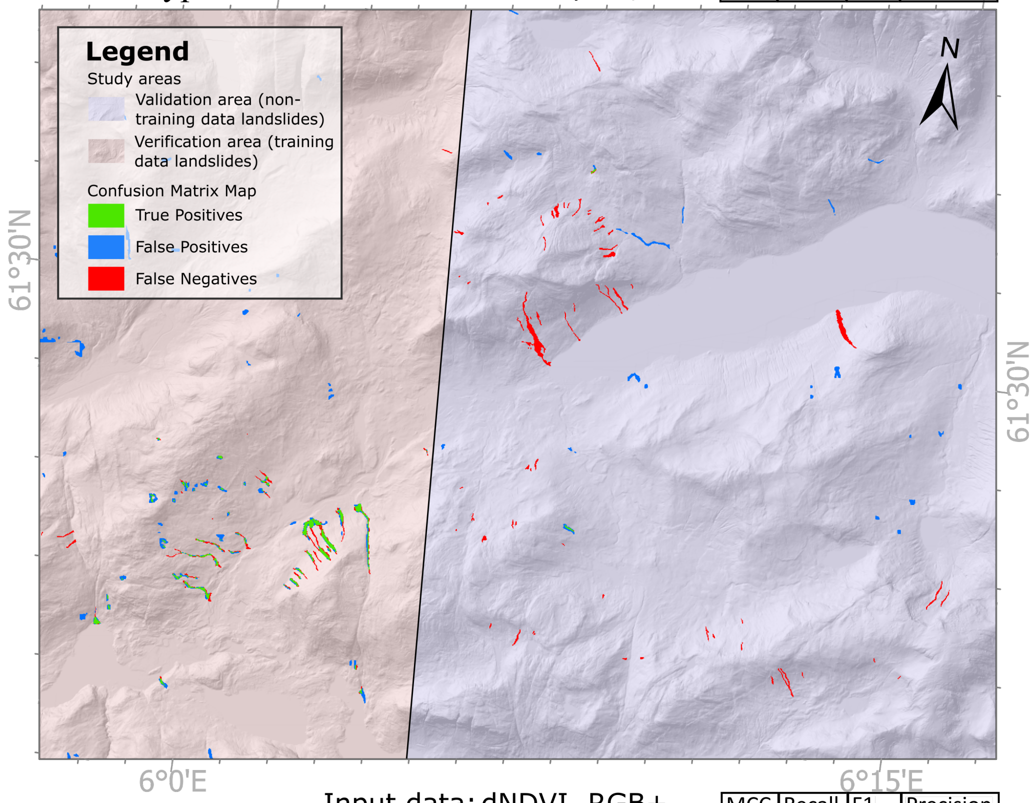


Figure 4.16: *Confusion Matrix Maps and performance metrics from models 1c and 1d trained on* ***local*** *data set.*

Model Type: 2

Input data: dVV, dVH,dNDVI, RGB+NIR_post, DEM

MCC	Recall	F1	Precision
0.03	0.02	0.03	0.04



Model Type: 3

Input data: dNDVI, RGB+ NIR_post, slope

MCC	Recall	F1	Precision
0.13	0.11	0.13	0.15

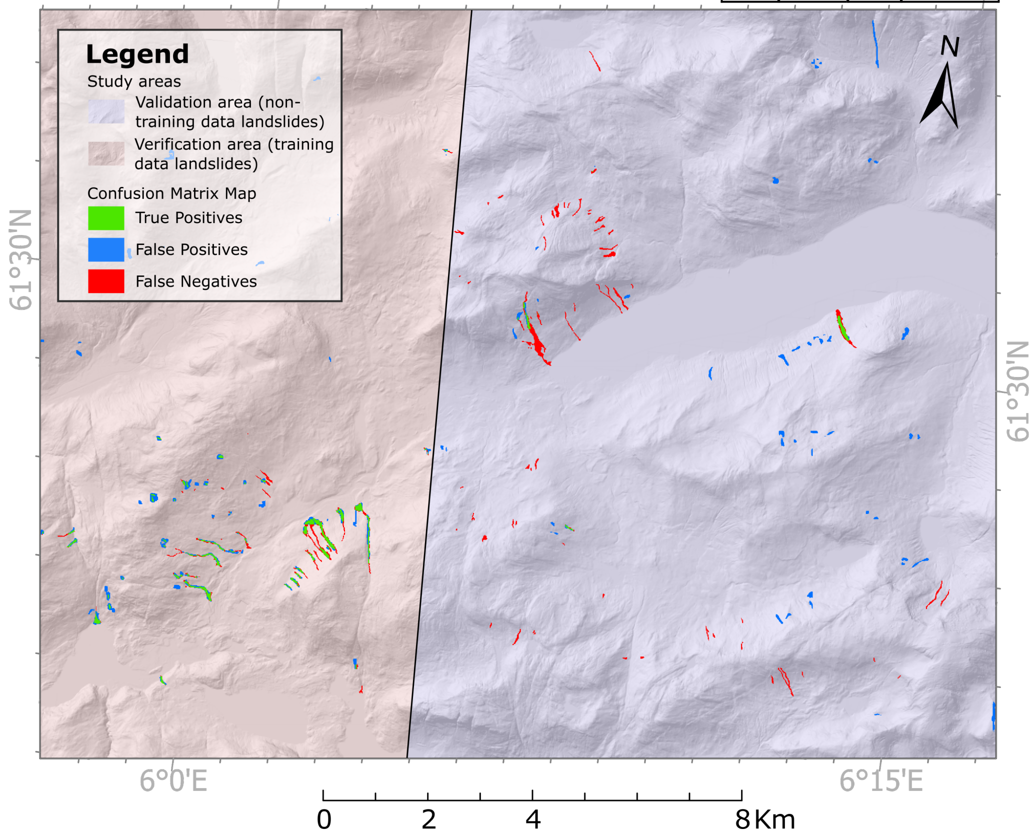
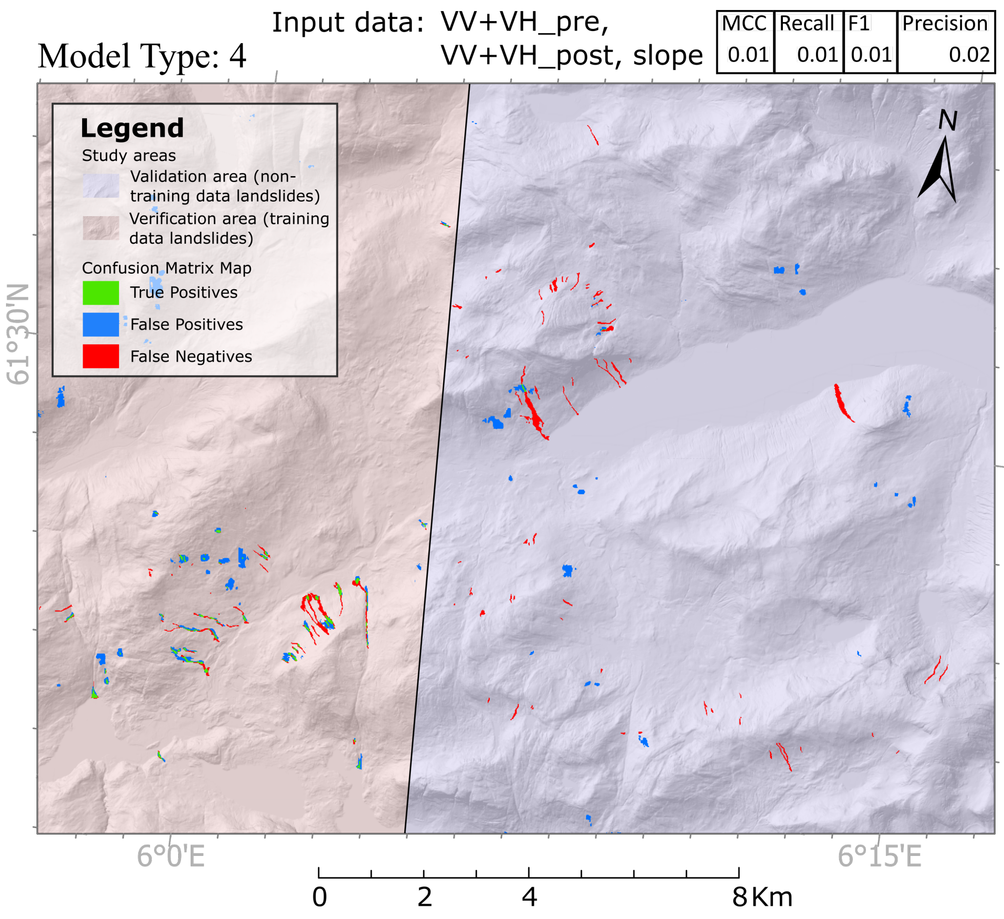


Figure 4.17: *Confusion Matrix Maps and performance metrics from models 2 and 3 trained on **local** data set.*



MCC	Recall	F1	Precision
0.01	0.01	0.01	0.02

Figure 4.18: *Confusion Matrix Maps and performance metrics from model 4 trained on **local** data set.*

4.3.3 Compiled performance metrics

Performance metrics for all models for case A, B, and C are presented in Table 4.5. Also included are the performance metrics for the verification of the locally trained models inferred on the same data set as they have been trained on. This has been done as means of verifying how well the model recognises the landslides it has been trained on. Table is colour graded, with the highest score receiving a green cell, while median values are coloured white, and scores close to zero receive a red cell colour.

Table 4.5 shows that the models inferred on the same landslide samples as they had been trained performed better for all model Types. The maximum recall value was 0.6262 for the Type 3 model.

For illustration of some trends within each model as set of *spider plots* have been created (Figure 4.19). A *spider plot*, also known as radar diagram, presents data from several parameters in a circular chart, with the higher values away from the centre of the chart. It can give a good overview of the predictive performance of a model at just a quick "glance"

Generally, this shows that models trained on a national data set show higher precision and lower recall values, while models trained on local data set show lower precision and higher recall values.

