Interdisciplinary Project

Detecting deforestation using Sentinel-1 SAR data and deep learning

Hasret Gümgümcü

June 21, 2018

Institute of Geodesy and Photogrammetry
ETH Zurich

Professorship
Prof. Dr. Konrad Schindler

Supervisors
Dr. Jan Dirk Wegner
Andrés Rodríguez Escallón

Abstract

The goal of this project is to analyze if sentinel-1 C-SAR images are a good source to detect deforestation with the use of deep learning in wooded regions. We developed a deep learning network which takes two sentinel-1 SAR images from the same region but different dates as input data. With the ground truth of the same region, we were able to train the network pixel wise to detect deforestation between those two dates with around one year time difference (March 2016 & June 2017). We treat the two images as two channels for training the network.

Although being limited in labeled training data we were able to show that sentinel-1 SAR images are an optimal source to detect deforestation because of its advantage being all weather resistant and not affected from clouds or other environmental conditions.

Acknowledgement

I would like to thank the following people:

- Prof. Dr. Konrad Schindler, for making this project possible.
- Dr. Jan Dirk Wegner, for his support and valuable input during the meetings.
- Andrés Rodríguez Escallón, for his continuous help in technical questions as well as new ideas to realize.

Contents

1 Introduction

Nowadays there is a huge change for remote sensing applications with the rise of deep learning in the last few years and with the freely available satellite imagery. Since the beginning of the sentinel-1 mission in 2014, we are able to collect high precision SAR images around the entire globe with a revisiting time of just a few days. The advantage of SAR images being weather independent and providing continuous usable data (not affected from cloud or other environmental effects) makes it an ideal source for monitoring applications. New earth observation satellite missions starting in near future will generate even more data which will be ready to use for research and development.

A big issue that we have everywhere around the globe is the permanent deforestation especially for illegal activities. Since some government lack of resource to discover illegal deforestation, earth observation data can be used.

To solve this issue we define the aim of our project:
Analyze if sentinel-1 SAR images are an appropriate source for detecting deforestation with deep learning in wooded regions.

We will use a deep learning network to train with sentinel-1 SAR images for detecting deforestation in a period of time. This gives us the advantage not only to expose deforestation but making it possible to find out when this particular event took place.

2 Theoretical Principles

2.1 Synthetic Aperture Radar (SAR)

All information given in this section is based on Alberto Moreira et al. (2013).
SAR imagery for earth observation is already used for more than 30 years. In the 80s, it became the follower of SLAR (side-looking airborne radar) which was already used in the 50s for imaging radars. SLAR operates with other methods, which leads to a moderate azimuth resolution (azimuth is defined as the flight direction). It was used mainly by military research for reconnaissance purpose and man-made target detection.
The big advantage of radar is its continuous data collection under all weather conditions. It is independent from daylight and cloud coverage.

2.1.1 Basic SAR principles

SAR systems have in general a side-looking imagining geometry. It is based on a pulsed radar with a forward movement mounted on a moving platform. The system sends electromagnetic pulses and receives the echoes of the backscattered signal in a sequential way. Pulse repetition rate, as well as the swath width can vary a lot for different systems. The measured data is actually the amplitude and phase of the backscattered signal. This is affected from the physical and electrical circumstances. Different frequency bands can be used depending on the purpose of the measurement. The penetration depth varies for each band. Longer wavelengths (see Table 1) can penetrate more in the media of the surface. The most used frequency bands for SAR are L-, C-, and X- band.

Frequency Band	**Ka**	**Ku**	**X**	**C**	**S**	**L**	**P**
Frequency [GHz]	40-25	17.6-12	12-7.5	7.5-3.75	3.75-2	2-1	0.5-0.25
Wavelength [cm]	0.75-1.2	1.7-2.5	2.5-4	4-8	8-15	15-30	60-120

Table 1: Comparison of frequency bands for SAR systems

The collection of backscattered echoes happens during the movement of the platform and therefore consecutive time of reception translates into different positions. This effect allows to construct a virtual aperture that is much longer than the physical antenna length. During the pulse time τ is the amplitude of the transmitted waveform constant whereas the frequency varies with the time according to $f_i = k_r \cdot t$ where k_r is known as the chirp rate (chirp signal is a frequency modulated waveform for the transmission). The bandwidth