Table 4.5: *Statistics from predicted landslides using all model types for different training data sets. Matthews Corrolation Coefficient (MCC), recall value, F1 score and precision (See section 2.3.7) have been calculated for the three cases (A), (B) and (C).* ***(A)*** *Validation from models trained on the national data set. The models have been applied to landslides it has not been trained on.* ***(B)*** *Validation from models trained on the local data set, where 56 out of 120 mapped landslides have been used as training data. The models are applied to the other half containing the 64 "unseen" landslides.* ***(C)*** *Verification of models trained on the local data set. This is the result from applying the models from (B) to the area containing the 56 training samples. This explains the relatively higher scores.*

(A) Validation national data set

Model Type	MCC	Recall	F1	Precision
1a	0.3286	0.3640	0.3297	0.3013
1b	0.2075	0.1914	0.2091	0.2304
1c	0.0122	0.0056	0.0095	0.0329
1d	0.0184	0.0076	0.0132	0.0513
2	0.1128	0.1174	0.1158	0.1144
3	0.1187	0.0471	0.0816	0.3052
4	0.0875	0.0305	0.0546	0.2564

(B) Validation local data set

Model Type	MCC	Recall	F1	Precision
1a	0.2567	0.3892	0.2402	0.1736
1b	0.2679	0.2926	0.2694	0.2496
1c	0.0440	0.0317	0.0430	0.0665
1d	0.1742	0.1490	0.1738	0.2086
2	0.0252	0.0180	0.0250	0.0408
3	0.1252	0.1116	0.1264	0.1456
4	0.0127	0.0126	0.0149	0.0183

(C) Verification local data set

Model Type	MCC	Recall	F1	Precision
1a	0.4040	0.5988	0.3788	0.2771
1b	0.4552	0.6055	0.4407	0.3464
1c	0.3613	0.4525	0.3561	0.2936
1d	0.3794	0.4838	0.3721	0.3024
2	0.4665	0.6004	0.4552	0.3666
3	0.5222	0.6262	0.5163	0.4392
4	0.2942	0.3872	0.2878	0.2290

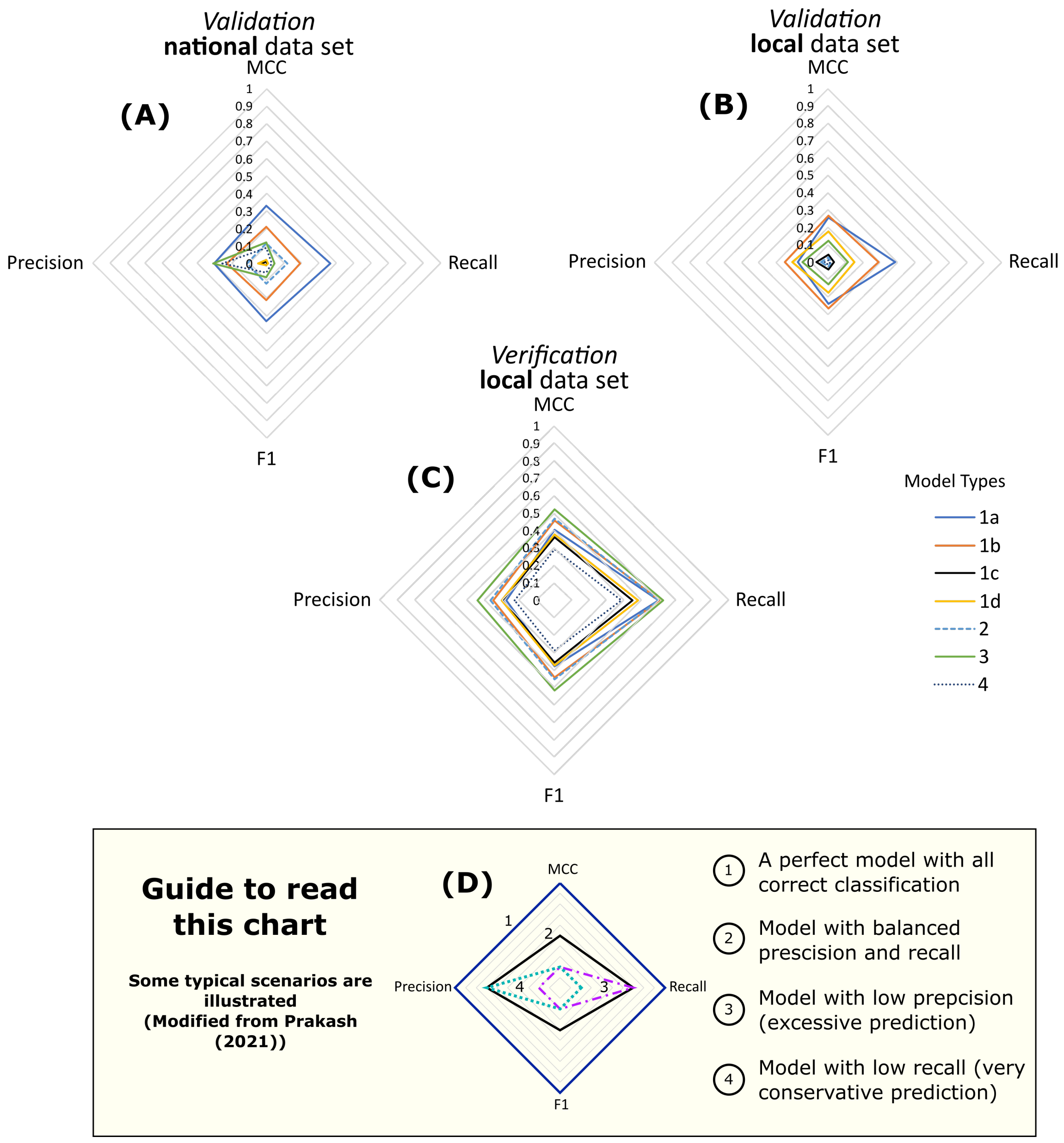


Figure 4.19: *Spider plots showing the performance of the different Deep Learning models. (**A**): Models trained on national data set, inferred on unseen landslides. (**B**): Models trained on local data set, inferred on unseen landslides. (**C**): Models trained on local data set, inferred on same data set as they have been trained on. (**D**): Illustration of some shapes that display models with distinct characteristic.*

Model Type 2 has only been trained on 25 epoch as opposed to 75 for the rest of the models. This was done as it was found that that the model trained on fewer epochs performed better when applied to the Jølster area.

CHAPTER 5

DISCUSSION

In this chapter, the results presented in chapter 4 will be discussed with the research questions from section 1.2 in mind:

How can Deep Learning be applied to detect landslides from satellite images?

where the the sub-research questions will be directly addressed in their separate sections:

1. How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro? (Discussed in section 5.1)
2. What input data will be best suited for detecting landslides? (Discussed in section 5.2)
3. Will training on a national set of landslides compare differently from training only on local training samples? (Discussed in section 5.3)

An important aspect of deep learning is that the data is relevant and shows the feature of interest. The work done in this study has in part not succeeded at creating a set of training samples that show all the relevant features. Some landslide samples that the models has been trained on have not been visible to the DL algorithm. This makes comparison with other studies challenging as the sources of error are quite substantial and the trained models performs poorly.

5.1 WORKFLOW

This sections aims at answering the first sub-research question: *"How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?"*

The first thing worth mentioning is that the workflow for generating and processing remote sensing images for deep learning purposes can be long and complex, as illustrated in the number of flowcharts presented in the methodology chapter (chapter 3). The process involves selection of possible appropriate landslides from the Norwegian Landslide Inventory and then verifying that they are in fact detectable in satellite images. Furthermore, cloud-free image tiles of these landslides must be downloaded using GEE and put together in the same raster file in ArcGIS Pro. Once the desired band combinations are made, landslide polygons can be created and used to export training data for the DL model. The exported training data are then used for training the models. Trained models are tested on the 120 mapped landslides in the Jølster area. The resulting detected landslides polygons are used for creating confusion matrix maps (CMM) and performance metrics are calculated for each model.

The work of Herrera Herrera (2019) also display fairly comprehensive processes for the workflow of image-processing and model testing. This included pre-processing in GEE, feature computations at pixel level, and image segmentation, and image classification, to name a few. All for the purpose of detecting landslides from satellite images.

For the purpose of this study, where the aim has been to find the best suited input data, it has been generated and downloaded many unnecessary image bands that were not used as training data for the final model. Bands not used included pre-images of RGB and NIR, and difference images in RGB, NIR, VV and VH. These bands will redundantly take up computer storage space. The process could therefore have been optimised by not including those bands in the GEE workflow. As an extra benefit, the step in the processing of downloaded image tiles (Figure 3.3) where raster bands are extracted and made into new composites would not be needed, as these band combinations could simply be created directly in the GEE scripts.

The integration of the image tiles downloaded from GEE to ArcGIS Pro is made quite easy as all image tiles are georeferenced and will show up in the ArcGIS map with a simple "drag-and-drop"-manoeuvre. There are many steps in this procedure, and it has been essential to have a software that makes organising and processing of satellites images as uncomplicated as possible.

Bumping into a minor problem or setback that will halt the progress is not uncommon. A full reinstallation of the ArcGIS Pro software and Deep Learning libraries has been performed more than once. Hopefully, the workflow presented in the methodology chapter (chapter 3) will aid the next work examining this problem, so that the setbacks are not as frequent.

It has been a great challenge in having a data-set that has such a large spacial distribution. It would have required less work to simply have one area of limited extent, e.g. 20x20 km, but in Norway, places where enough landslide events are available, are rare. Mapping from low resolution satellite images is very challenging. dNDVI in combination with post RGB is probably the best tool for mapping landslides.

On a final remark, it should be noted the importance of keeping a well organised folder hierarchy. This is of paramount importance for having an efficient workflow in any type of computer work. It has proven valuable for this study.