$B_r = k_r \tau$ can also be derived with this information.
The distance between the moving radar (with constant velocity v) and a point on the ground can be described at anytime t during the illumination with the following equation:

$$r(t) = \sqrt[2]{r_0^2 + (vt)^2} \approx r_0 + \frac{(vt)^2}{2r_0} \qquad \text{for } vt/r_0 \ll 1 \tag{1}$$

$$r(t_0) = r_0 = \sqrt[2]{(H - \Delta h)^2 + x_0^2}, \tag{2}$$

where H is the platform height as shown in Fig 1. We can simplify equation 1 with the fact that in general the distance r_0 is much larger than vt with a Taylor series. For highly accurate SAR data processing the complete equation should be considered.

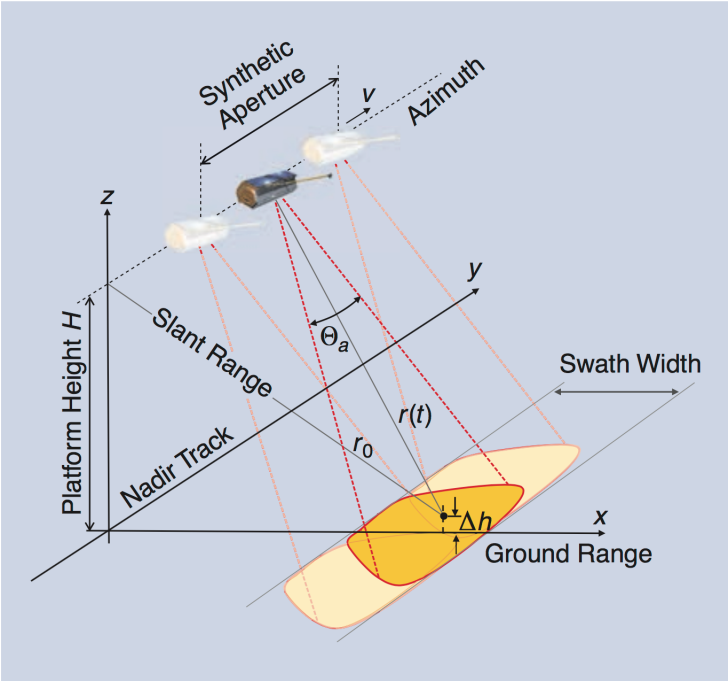


Figure 1: SAR imaging geometry.

Image resolution
Imaging radar has the following two resolutions:

- *Slant-range resolution* δ_r: Given the bandwidth B_r and speed of light c_0 we can define it with the equation $\delta_r = c_0/2B_r$.

- *Azimuth resolution* δ_a: The construction of the synthetic aperture (path length during which the radar receives echo signals) defines $\delta_a = r_0\Omega_{sa}$ where Ω_{sa} is a narrow virtual beamwidth. It can also be written like $\delta_a = \frac{d_a}{2}$ with d_a is the length of the antenna. The last equation shows how high the azimuth resolution can be achieved with synthetic aperture.

Signal processing

For getting visually useful data, signal processing steps needs to be done. Two main steps are important as shown in Fig 2. For computational reason a multiplication in the frequency space can be done which represents a convolution in the time domain along both dimensions (range and azimuth). The range compressed data results after the convolution with the range reference function (depends on the transmitted chirp waveform). To get the image data another convolution with the azimuth reference function (depends on the geometry and changes from near to far range) needs to be done as a last step. Analog to the Doppler effect the azimuth frequency is also called Doppler frequency.

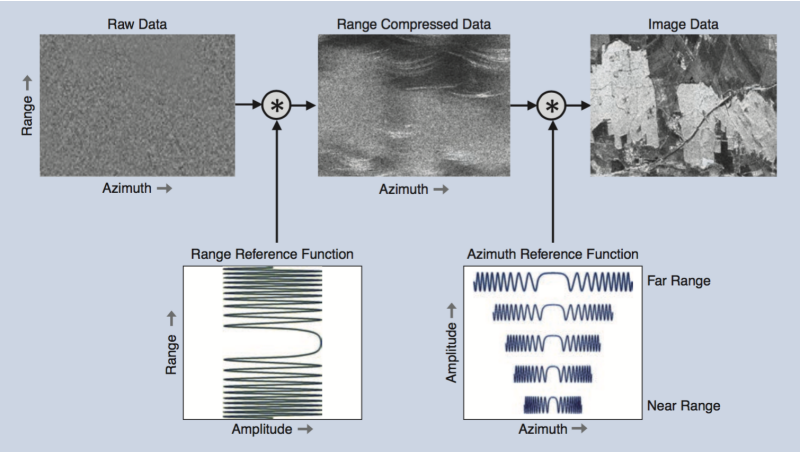


Figure 2: SAR processing steps for getting visually useful output of the raw data

The image data after processing represents the reflectivity with the intensity values transmitted from the measured ground points. Two following steps needs to be applied to the image data:

- *Calibration*: It calibrates the image data to radar backscatter coefficient σ_0. This step requires a DEM (Digital Elevation Model) image to accurately calculate the incidence angle over the image. The radar backscatter coefficient values are given in decibels (dB) [Harris (2018)].

- *Geocoding*: SAR images are geometrically distorted. This distortion depends on different parameters like the particular side-looking geometry and on the magnitude of the undulation of the terrain's surface. Radar measures the projection of a three-dimensional scene on the radar coordinates (slant-range and azimuth). To make it useful for comparing with other types of data it is important to correct it geometrically in order to integrate it with other types of data (satellite images, maps, etc.). A DEM is used to create a grid map for projecting the SAR image to the grid and locate correctly the positions of the pixels [ESA (2018b)].

Speckle

Elemental scatterers with a random distribution can occur within a resolution cell. This causes the speckle. From one cell to other due to the strong fluctuation of the backscattering (sum of their amplitudes and phase results). This leads to the fact, that intensity and the phase in the final image are no longer deterministic. It follows instead an exponential and uniform distribution. Generally speaking, we can say that speckle is a physical measurement at sub-resolution level. Although it is often seen as noise, it can not be reduced by increasing the transmit signal power because its variance increases with the intensity. An approach to reduce it is called *multi-look*. It is a non coherent averaging of the intensity image. Even though it reduces the resolution of the image it increases the visual interpretability strongly (see Fig 3) until some point. With higher multi-look the entire image gets smoothed a lot and this can result to bad quality.

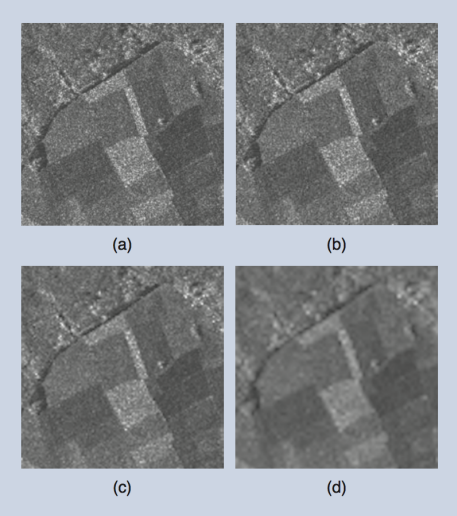


Figure 3: (a) without multi look, (b) - (d) multi-look variations of 2x2, 4x4 and 8x8. Higher multi-look reduces the speckle but at the same time worsens the resolution of the image (more smoothed)

Operation modes

A useful feature in SAR imagery is that by controlling the antenna radiation pattern it can process with different operating modes. This can be achieved by dividing the antenna into sub-apertures and controlling the phase and amplitude of each sub-aperture through transmit/receive modules. Working methods of different operation modes are explained in section 2.1.2 in detail.

2.1.2 Sentinel-1 mission

All information given in this chapter is taken from ESA (2018c) if nothing else is quoted. Sentinel-1 mission is the first mission that ESA (European Space Agency) started for the Copernicus program [ESA (2018a)]. Two of the planned four satellites are already in space (Sentinel-1A launched 2014 & Sentinel-1B launched 2016) and are collecting open source data.