5.2 INPUT DATA

This section will answer the second sub-research question: *"What input data will be best suited for detecting landslides?"*

The results presented in section 4.2 shows that a single band of dNDVI is the best input data for detecting landslides (Model Type 1a). The maximum MCC score for this model type was 0.33 and maximum recall value was 0.36. For comparison, Prakash et al. (2020) got an MCC score of 0.495 and a recall value of 0.72 as their maximum for their best performing model.

This finding corresponds to that of Fjeld (2018), where it was concluded that "The NDVI is highly applicable for indication of landslide activity". The creation of difference NDVI images makes landslides appear more clearly than in just post-NDVI (Lindsay et al., 2022), and thus, more easily detected by the DL model.

It is still surprising to the author that the performance is reduced when more features are added. It is highly likely that this is due to the problem where some landslide samples were not showing the landslides as stated in section 4.2 and Figure 4.5 and Figure 4.2.

No other research that the author know of has been done on automated landslide detection from dNDVI images, so it is difficult to analyse this in detail using other literature, but it can be said that it performs better than RGB which is often used in other research.

One problem with the dNDVI-approach is that it require more pre-processing effort than acquiring only post-images, which are commonly used (Ghorbanzadeh et al., 2019; Prakash et al., 2020; Prakash et al., 2021; Nava et al., 2021). The extra task of getting pre-event images makes the process particularly more complex, but eventually, when the proper tools get more available this extra "burden" should not be as great.

The results from verification of the reported landslides demonstrates why it is necessary to do the verification in the first place—There might be landslides that have the wrong date reported, wrong location of the point, or it simply might not be visible in the satellite image. If this step is not done, acquiring good quality input data will be difficult.

When assessing the suitability for the different types of input data it was decided that it was best to test on a range of different combinations of image bands to see which combination resulted in the best performing model. The band combinations could have been arranged differently, particularly, the DEM or slope should probably not have been assigned to 6 out of 7 training types. This is reasoned in that the performance metrics drop when DEM and slope is added to the the models with only dNDVI band (type 1 models). The DEM and slope was not removed from the rest of the models as the analysis was not done before all models were trained, and due to time constraints new models have not been trained since discovering that DEM and slope have negative impact on the performance of the models.

It was believed that the topographic information would help in limiting the FP predictions in flatter regions as proposed by Prakash et al. (2021). However, this has not been the case in this study. Though surface topography is considered to be one of the main driving forces for landslides (Ghorbanzadeh et al., 2019), the model does not seem to make that connection. This unexpected outcome was also encountered by Ghorbanzadeh et al. (2019) and Herrera Herrera (2019). They did not attempt to suggest a reason for this issue. One possible explanation to this misinterpretation by the model could be that the neural network does not correlate the value of the DEM layer to the other layers. As no visual information about a landslide is given by the DEM or slope it might draw the conclusion that "anything that is not flat could be a landslide", and then suppress the information given in the optical bands.

It can be interpreted that the reduced model performance for the models is a result of the model complexity getting too large, and the model becomes overfitted, as described in section 2.3.5. It might be that the added layers adds more noise than structured information on the landslides, making the models perform poorly. This is, however, contrary to the results from Herrera Herrera (2019), where the performance metric scores *increased* when more features were added. It is strange, then, that a so abrupt reduction in performance is observed with our models.

Looking at the spider plots in Figure 4.19 it is very clear that even the best performing models still has great potential for improvement. Prakash et al. (2021) present their performance metrics in a similar way, and generally, most of their DL models show better performance.

The results from Prakash et al. (2021) show a small bias towards higher recall scores, which is also the case for the Type 1a model in this study, with only dNDVI. But, the Type 3 and 4 models show very low recall values, but better precision scores. Possibly, the performance scores in Prakash et al. (2021) are higher due to a better quality control of the input data.

Another explanation for the relative low performance is to do with the quality of the data itself. As the results show, not all image tiles contained the relevant data. Five image tiles did in fact not show any landslide at all. Two image tiles lacked RGB and NIR data completely (LS_ID 63 and 33), while three were entirely obscured by clouds (LS_ID 24, 31, and 74). These images should have been excluded from the data set before training commenced.

A perhaps more successful way of creating cloud free images would have been to create a multi-temporal post-event image stack and select the pixels with the maximum NDVI value to produce an aggregate image, a method developed by Lindsay et al. (2022). This would have had the effect of removing cloudy pixels and evening out reduced NDVI signals from agricultural activity (Lindsay et al., 2022). However, the generation of such images require considerable calculation time, as they utilises per-pixel calculations. An effect is also that landslides will appear discoloured (Erin Lindsay 2022, personal communication, 21 June).

As for the Sentinel-1 SAR data (VV and VH bands), the reason for selecting those as training data was that there has been an increasing interest in using SAR data for landslide detection (Nava et al., 2021), due to the radar's "cloud penetrating" abilities (Lindsay et al., 2022). The Type 2 and 4 models that use VV and VH bands both got low performance scores. Type 4 model did receive a precision score of 0.26, which is among the highest precision score for all model types. But, the model is very conservative in its predictions, so it only found 1 of 120 landslides. The analysis methods and classification algorithms needed for landslide detection with SAR data is not yet fully developed, but the research is growing (Nava et al., 2021; Lindsay et al., 2022; Rouault, 2020).

The DL process is still perceived as a "black box", where it is not known how each layer in the image is interpreted and how the model weights the relation between e.g. slope and dNDVI. To the human mind it is clear that if you have a spatial feature with lower dNDVI values and high slope values the feature in question is somewhere steep where there has been a recent loss in vegetation — most likely a landslide. However, the machine does not in this case see that. A better understanding of DL is needed to process this further.

5.3 NATIONAL DATA SET VS. LOCAL DATA SET

In this section the third sub-research question will be addressed: *"Will training on a national set of landslides compare differently from training only on local training samples?"*

Despite the national data set having five landslides that were not visible in its training data, it still performs similarly to the models trained on the local data set. This implies that the rest of the training samples are of good quality. It is quite certain that all the training samples in the local data set show the relevant landslides in the images. One would therefore expect the local models to yield higher performance metric scores than the national models. They do, however, perform relatively similar. Thus, the effect of the five image tiles not showing the landslide in the national landslide data set might not be too large.

Since no model do a successful job of detecting most of the landslide in the Jølster area it is difficult to draw some conclusion on exactly why one model performs better than the other. Particularly for the Type 1c, 1d, 2, 3, and 4 models the performance metrics are so small (See Table 4.5) that it can definitely be said that they do not work as intended.

The local 1b model performs slightly better than the national 1b model. The recall value of the local model is 0.29 compared to 0.19 for the national model. Still, a relative low score, but a noticeable difference.

The reason for the model's low validation results becomes apparent when looking at the verification results (case C in Table 4.5). Even if the models in this case are inferred on the same data set they have been trained on, precision scores are still below 0.5 as shown in Table 4.5. Most models in Prakash et al. (2021) had precision scores exceeding 0.6, and these are for models inferred on unseen landslides (verification), which should be expected to yield poorer results than inferring on the training data set. The same comparison can be made with the results from Ghorbanzadeh et al. (2019). 13 out of their 20 landslide detection models got precision scores well above 0.5.

Given the low validation results compared to the verification results in literature, it is deduced that our models are not trained optimally. The reasons for this can be complex, relating to both the data itself and to the internal processes in the convolutional neural network (CNN). It might be that the training sample size is too low. Our sample size is 52 individual landslides, while Herrera Herrera (2019) had a sample size of 110 individual landslides and Prakash et al. (2021) had a total sample size of more than 11 000 landslides distributed on seven different study areas all over the world. This is quite possibly the most significant explanation for the difference in model performance. In addition, the parameter values set for our DL algorithm are most likely not the best suited values, but it was not the aim of this study to find the optimal model.

This study show that training DL models on a national set of landslides results in models that do not differ considerably from models trained on a local set of landslides. The recall values were slightly higher with the models trained on the local data set, while the precision scores were lower than that of the model trained on the national data set.

5.4 SUMMARY OF DISCUSSION

Here follows the most important message from this discussion:

This work has succeeded in creating a workflow that enables creation of DL models for automated landslide detection in Norway. It does so by generating appropriate images of landslides in Google Earth Engine (GEE). These images are then processed in ArcGIS pro, where landslides are labelled and then training samples are exported so that a DL algorithm can make a model learn what a landslide look like.

Although the workflow does work, it has not succeeded in creating a well-functioning model that detects most landslides. It was found that dNDVI shows the most promise as input data for landslide detection. However, the performance of the models trained on dNDVI are still not as precise as that of other studies, such as Prakash et al. (2021) or Ghorbanzadeh et al. (2019).

It was not observed a considerable difference in training the models on different data sets. There were some slight variations in model performance between models trained on national and local data set, but the overall performance scores were low in both cases and no distinct improvement could be observed.

5.5 SUGGESTED FURTHER STUDIES

For further studies, it is suggested to look at the following:

- Include more training samples from outside of Norway, particularly the Nordic countries, using the methods presented.
- Explore the optimum image "chipification", i.e. the number of augmented images created per landslide polygon. Does the accuracy of the model improve when more data is generated from the same landslides, or does it only take up unnecessary storage space and make the model become overfitted?
- Train a single model on all training samples — both national data set and local data set.
- Explore the effect of different validation percentages. This is something that has not been discussed in this thesis, since it is related to the DL process. By altering the percentage of training samples that the DL algorithm sees during training,

the model will perform differently[1]. The choice of keeping this parameter to 20 % for the work in this thesis was an “educated guess”, so there is room for improvement.

- Fine-tune the model by unfreezing the backbone model[2].
- Make some corrections to GEE script so that cloudy images are not generated.

[1] https://playground.tensorflow.org/

[2] https://keras.io/guides/transfer_learning/

CHAPTER 6

Conclusions

The main research question of this thesis is *How can Deep Learning be applied to detect landslides from satellite images?* To answer this question three sub-research questions was investigated:

1. *How can a national set of landslides be implemented in the same Deep Learning workflow using Google Earth Engine and ArcGIS Pro?*

   The workflow for generating and processing remote sensing images for deep learning purposes is long and complex. this study found that a national set of landslides can be implemented in the same Deep Learning workflow by first verifying that landslides reported to the Norwegian national landslide inventory are detectable in satellite images using Google Earth Engine (GEE). From this verification Sentinel-1 and Sentinel-2 images can be acquired from GEE and processed in ArcGIS Pro. Landslides must be delineated an labelled for the DL algorithm. Once models are trained they can be tested on an landslide inventory that the model has not been trained on. In this case, the Jølster inventory. Prediction results can then be assessed and analysed using the metrics, recall, precision, F1, and MCC.

   The method could be improved by knowing on beforehand what bands to use as training data, as this method currently acquires an excess of data from GEE that will not be used as training data. It is recommended to do more research involving the DL libraries in ArcGIS Pro, due to the simplicity of managing and creating training samples and training models.

2. *What input data will be best suited for detecting landslides?*

   The best suited input data is a "difference Normalised Difference Vegetation Index" (dNDVI). The best performing DL model achieved a precision of 0.30, recall of 0.36, F1 of 0.33 and a Matthews Correlation Coefficient (MCC) of 0.33. A decent DL model that could predict most landslides was *not* produced. The performance metric scores of the best performing model were considerably lower compared to the same metrics from other studies. The reason for the model's poor performance can have root in a number of causes, but low sample size, unrepresentative training samples, and *overfitting* are the most probable causes.

3. *Will training on a national set of landslides compare differently from training only on local training samples?*

   The results demonstrated that there were no significant improvement when using a local training sample set compared to a national sample set. Recall values were slightly higher with the models trained on local data set, while precision scores were lower than that of the national model. It is difficult to draw some conclusion on exactly why one model performs better than the other, as the performance metrics of most models are so low that it can definitely be said that they do not work as intended.

Still the DL process is perceived as a "black box" in that it is not known how each layer in the image is interpreted and how the model weights the relation between e.g. slope and dNDVI. To the human mind it is clear that if you have a spatial feature with lower dNDVI values and high slope values the feature in question is somewhere steep where there has been a recent loss in vegetation — most likely a landslide. However, the machine does not in this case see that. A better understanding of DL is needed to process this further.

References

Arnx, A. (Jan. 2019). *First neural network for beginners explained (with code)*. URL: https://towardsdatascience.com/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf (visited on June 12, 2022).

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., and Nielsen, H. (May 2000). "Assessing the accuracy of prediction algorithms for classification: an overview". In: *Bioinformatics* 16.5, pp. 412–424. ISSN: 1367-4803. DOI: 10.1093/BIOINFORMATICS/16.5.412. URL: https://academic.oup.com/bioinformatics/article/16/5/412/192336.

Brunetti, M. T., Melillo, M., Peruccacci, S., Ciabatta, L., and Brocca, L. (June 2018). "How far are we from the use of satellite rainfall products in landslide forecasting?" In: *Remote Sensing of Environment* 210, pp. 65–75. ISSN: 00344257. DOI: 10.1016/J.RSE.2018.03.016.

Büttner, G., Kosztra, B., Maucha, G., Pataki, R., Kleeschulte, S., Hazeu, G., Vittek, M., et al. (Apr. 2021). *CLC Product User Manual*. URL: https://land.copernicus.eu/user-corner/technical-library/clc-product-user-manual.

Camps-Valls, G., Tuia, D., Zhu, X. X., and Reichstein, M. (2021). *Deep Learning for the Earth Sciences*. Ed. by Camps-Valls, G., Tuia, D., Zhu, X. X., and Reichstein, M. 1st ed. John Wiley \& Sons, p. 405.

Canada Centre for remote Sensing (2019). *Fundamentals of Remote sensing*. URL: https://www.nrcan.gc.ca/maps-tools-and-publications/satellite-imagery-and-air-photos/tutorial-fundamentals-remote-sensing/9309 (visited on Dec. 7, 2021).

Devoli, G. (2017). *Regional early warning systems for rainfall-and snowmelt-induced landslides. Need for an international forum?* Tech. rep. Oslo. URL: www.nve.no.

ESA (2020). *Sentinel-2*. URL: https://sentinel.esa.int/web/sentinel/missions/sentinel-2 (visited on Dec. 7, 2021).

ESA (2021a). *Radiometric Resolutions - Sentinel-2 MSI - User Guides*. URL: https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-2-msi/resolutions/radiometric (visited on Dec. 7, 2021).

ESA (2021b). *WorldView-3*. URL: https://earth.esa.int/eogateway/missions/worldview-3 (visited on Jan. 13, 2022).

ESA's Sentinel-2 team (2015). "The story of Sentinel-2". In: *European Space Agency | Bulletin* 161.1, pp. 2–9. URL: https://www.esa.int/About_Us/ESA_Publications/ESA_Bulletin_161_1st_quarter_2015.

Esri (2021a). *ArcGIS Pro 2.9.2 [Computer software]*.

Esri (2021b). *Deep learning in ArcGIS Pro—ArcGIS Pro | Documentation*. URL: https://pro.arcgis.com/en/pro-app/latest/help/analysis/deep-learning/deep-learning-in-arcgis-pro.htm (visited on June 14, 2022).

Esri (2021c). *Deep Learning Libraries Installers for ArcGIS*. URL: https://github.com/Esri/deep-learning-frameworks (visited on May 15, 2022).

Esri (2021d). *Introduction to deep learning—ArcGIS Pro | Documentation*. URL: https://pro.arcgis.com/en/pro-app/latest/help/analysis/deep-learning/what-is-deep-learning-.htm (visited on May 16, 2022).

Fjeld, M. B. (2018). *Detection of Landslides [MSc thesis]*. Trondheim.

Fukushima, K. (1980). "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". In: *Biological Cybernetics* 36, p. 202.

Furuseth, J. I. (2022). *Landslide mapping using automated remote sensing techniques*. Trondheim.

Gandhi, U. (2021). *End-to-End Google Earth Engine Course*. URL: https://courses.spatialthoughts.com/end-to-end-gee.html (visited on June 16, 2022).

Geological Survey of Norway (Feb. 2015). *Digital elevation models*. URL: https://www.ngu.no/en/topic/digital-elevation-models (visited on June 16, 2022).

Geological Survey of Norway (2021). "Bedrock map of Norway 1 : 1 350 000". In.

Ghorbanzadeh, O., Blaschke, T., Gholamnia, K., Meena, S. R., Tiede, D., and Aryal, J. (Jan. 2019). "Evaluation of Different Machine Learning Methods and Deep-Learning Convolutional Neural Networks for Landslide Detection". In: *Remote Sensing 2019, Vol. 11, Page 196* 11.2, p. 196. ISSN: 20724292. DOI: 10.3390/RS11020196. URL: https://www.mdpi.com/2072-4292/11/2/196/htm%20https://www.mdpi.com/2072-4292/11/2/196.

GISGeography (2021a). *What is NDVI (Normalized Difference Vegetation Index)?* URL: https://gisgeography.com/ndvi-normalized-difference-vegetation-index/ (visited on Jan. 13, 2022).

GISGeography (Oct. 2021b). *Why the Atmospheric Window Matters in Earth Science - GIS Geography*. URL: https://gisgeography.com/atmospheric-window/ (visited on Dec. 8, 2021).

Google Developers (2022). *Classification: True vs. False and Positive vs. Negative | Machine Learning Crash Course*. URL: https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative (visited on June 25, 2022).

Grov, B. (July 2020). *Det farlege regnet*. URL: https://www.nrk.no/vestland/xl/slik-var-det-ekstreme-sommarregnet-i-jolster-1.15030175 (visited on May 13, 2022).

Guzzetti, F., Mondini, A. C., Cardinali, M., Fiorucci, F., Santangelo, M., and Chang, K.-T. (Apr. 2012). "Landslide inventory maps: New tools for an old problem". In: *Earth-Science Reviews* 112.1-2, pp. 42–66. ISSN: 00128252. DOI: 10.1016/j.earscirev.2012.02.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0012825212000128.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (Mar. 2017). "Mask R-CNN". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.2, pp. 386–397. ISSN: 19393539. DOI: 10.48550/arxiv.1703.06870. arXiv: 1703.06870. URL: https://arxiv.org/abs/1703.06870v3.

He, K., Zhang, X., Ren, S., and Sun, J. (Dec. 2015). "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-Decem, pp. 770–778. ISSN: 10636919. DOI: 10.48550/arxiv.1512.03385. arXiv: 1512.03385. URL: https://arxiv.org/abs/1512.03385v1.

Herrera Herrera, M. (2019). "Landslide Detection using Random Forest Classifier [MSc thesis]". PhD thesis. TU Delft.

Highland, L. M. and Bobrowsky, P. (2008). *The landslide handbook - A guide to understanding landslides*. 1325. Reston, Virginia: US Geological Survey Circular, p. 129. DOI: 10.3133/cir1325.

Ibañes, D. (Aug. 2016). *Artificial Neural Networks – The Rosenblatt Perceptron*. URL: https://www.neuroelectrics.com/blog/2016/08/02/artificial-neural-networks-the-rosenblatt-perceptron/ (visited on June 17, 2022).

IBM Cloud Education (Oct. 2020a). *What are Convolutional Neural Networks?* URL: https://www.ibm.com/cloud/learn/convolutional-neural-networks (visited on June 17, 2022).

IBM Cloud Education (Aug. 2020b). *What are Neural Networks?* URL: https://www.ibm.com/cloud/learn/neural-networks (visited on June 9, 2022).

IBM Cloud Education (Mar. 2021). *What is Overfitting?* URL: https://www.ibm.com/cloud/learn/overfitting (visited on May 15, 2022).

Ingale, V., Singh, R., and Patwal, P. (May 2021). "Image to Image Translation : Generating maps from satellite images". In: DOI: 10.48550/arxiv.2105.09253. arXiv: 2105.09253. URL: https://arxiv.org/abs/2105.09253v1.