Overview

Four different imaging modes are operated with C-band (see Table 1) with different resolutions (down to 5m) and coverage (up to 400km). The most important facts are that it has a really short revisiting time (only a few days) and it has dual-polarization capabilities and rapid product delivery. Due to the fact that, SAR imagery is not affected of cloud coverage or lack of illumination and can provide data during day and night under all weather conditions it can offer reliable, repeated wide area monitoring.
The mission is especially designed to work in a pre-programmed, conflict-free operation mode. Its imaging the entire globe to ensure a consistent long term data archive built on applications based on long time series.
Some of the services which will benefit from this mission are:

- monitoring of arctic sea-ice extent
- routine sea-ice mapping
- surveillance of the marine environment, including oil-spill monitoring
- ship detection for maritime security
- monitoring land-surface for motion risks
- mapping for forest, water and soil management
- mapping to support humanitarian aid and crisis situations.

Data acquisition

As already mentioned before, sentinel-1 satellites are carrying a single C-band synthetic aperture radar instrument operating at a center frequency of 5.405 GHz. The C-SAR (SAR operating with C-band) instrument supports dual polarization (HH+HV or VV+VH) implemented through one transmit chain (switchable to H or V) and two parallel receive chains for H and V polarization.

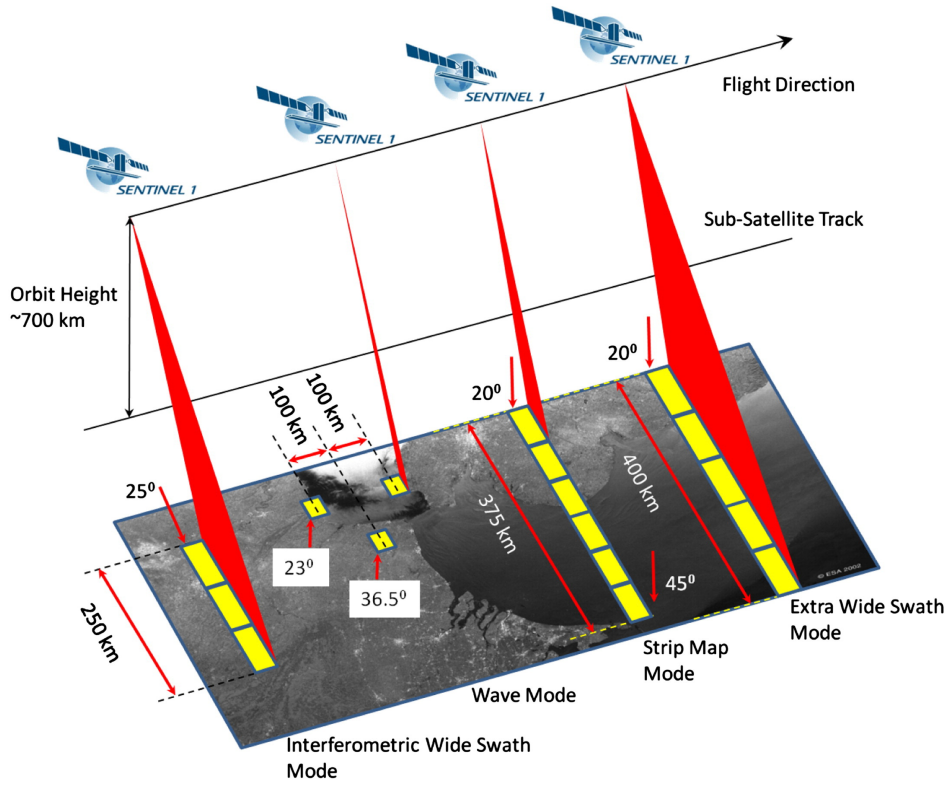


Figure 4: Sentinel-1 provides four different acquisition modes: Interferometric wide swath (IW), wave (WV), strip map (SM) and extra wide swath (EW). They all have different functionalities and usage.

The SAR instrument operates in one of four modes as shown in Fig 4:

- *Strip Map (SM)*: It provides coverage of a narrow swath width of 80 km with a resolution of 5 m by 5 m. The range coverage is 375 km for six overlapping swaths. These can be selected individually by changing the beam incidence angle and the elevation beamwidth. This mode will only be operated on request for extraordinary situations.

- *Interferometric Wide Swath (IW)*: It is the default acquisition mode over land. A swath width of 250 km is measured with a geometric resolution of 5 m by 20 m. For generating the three swaths, it uses the TOPSAR (Terrain Observation with Progressive Scans SAR) technique. The beam is electronically steered from backward to forward in the azimuth direction for each burst to get higher quality image. It ensures homogeneous image quality throughout the swath. Due to the overlap in the azimuth and elevation domain by Doppler and wave number spectrum, interferometry is ensured. An important requirement is high accuracy for image co-registration. Because a small co-registration error in the azimuth can introduce an azimuth phase ramp.

- *Extra Wide Swath (EW)*: It is mainly important for applications where wide coverage is important. The workflow is similar to the IW where, instead of three swaths, five swaths (400 km) are used which results in a lower resolution of 20 m by 40 m. It could also be used with interferometry similar to IW. This method is intended for maritime, ice and polar zone operational services where wide coverage is of big importance.

- *Wave (WV)*: As the name already reveals, this model can help to analyze waves, especially for its direction, wavelength and height on the open ocean. It is the only mode which does not support dual polarization.

Center frequency	5.405 GHz
Incidence Angle Range	20° - 46°
Look direction	right
Antenna size	12.3 m x 0.821 m
Azimuth beam width	0.23°
PRF (Pulse Repetition Frequency)	1 000 - 3 000 Hz (programmable)
Data quantization	10 bit

Table 2: Important parameters of the Sentinel-1 instrument

Data products

All products of the Sentinel-1 missions are free of charge and can be used for different purposes. The output product for the user can vary and should be chosen depending on the usage (see Fig 5). Data delivery time depends on the area. This can be within an hour (for near real time emergency response), three hours for priority areas and 24 hours for systematically stored data.

The product range is divided in the following groups which is also illustrated in Fig 5:

- Level 0: It is the basis product (unfocused and compressed raw SAR data) where all other product levels are based on.
- Level 1: The product level is distinguished between Level 1-SLC (single look complex) and Level 1-GRD (ground range detected) where both are already focused. It is intended for the most data users. The main difference between them is that SLC product preserves the phase information, is geo-referenced using orbit and attitude data from the satellite and is provided in zero-Doppler slant-range geometry. On the other side GRD is multi-looked and projected to ground range using an Earth ellipsoid model. GRD products can either be full, high or medium resolution. This depends only upon the amount of multi-look performed.
- Level 2: It is generated only from strip map (SM) and wave mode (WV). It includes components for ocean swell spectra (OSW), ocean wind fields (OWI) and surface radial velocities (RVL).

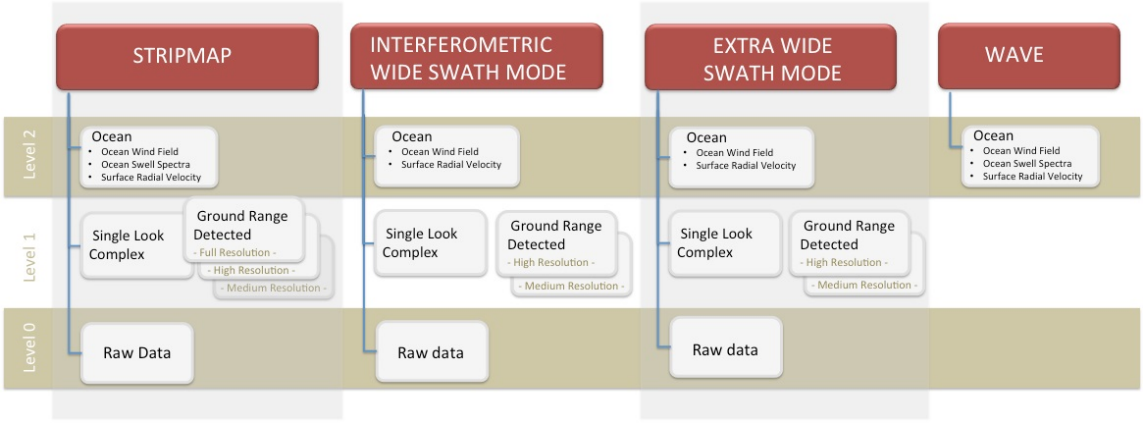


Figure 5: Sentinel-1 products are divided in three levels. Level 0 are the unprocessed raw data. Level 1 contains single look complex (SLC) and ground range detected (GRD) data. Level 2 are Ocean (OCN) products.