Ismail, Z. and Jaafar, J. (2013). "DEM derived from photogrammetric generated DSM using morphological filter". In: *Proceedings - 2013 IEEE 4th Control and System Graduate Research Colloquium, ICSGRC 2013*, pp. 103–106. DOI: 10.1109/ICSGRC.2013.6653284.

Jaedicke, C., Lied, K., and Kronholm, K. (2009). "Integrated database for rapid mass movements in Norway". In: *Natural Hazards and Earth System Science* 9.2, pp. 469–479. ISSN: 16849981. DOI: 10.5194/nhess-9-469-2009.

Ji, S., Yu, D., Shen, C., Li, W., and Xu, Q. (Feb. 2020). "Landslide detection from an open satellite imagery and digital elevation model dataset using attention boosted convolutional neural networks". In: *Landslides* 17.6, pp. 1337–1352. ISSN: 16125118. DOI: 10.1007/s10346-020-01353-2. URL: https://link.springer.com/article/10.1007/s10346-020-01353-2.

Klima 2050 (2020). *Klima 2050 Annual Report 2020 No 24*. Tech. rep. Trondheim. URL: https://www.klima2050.no/klima-2050-annual-report-2019-1.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks*. Ed. by Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. Vol. 25. Curran Associates, Inc. URL: http://code.google.com/p/cuda-convnet/%20https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2323. ISSN: 00189219. DOI: 10.1109/5.726791.

Lei, T., Zhang, Y., Lv, Z., Li, S., Liu, S., and Nandi, A. K. (June 2019). “Landslide Inventory Mapping From Bitemporal Images Using Deep Convolutional Neural Networks”. In: *IEEE Geoscience and Remote Sensing Letters* 16.6, pp. 982–986. DOI: 10.1109/LGRS.2018.2889307.

Lillesand, T. M. and Kiefer, R. W. (1979). *Remote Sensing and Image Interpretation*. New York: John Wiley & Sons, Ltd, p. 612. ISBN: 0-471-02609-3.

Lindsay, E., Frauenfelder, R., Ruther, D., Rubensdotter, L., and Nordal, S. (2021). “Landslide detection using Sentinel-2 – Potential and limitations”. In: *In press*, pp. 1–24.

Lindsay, E., Frauenfelder, R., Nava, L., Rüther, D., Rubensdotter, L., Strout, J., and Nordal, S. (2022). “Multi-temporal satellite image composites in Google Earth Engine for improved landslide visibility: Jølster case study”.

Mæhlum, L. (2021). *UTM*. URL: https://snl.no/UTM (visited on June 15, 2022).

Matthews, B. W. (Oct. 1975). “Comparison of the predicted and observed secondary structure of T4 phage lysozyme”. In: *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2, pp. 442–451. ISSN: 0005-2795. DOI: 10.1016/0005-2795(75)90109-9.

Meteorologisk Institutt (Aug. 2019). *Intense byger med store konsekvenser i Sogn og Fjordane 30. juli 2019*. Tech. rep. Bergen. URL: https://www.met.no/nyhetsarkiv/rapport-om-intense-byger-med-store-konsekvenser-i-sogn-og-fjordane-30.juli.

Mohanta, N. (May 2021). *How many satellites are orbiting the Earth in 2021? – Geospatial World*. URL: https://www.geospatialworld.net/blogs/how-many-satellites-are-orbiting-the-earth-in-2021/ (visited on Dec. 6, 2021).

Moreira, A., Prats-Iraola, P., Younis, M., Krieger, G., Hajnsek, I., and Papathanassiou, K. P. (2013). “A tutorial on synthetic aperture radar”. In: *IEEE Geoscience and Remote Sensing Magazine* 1.1, pp. 6–43. ISSN: 21686831. DOI: 10.1109/MGRS.2013.2248301.

Nava, L., Monserrat, O., and Catani, F. (May 2021). *Improving Landslide Detection on SAR Data through Deep Learning*. DOI: 10.1109/lgrs.2021.3127073. arXiv: 2105.00782. URL: https://arxiv.org/abs/2105.00782v1.

Nilelsen, M. A. (2019). *Neural Networks and Deep Learning*. Determination Press. URL: http://neuralnetworksanddeeplearning.com/index.html.

NTB (Aug. 2019). *Skader for minst 50 millioner etter jordrasene i Jølster – Dagsavisen*. URL: https://www.dagsavisen.no/nyheter/innenriks/2019/08/06/skader-for-minst-50-millioner-etter-jordrasene-i-jolster/ (visited on May 15, 2022).

Nyuytiymbiy, K. (Dec. 2020). *Parameters, Hyperparameters, Machine Learning*. URL: https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac (visited on June 21, 2022).

Prakash, N., Manconi, A., and Loew, S. (Jan. 2020). “Mapping Landslides on EO Data: Performance of Deep Learning Models vs. Traditional Machine Learning Models”. In: *Remote Sensing 2020, Vol. 12, Page 346* 12.3, p. 346. ISSN: 20724292. DOI: 10.3390/RS12030346. URL: https://www.mdpi.com/2072-4292/12/3/346/htm%20https://www.mdpi.com/2072-4292/12/3/346.

Prakash, N., Manconi, A., and Loew, S. (May 2021). “A new strategy to map landslides with a generalized convolutional neural network”. In: *Scientific Reports 2021* 11.1, pp. 1–15. ISSN: 2045-2322. DOI: 10.1038/s41598-021-89015-8. URL: https://www.nature.com/articles/s41598-021-89015-8.

Pramoditha, R. (Apr. 2021). *Dimensionality reduction techniques you should know in 2021*. URL: https://towardsdatascience.com/11-dimensionality-reduction-techniques-you-should-know-in-2021-dcb9500d388b (visited on June 14, 2022).

Radio2Space (July 2013). *Components of electromagnetic spectrum - Radio2Space*. URL: https://www.radio2space.com/components-of-electromagnetic-spectrum/.

Rosenblatt, F. (Nov. 1958). "The perceptron: A probabilistic model for information storage and organization in the brain". In: *Psychological Review* 65.6, pp. 386–408. ISSN: 0033295X. DOI: 10.1037/H0042519.

Rouault, C. (2020). "Extreme Multiple Landslide Events in Norway An Investigation of Rainfall and Snowmelt Induced Soil Landslide Detection and Forecasting". In: June.

Sagar, R. (May 2019). *What Does Freezing A Layer Mean And How Does It Help In Fine Tuning Neural Networks*. URL: https://analyticsindiamag.com/what-does-freezing-a-layer-mean-and-how-does-it-help-in-fine-tuning-neural-networks/ (visited on June 21, 2022).

Sameen, M. I. and Pradhan, B. (2019). "Landslide Detection Using Residual Networks and the Fusion of Spectral and Topographic Information". In: *IEEE Access* 7, pp. 114363–114373. ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2935761.

Seldon.io (June 2021). *Transfer Learning for Machine Learning*. URL: https://www.seldon.io/transfer-learning (visited on June 10, 2022).

Shakhadri, S. A. G. (June 2021). *Build ResNet from Scratch With Python*. URL: https://www.analyticsvidhya.com/blog/2021/06/build-resnet-from-scratch-with-python/ (visited on June 25, 2022).

Smilkov, D. and Carter, S. (2022). *A Neural Network Playground*. URL: https://playground.tensorflow.org (visited on June 9, 2022).

Tandberg, E. (2013). *Copernicus – romprogram – Store norske leksikon*. URL: https://snl.no/Copernicus_-_romprogram (visited on Jan. 13, 2022).

United Nations Office for Outer Space Affairs (2021). *Outer Space Objects Index*. URL: https://www.unoosa.org/oosa/osoindex/index.jspx?lf_id= (visited on Dec. 6, 2021).

US Geological Survey (2021). *What is remote sensing and what is it used for?* URL: https://www.usgs.gov/faqs/what-remote-sensing-and-what-it-used?qt-news_science_products=0%7B%5C#%7Dqt-news_science_products (visited on Dec. 3, 2021).

Weisstein, E. W. (2022). *Convolution*. URL: https://mathworld.wolfram.com/Convolution.html (visited on June 18, 2022).

Yang, J., Gong, P., Fu, R., Zhang, M., Chen, J., Liang, S., Xu, B., et al. (Sept. 2013). "The role of satellite remote sensing in climate change studies". In: *Nature Climate Change* 3.10, pp. 875–883. ISSN: 1758678X. DOI: 10.1038/nclimate1908. URL: https://www.nature.com/articles/nclimate1908.

Appendix

A Google Earth Engine Scripts

Two of the GEE scripts will be shown here in full-text, while the rest are available only through a url-link to the GEE site.

A.1 Validation of landslide occurrence and visibility

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3ALS_Verification

```
// Modified time-series script: https://code.earthengine.google.com/9c091e27ca86c3d1e6adbefea8769cc3

//    using cloud filter from: https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S2

// How to use this script:
//   1. Set band description and import table (shapefile) for analysis - can be points or polygons
//   2. Select band from list below and add in SET BAND FOR ANALYSIS

var region = geometry.buffer(2000).bounds();

//###############     1. SET BAND  DESCRIPTION  #################################
var band_description = 'NDVI';

//###############################################################################

// Load imagery.
var s2 = ee.ImageCollection('COPERNICUS/S2')
    .filterBounds(region)
    .filterDate('2015-01-01', '2022-03-16')
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 50)) // Pre-filter to get less cloudy granules.
    .map(function(image) {

//###############     2. SET BAND  FOR ANALYSIS #################################
   var nir = image.select('B8');
                           var red = image.select('B4');
                           var blue = image.select('B2');
                           var green = image.select('B3');

      var ndvi = image.normalizedDifference(['B8', 'B4']).rename(band_description);
      var band = ndvi

//################################################################################
      var qa = image.select('QA60');
      var cloudBitMask = 1 << 10;        // Bits 10 and 11 are clouds and cirrus, respectively.
      var cirrusBitMask = 1 << 11;
      var mask = qa.bitwiseAnd(cloudBitMask).eq(0)  // Both flags should be set to zero, indicating clear conditions.
           .and(qa.bitwiseAnd(cirrusBitMask).eq(0));
      return image.updateMask(mask)
          .divide(10000)
          .addBands(band)
          .select(band_description)
          .copyProperties(image, ['system:time_start']);
    });

print('Size of S2 collection', s2.size());

Map.centerObject(region);
Map.addLayer(geometry, {color: 'gray'}, 'landslide');

var testlandslide = geometry

//Map.centerObject(testPoint, 10)
var chart = ui.Chart.image.series({
    imageCollection: s2.select(band_description),
    region: geometry,
    }).setOptions({
      interpolateNulls: true,
      lineWidth: 1,
      pointSize: 3,
      title: band_description + ' timeseries for a single landslide',
      vAxis: {title: band_description},
      hAxis: {title: 'Date', format: 'YYYY-MMM', gridlines: {count: 12}}
    });

print(band_description + ' chart', chart);

// View cloud free images per year
var s2_best_rgb_17 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2017-06-01', '2017-07-31') // filter for summer 2017
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);

var s2_best_rgb_18 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2018-06-01', '2018-07-31') // filter for summer 2018
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);

 var s2_best_rgb_19 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2019-06-01', '2019-07-31') // filter for summer 2019
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);

var s2_best_rgb_20 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2020-06-01', '2020-07-31') // filter for summer 2020
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);

var s2_best_rgb_21 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2021-06-01', '2021-07-31') // filter for summer 2021
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);

var s2_best_rgb_22 = ee.ImageCollection('COPERNICUS/S2_SR')
    .filterBounds(geometry)
    .filterDate('2021-08-25', '2021-09-26') // filter for september 2021
    .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .clip(region);
```

```

var RGB_params = {bands: ['B4', 'B3', 'B2'], min: 0, max: 3000, gamma: 1.5};

Map.addLayer(s2_best_rgb_17, RGB_params, 'best_RGB_17');
Map.addLayer(s2_best_rgb_18, RGB_params, 'best_RGB_18');
Map.addLayer(s2_best_rgb_19, RGB_params, 'best_RGB_19');
Map.addLayer(s2_best_rgb_20, RGB_params, 'best_RGB_20');
Map.addLayer(s2_best_rgb_21, RGB_params, 'best_RGB_21');
Map.addLayer(s2_best_rgb_22, RGB_params, 'best_RGB_sep21');

print(s2_best_rgb_18);

Map.addLayer(landslides);
```

A.2 Post event – Export training data A

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FA.%20Post_training_1_month_S1_S2

```
// Make Post images for training data

// Input: shapefile points with the following mandatory attributes: 'LS_ID',
↪  'event_date'

//Processing:
  //S1- filter by date, location, polarisation (VV, VH), sensor mode (IW).
  //Apply terrain correction by Vollrath
  //Take median of image stack for each of VV and VH bands.

  //S2 - filter by date and location.
  //take maximum NDVI value for greenest pixel composites, used for dNDVI images.
  //take least cloudy image for RGB and NIR bands.

//Outputs:
 // merges bands from S1, S2 and DEM into final image for each point that can be
 ↪  exported.


// ##########################   User Inputs:
↪  #########################################

// 1. set points - mandatory columns: ['LS_ID, 'event_date']
var fc_points = ee.FeatureCollection(landslides); // Imported landslide point
↪  shape-file with mandatory columns: ['LS_ID, 'event_date']
Map.addLayer(fc_points, {color: 'FF0000'}, 'points');

//2. Set number of months for making composite image
var months = 1;
var my_google_drive_folder = 'earthengine/A_Post'; // Folder name in Google Drive
↪  that files will be exported to.

//###################### IMAGE COLLECTIONS
↪  ###########################################

var S2 = ee.ImageCollection('COPERNICUS/S2_SR'); //Level-2C - but note there is a
↪  problem with over correction of shadows
var S1 = ee.ImageCollection('COPERNICUS/S1_GRD');
var DEM = ee.Image('users/jarnaalexandra/TerrengNorge3').select('elev');
var DEM_all = ee.Image('users/jarnaalexandra/TerrengNorge3').select(['elev',
↪  'slope', 'slope_sum']);

// Visualisation parameters:
var viz_rgb = {bands: ['B4', 'B3', 'B2'], min: 0, max: 3000, gamma: 1.5};

// ######################   FUNCTIONS
↪  ##############################################

// function: create a new feature.
var getPoints = function(feature) {
  var point = feature.geometry();
  // Return a new Feature, copying properties from the old Feature.
  return ee.Feature(point).copyProperties(feature);
};

//function: create region
var bufferPoly = function(feature) {
  return feature.buffer(2500).bounds();
};

  function addS2NDVI(image) {
    var ndvi = image.normalizedDifference(['B8', 'B4']).rename('NDVI');
    return image.addBands(ndvi);
  }

  //Function: add pre and post dates as a property
var addDates = function(feature) {
  var event_date = ee.Date(feature.get('event_date'));
    var keepProperties = ['LS_ID'];
      var pre_date = ((event_date).advance((ee.Number(months)).multiply(-1),
       ↪   'Month'));
  return feature.set({
    post_date : ee.Date((event_date).advance(ee.Number(months), 'Month')),
    pre_date : pre_date,
    event_date : ee.Date((pre_date).advance(ee.Number(months), 'Month'))
  }).copyProperties(feature, keepProperties);
};
  // Topographic correction function
/**
* Radiometric slope correction algorithm for topographic correction
* The library by Andreas Vollrath that has topographic slope correction described
↪   in https://doi.org/10.3390/rs12111867
* This is a copy of Vollrath's script with a bug fix (by Eric Bullock)
*/
var slope_correction = function (collection,
                                options
                                ){
    // set defaults if undefined options
    options = options || {};
    var model = options.model || 'volume';
    var elevation = options.elevation || DEM
    // var elevation = options.elevation || ee.Image(DEM);
    var buffer = options.buffer || 0;
    // we need a 90 degree in radians image for a couple of calculations
    var ninetyRad = ee.Image.constant(90).multiply(Math.PI/180);
    // Volumetric Model Hoekman 1990
    function _volume_model(theta_iRad, alpha_rRad){
      var nominator = (ninetyRad.subtract(theta_iRad).add(alpha_rRad)).tan();
      var denominator = (ninetyRad.subtract(theta_iRad)).tan();
      return nominator.divide(denominator);    }
   // surface model Ulander et al. 1996
    function _surface_model(theta_iRad, alpha_rRad, alpha_azRad){
      var nominator = (ninetyRad.subtract(theta_iRad)).cos();
      var denominator = alpha_azRad.cos()
        .multiply((ninetyRad.subtract(theta_iRad).add(alpha_rRad)).cos());
      return nominator.divide(denominator);    }
    // buffer function (thanks Noel)
    function _erode(img, distance) {
      var d = (img.not().unmask(1)
          .fastDistanceTransform(30).sqrt()
          .multiply(ee.Image.pixelArea().sqrt()));
      return img.updateMask(d.gt(distance));    }
    // calculate masks
    function _masking(alpha_rRad, theta_iRad, proj, buffer){
        // layover, where slope > radar viewing angle
        var layover = alpha_rRad.lt(theta_iRad).rename('layover');
        // shadow
        var shadow =
        ↪   alpha_rRad.gt(ee.Image.constant(-1).multiply(ninetyRad.subtract(theta_iRad))).rena
        // combine layover and shadow
        var mask = layover.and(shadow);
        // add buffer to final mask
    if (buffer > 0)
      mask = _erode(mask, buffer);
    return mask.rename('no_data_mask');  }
  function _correct(image){
    // get image geometry and projection
    var geom = image.geometry();
    var proj = image.select(1).projection();
    // get look direction angle
    var heading = (ee.Terrain.aspect(
      image.select('angle')).reduceRegion(ee.Reducer.mean(), geom,
       ↪  1000).get('aspect')           );
    // Sigma0 to Power of input image
    var sigma0Pow = ee.Image.constant(10).pow(image.divide(10.0));
    // Radar geometry
    var theta_iRad = image.select('angle').multiply(Math.PI/180).clip(geom);
    var phi_iRad = ee.Image.constant(heading).multiply(Math.PI/180);
    // Terrain geometry
    var alpha_sRad = ee.Terrain.slope(elevation).select('slope')
      .multiply(Math.PI/180).setDefaultProjection(proj).clip(geom);
    var phi_sRad = ee.Terrain.aspect(elevation).select('aspect')
      .multiply(Math.PI/180).setDefaultProjection(proj).clip(geom);
    // Model geometry
    //reduce to 3 angle
    var phi_rRad = phi_iRad.subtract(phi_sRad);
    // slope steepness in range
    var alpha_rRad = (alpha_sRad.tan().multiply(phi_rRad.cos())).atan();
    // slope steepness in azimuth
    var alpha_azRad = (alpha_sRad.tan().multiply(phi_rRad.sin())).atan();
    // Gamma_nought
    var gamma0 = sigma0Pow .divide(theta_iRad.cos());
       // models
    if (model == 'volume')
     var corrModel = _volume_model(theta_iRad, alpha_rRad);
    if (model == 'surface')
     var corrModel = _surface_model(theta_iRad, alpha_rRad, alpha_azRad);
    if (model == 'direct')
     var corrModel = _direct_model(theta_iRad, alpha_rRad, alpha_azRad);
    // apply model to derive gamma0_flat
    var gamma0_flat = gamma0.divide(corrModel);
    // transform to dB-scale
    var gamma0_flatDB = (ee.Image.constant(10)
      .multiply(gamma0_flat.log10()).select(['VV', 'VH']));
    // get Layover/Shadow mask
    var mask = _masking(alpha_rRad, theta_iRad, proj, buffer);
    // return gamma_flat plus mask
    return gamma0_flatDB.addBands(mask).copyProperties(image);}
  // run correction function and return corrected collection
  return collection.map(_correct);};
// export function
exports.slope_correction = slope_correction;
//####################################################################################

// Map the point getting, date adding and polygon making functions over the
 ↪  features.
var fc_ROIs = fc_points.map(getPoints).map(addDates).map(bufferPoly); //take
 ↪  points, end up with polygon ROIs with pre and post dates added
var count = fc_points.size();
print ('count', count);

Map.addLayer(fc_ROIs, {}, 'ROIs');
print ('ROIs', fc_ROIs);

//##########################################################################

// function: create post image composites from feature collection
var createPost = function(feature) {
 var region = feature.geometry();
 var post_start = ee.Date(feature.get('event_date'))
 var post_end = ee.Date(feature.get('post_date'))
 var LS_ID = feature.get('LS_ID');