2.2 Convolutional Neural Network (CNN)

All information given in this section are based on Karpathy (2018) if nothing else is quoted. The beginning of neural networks was not inspired by engineering a good machine learning task rather by the goal of modeling biological neural systems. All the tasks that human can master are due to biological neurons in the brain. The idea of modeling the functionality of a biological brain is not far-fetched.

2.2.1 General idea

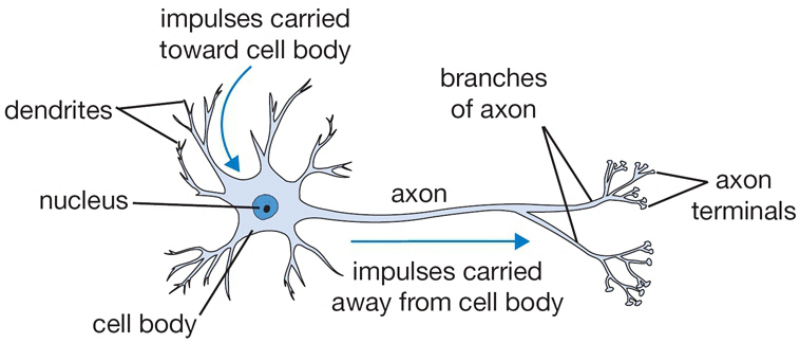


Figure 6: Biological neuron which can send information through the axon to the next neuron

In Fig 6 we can see how artificial neural network has been inspired in its beginning. The same principle is applied for artificial neurals where in a layer each cell can send its information stored to the next layer and so on. With this principle a network can be constructed called artificial neural network.
Deep learning is commonly used for terminology. The term deep is based on the layer wise structure of the network as shown in Fig 8. Learning refers to the function of the network which has the aim to learn given tasks from the input data sending to the network. For this reason, it is a subpart of machine learning which in turn is a field of artificial intelligence (AI). The goal is to build a network which works similar to a human brain. Fig 7 describes the mathematical representation of a biological neuron.

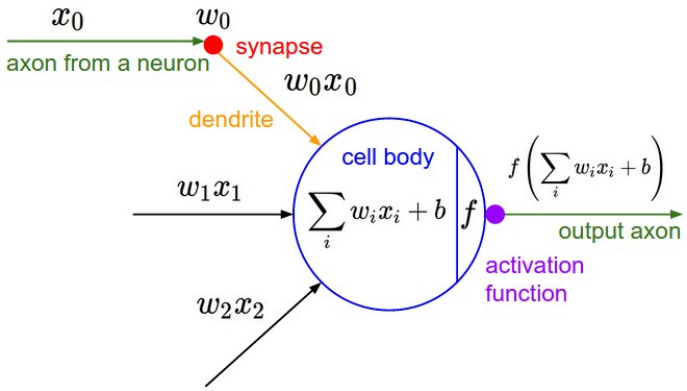


Figure 7: Mathematical representation of a biological neuron. Input data from a previous neuron is sent by the axon. In the cell body are all incoming data put together and through an activation function send to the next cell trough the output axon.

Many cells putting together build a layer. Each of them gets all the information from the previous layer and does some computations and sends it to all cells in the next layer. All neurons in a single layer function completely independently and do not share any connections. In Fig 8 we can see how a neural network can look like.

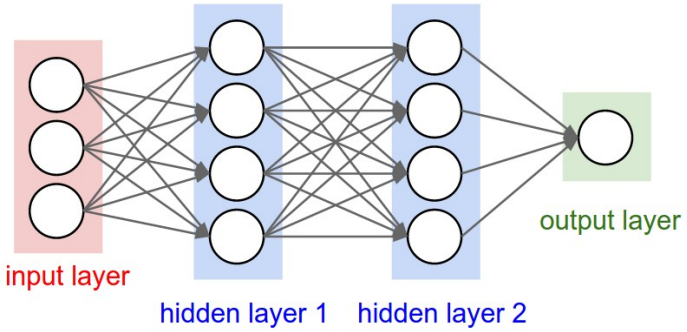


Figure 8: Example of a neural network build of one input layer (input data), two hidden layers (where learning takes place) and an output layer (output data what the network predicts).