```

```javascript
var S2_post_NDVI = S2.filterDate(post_start, post_end)
    .filterBounds(region)
    .map(addS2NDVI)
    .qualityMosaic('NDVI') //greenest pixel - maximum Ndvi
    .select('NDVI')
    .set({'LS_ID': LS_ID});

// Filter S2 images
var S2_filtered = S2
    .filterBounds(region)
    .filterDate(post_start, post_end)
    // .sort('CLOUDY_PIXEL_PERCENTAGE').first() // gives least cloudy image
    .select('B2','B3', 'B4', 'B8')
    .set({'LS_ID': LS_ID});
var S2_sorted = S2_filtered
    .sort('CLOUDY_PIXEL_PERCENTAGE').toList(S2_filtered.size()); // gives least cloudy image
  var S2_least_clouds_post = ee.List(S2_sorted.get(1));

var S1_post_data = S1.filterDate(post_start, post_end)
  .filterBounds(region)
  .filterMetadata('transmitterReceiverPolarisation','equals',["VV", "VH"])
  .filterMetadata('instrumentMode','equals','IW')
  .set({'LS_ID': LS_ID});

var S1_post_corrected = slope_correction(S1_post_data) //apply terrain correction function to S1_post_data
  .map(function(im) {return im.updateMask(im.select('no_data_mask'))}) // Apply no data mask
  .reduce(ee.Reducer.median())
  .rename(S1_post_data.first().bandNames())
  .select('VV', 'VH')
  .set({'LS_ID': LS_ID});

var merged_post = S1_post_corrected // VV, VH
.addBands(S2_post_NDVI) //NDVI
.addBands(S2_least_clouds_post) // R, G, B, NIR
.addBands(DEM_all) // elevation, slope, slope_sum
.set({'LS_ID': LS_ID})
.toFloat()
.clip(region);

return ee.Image(merged_post);
};

// functions: export images
var exportPostImage = function (image) {
  image = ee.Image(image);
  var LS_ID = image.get('LS_ID');
  var name = 'A_post_' + LS_ID.getInfo();
  var ROI = image.geometry();
  Export.image.toDrive({
              image: image,
              description : name,
              folder : my_google_drive_folder,
              region : ROI,
              scale: 10,
  });
return 0;
};

//##############################################################################
//Map the image creating function over the feature collection
var post_collection = ee.ImageCollection(fc_ROIs.map(createPost)); //for each polygon ROI, make a post image
Map.addLayer(post_collection.mosaic(), viz_rgb, 'post least clouds RGB');
print('post coll', post_collection);

// For loop. getInfo to get properties from server, then Export images
var post_list = post_collection.toList(count, 0);
for (var i = 0; i < count.getInfo(); i++){
  exportPostImage(post_list.get(i));
}
```

A.3 Difference — Export training data A

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FA.%20Diff_training_1_month_S1_S2

A.4 Pre event — Export training data A

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FA.%20Pre_training_1_month_S1_S2

A.5 Post event — Export training data B

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FB.%20Post_training_Jul-Sep_S1_S2

A.6 Difference — Export training data B

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FB.%20Diff_training_Jul-Sep_S1_S2

A.7 Pre event — Export training data B

https://code.earthengine.google.com/?scriptPath=users%2FJohnIF%2FGEE_scripts%3Atraining%20data%2FB.%20Pre_training_Jul-Sep_S1_S2

B Tables

B.1 Verification Table

Table B.1: *Verification of landslides from Landslide Detection Database. For column "Confirmed?": A value of 1 corresponds to 'detected', 0.5 corresponds to 'detected, but only in post image', and 0 corresponds to 'not detected'. See Figure 3.2 for method for verification.*

LS_ID	Confirmed?	Event Date	Comment	Landslide Type	Location
3	1	2019-07-30	Big landslide in Vassenden. Clear NDVI-break. Clear image. Good training data.	144	Årsetelva, Vassenden
4	1	2019-07-30	Visible but not large in extent. Clear NDVI-break. Training data-worthy.	144	Løsetslåtten 2019
5	1	2019-07-30	Small landslide visible in both image and NDVI. OK training data.	144	Løsetslåtten 2019
6	1	2019-07-30	Large clearly visible landslide. Good training data.	144	Strandsvollen, south side of Jølstravatnet
7	1	2019-07-30	Small landslide visible in both image and NDVI. Related to larger LS closeby. OK training data.	144	Slåtten 2019
8	1	2019-07-30	Medium landslide clear in both NDVI and visual	144	Slåttene, Jølster
9	1	2019-07-30	Medium landslide clear in both NDVI and visual	144	Slåttene, Jølster
10	1	2019-07-30	Runout area of landslides 7–9	142	Slåtten
11	1	2019-07-30	Flood slide? Clear in both NDVI and visual, but atypical landslide colour?	142	E39, Vassenden
17	1	2018-05-09	No clear NDVI-break, but values are slightly lower. Landslide is visible.	144	Svatsum
21	1	2018-09-26	Small landslide. Hard to see a break in NDVI, but values are lower following years.	144	Skar ved Liavegen
22	1	2018-09-19	Good training data. Clear in both visual and NDVI. A road goes now in parts of the debris.	140	Slåttene, Førde
23	1	2019-09-16	Quick clay slide. Large enough for training data. Easily detected in both visual and NDVI. (https://www.aftenposten.no/norge/i/4qvEmR/jordraset-i-nittedal-har-skapt-dette-hullet-i-veien-ordfoereren-frykte)	141	Quick clay slide at Li
24	1	2019-08-19	LS follows old landslide path. Observable from both visual and NDVI, but very small. Another undregistered LS slightly south-east of point around same date, maybe a few days later – can be also from avalanche?	142	Småskreden

31	1	2018-08-08	Very small LS, but visible. No NDVI from time of occurance.	144	Skogsvika
33	1	2020-08-23	LS clear to see in both visible and NDVI. Should provide good dNDVI training data. LS follows a narow valley.	142	Tindbekken, between Fjellstad and Tunebrua
34	1	2020-06-06	LS clear to see in visible. NDVI shows clear reduction the next years, but no clear break as event was in early summer.	144	Nergård
35	1	2020-08-23	LS right by 34, but happened later that summer. Clear NDVI-break	144	Haugnes
37	1	2020-10-28	Good training data for small LS. Happened midfall, so no clear NDVI-break.	144	Mundheimsdalen /KVAM
38	1	2021-01-01	LS follows narrow valley and ends on agricultural land. Difficult to notice the landslide from image, but NDVI values are clearly reduced.	144	Skulerud
39	1	2020-12-30	Quick clay slide at Gjerdrum. Too large to not notice	141	Ask, Gjerdrum
40	1	2021-06-14	LS faintly visible but noticeable. Long and narrow.	144	Indre Nordneset
48	1	2020-09-22	Not able to see LS by point, but another LS is visiblie right north of point.	144	Pollen
51	1	2018-05-14	LS is dated to Aug 2019, but from NDVI it might look like it happened in late 2017. News report was from May 2018. Good training data. `https://www.nrk.no/tromsogfinnmark/stort-jordskred-i-finnmark_-_-det-ser-brutalt-ut-1.14050020&144`	Muotkkenjarga	
52	1	2019-06-04	Small LS, but decent training data for small LS. Trivia: Road relocated as a consequence of LS.	144	Berget
54	1	2019-07-30	Several LS in same are occurred in the same summer. Not sure if in same event. NDVI-break is precicely as reported. Same as Jølster event.	144	Torsnesdalen /JONDAL
58	1	2018-07-21	Landslide located 100 m west of point. Clear NDVI-break and landslide scar is clearly visible. regDato is 01.08.2019. Seem plausible.	142	Gjelgrova 2019
63	1	2018-07-30	Can't see reported LS but another LS is seen right above. From NDVI it seem it occurred in the end of July 2018.	144	Sogndal
67	1	2019-07-30	LS easily seen in image. Dosen't show up in NDVI. Area is problably too barren?	144	Strandsvollen, south side of Jølstravatnet

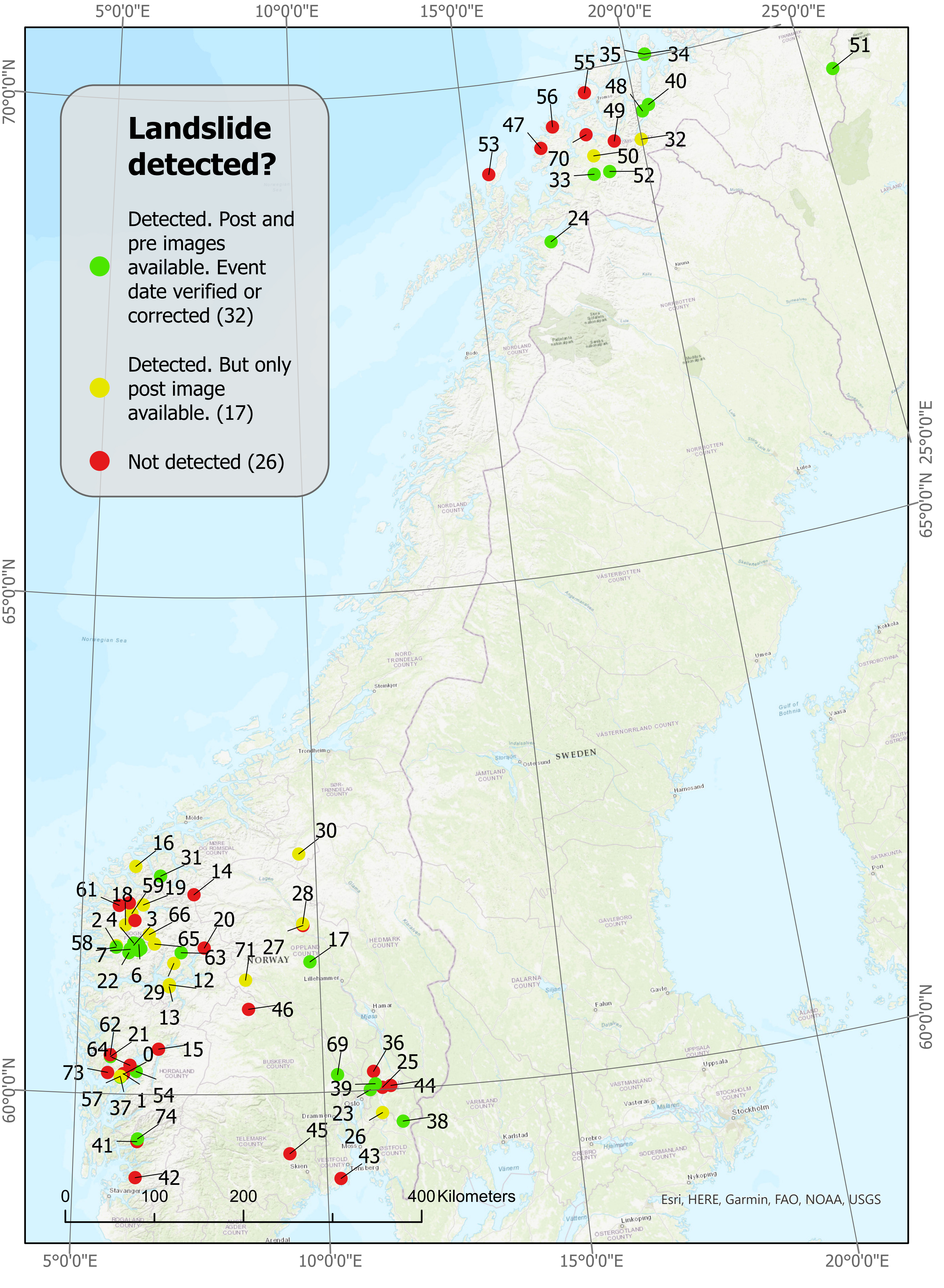
68	1	2019-07-30	From Jølster event. Starts high up, but follows old LS scar further down. 2020 best image has much snow in it.	144	Strandsvollen, south side of Jølstravatnet
69	1	2021-04-29	Main event happened in Nov. 2019. Clear LS cut, but it is clay, so different imprint. NGI-report: https://www.ringerike.kommune.no/contentassets/71e0a5fd65694512a5a829acb4365ae3/befaringsnotat-ngi-28.11.19.pdf	143	Hovsenga, Ringerike
74	1	2019-08-19	New point location from FID 41	144	Correct poin location, Saudafjorden
2	0.5	2019-07-30	regDate is 15.12.2021 and we have not enough data after that event? No clear NDVI-break. No visible sign of landslide. NDVI after 15.12.2021 might show lower than normal values.	142	Gjelgrova 2019
12	0.5	2019-12-29	Landslide too small to notice. Only follows small stream. Went only to ditch of road. NDVI-break noticeable.	144	Sogndal
13	0.5	2019-12-29	Very small impact on image. Slide follows small stream. Most noticeable location is a few hundred meters north of original point.	144	Sogndal
16	0.5	2018-12-31	Original landslide happended in November 2013 (https://www.nrk.no/mr/_-trodde-ikke-dette-kunne-skje-oss-1.11361407). So no difference image will be possible, but quite distinct landslide scar.	142	Årsetdalen Vartdal
18	0.5	2019-07-30	Landslilde is mapped in Geocache basis basemap. Most likely older than 2016, so no difference image available. Clear landslide scar.	142	
19	0.5	2018-12-31	Main landslide is from Nov 2013. This point is related to a flomskred not seen in image or NDVI. Origanal LS useful for training minus difference image. https://www.nrk.no/vestland/ras-stengjer-e39-mellom-eid-og-stryn-1.14360892	142	Skredestranda
26	0.5	2020-02-01	Not able to see in 2020, but in 2021. It might be the leveling after that we see. In that case it will not be possible to see.	143	Kirkebygda Enebakk
28	0.5	2020-02-11	Maybe one visible LS. Long and narrow.	144	Veikledalen
29	0.5	2019-12-29	Old landslide scar visible, but no change observed around date of occurance.	144	Sogndal

30	0.5	2018-05-08	Some small LS-scars observed, but not difference images. Could still be valuable as training data for non-difference images.	142	Hesthåggån
32	0.5	2020-08-24	Old LS scar visible, but not able to see new LS	142	Fv 7928 Signaldalen
50	0.5	2020-05-14	LS is visible, but appears as a small blob close to the road. Not sure if decent training data.	144	Krokseng Bru
57	0.5	2021-04-04	LS too small to spot? Some lower NDVI values after the date, but not confirmed in image.	142	Mundheimsdalen /KVAM
59	0.5	2019-07-30	Some debris detectable in image, but not very characterstic	142	
65	0.5	2019-07-30	Difficult to tell as there are many old landslides in the region. A new LS will not remove any vegetation. Decent non-difference-training data.	144	Strandsvollen, south side of Jølstravatnet
66	0.5	2019-07-30	Again, diffucult to tell as there are a few old LS in the area. Decent non-deifference-training data.	144	Strandsvollen, south side of Jølstravatnet
71	0.5	2021-07-28	LS too recent to be seen properly. Follows old LS scar or stream. Decent training data	144	
0	0	2021-04-04	Date reported (04.15.2021) does not correspond with NDVI-break (29.08.2020). NDVI-break shows up in almost any point, so it might be related to regional lighting. (Sorry my bad, NDVI was set to average of region) No landslide visible.	142	Mundheimsdalen /KVAM
1	0	2021-04-04	Same as FID 0. They are somehow related to the same event, but not visible.	142	Mundheimsdalen /KVAM
14	0	2021-10-20	Clear NDVI-break, but not able to see any change in visual. Most likely a rock fall.	144	Skjåk
15	0	2018-04-26	Unknown date and not possible to see. Might be older than 2018.	144	Tjoflot
20	0	2020-05-22	Landslide not visible in NDVI or image. Follows small stream.	142	Almhol
25	0	2019-12-24	Reported as fairly large quick clay slide, but not visible from imagery.	143	Løken, Lillestrøm kommune
27	0	2020-02-11	Many LS reported, but none visible. Narrow valley with flood slides.	142	Veikledalen
36	0	2020-03-01	Quick clay slide. Not really easy to see from image and very small extent. Leveling sone in 2021yields low NDVI-values	143	Leirbekken, Nannestad kommune

41	0	2019-08-19	News report clearly show LS, but point location is not precise. I think I have found correct location of LS. NDVI-break corrolates to reported date. LS is partly obsured by forest. Hooray, I'm a detective! `https://www.aftenbladet.no/lokalt/i/2GXqbR/flom-og-jordras-saudafjorden-ser-ut-som-glomma-i-glanstiden`	144	Saudafjorden
42	0	2018-09-26	Not verified. Flood slide that is hidden in the forest. No visible change.	142	Eventyrskogen Bønardalen Årdal
43	0	2020-01-06	Most likely very small slide not visible from satellite	141	Østerøyvegen 41
44	0	2020-01-03	Slide due to landfill. Not sure if image show LS or just landfill... not very useable as there's a lot going on in the image.	144	Auli
45	0	2019-08-29	Erosion on agricultural land difficult to spot. No NDVI result.	144	Akkerhaugen
46	0	2019-04-28	LS in alpine resort, so hard to detect any changes. LS mght be very small.	144	Såhaugløypa
47	0	2021-02-28	There has been a large landslide <1000 m^3, but for some reason it is not easy to see in image.	142	Sifjordura øst
49	0	2020-09-22	LS do not show up in image. Not so large and in a residential area.	144	Storsteinnes
53	0	2019-04-23	Image too "messy" to spot the LS.	144	Sandvika
55	0	2021-02-28	No LS spotted in image. Reported as blocking the road.	142	Skjellvika
56	0	2021-02-28	LS not visible in image. Reported as blocking the road. Probably too small to leave a scar	142	West of Kvalvika
60	0	2019-07-30	LS too small to spot? Some forestry in the area will make it difficult to be sure on what is LS and not. Landslidee is probably induced by forestry activity.	142	
61	0	2019-07-30	No LS spotted in image. Runs over an old landslide.	142	
62	0	2018-09-26	Not able to see. What looks like a landslide is most likely forestry. Report say that LS not visible from aerial image.	144	KV Liavegen /SAMNANGER
64	0	2019-08-29	Can't really see any LS in this area even though it was very heavy rain that night.	144	Fosso /KVAM
70	0	2020-05-14	No clear LS. It has most likely went in a small stream, but not large enough to remove fresh vegetation.	144	Krokseng bru, Målselv
72	0	2018-12-31	Not able to spot LS. Too small?	144	Skredestranda

73	0	2021-04-04	Not able to spot LS. Too small?	142	Mundheimsdalen /KVAM

C Maps


Landslide detected?
● Detected. Post and pre images available. Event date verified or corrected (32)
● Detected. But only post image available. (17)
● Not detected (26)
0 100 200 400 Kilometers
Esri, HERE, Garmin, FAO, NOAA, USGS
5°0'0"E
10°0'0"E
15°0'0"E
20°0'0"E
25°0'0"E
70°0'0"N
65°0'0"N
60°0'0"N
NORWAY
SWEDEN

John Isak Furuseth

Automation of landslide detection using Deep Learning

NTNU
Norwegian University of
Science and Technology