Convolutional neural networks indeed are neural networks which use a convolution. This network allows to take an image as input and learn features from this particular input and optimize the weights of all cells by a predefined function. After training the network with sufficient training data, it is able to do identify features on the image. The workflow is as followed:

- We define a network architecture.
- The goal or task that the network should solve is set (e.g. classification, detection, matching, etc.).
- Training algorithm with training set and its label.
- Validation of the algorithm by giving data to the predefined algorithm and see if it achieves a proper prediction.

Fig 9 illustrates a simplified graphic of a CNN network. For the input dataset (in this case mode of transport) the labels needs to be predefined in advance, so that the algorithm knows to which label the input corresponds to. Extracted information gets with each following hidden layer more abstract. The last hidden layer is a true probability for each class in the dataset. The prediction is the class with the highest probability. This is just one example how deep learning can process. Nevertheless, nowadays more improved methods are developed.

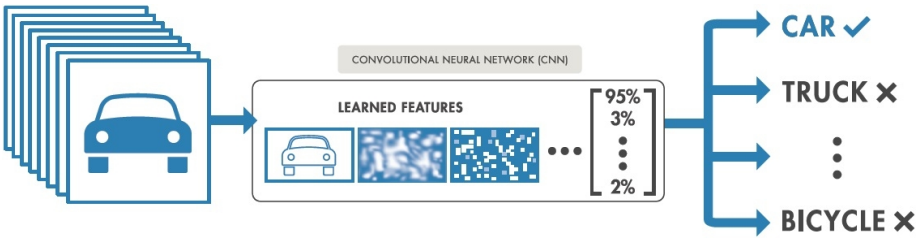


Figure 9: An easy illustration how a CNN works. It gets different mode of transport as input image and tries to classify them by learning features of different levels to gives a corresponding probability for each class.

2.2.2 Architecture

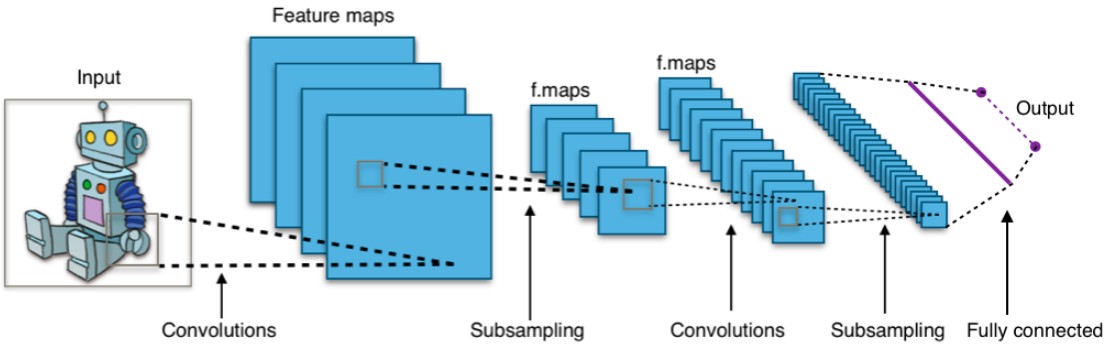


Figure 10: The input image gets convoluted first to get the feature maps. For subsampling the map size, pooling operation is used. After doing this process a second time we can generate a fully connected layer which will result in the output layer (fully connected) [MathWorks (2018)].

Fig 10 shows some of the main layer types used in a CNN architecture.
We will discuss in this section the functionalities and process of each layer type. We focus on the following types:

- Convolutional layer
- Pooling layer
- Fully-connected layer

Convolutional Layer

The main difference between a neural network and a convolutional neural network are the convolutional operations. The heavy computation of the algorithm is done in this layer. It consists a set of learnable features (size of the feature is a parameter to be set beforehand). Every filter is small spatially and slides through the previous layer cell by cell and saves the new values in the convolutional layer. Because each filter will produce a 2-D map usually convolutional layers have besides the two axis (height and width) also a third axis the depth which depends to the number of filters.
With high-dimensional inputs like images it is not efficient to connect all neurons to all neurons. This will lead to computational issues and is also unnecessary. Local information are in images of big importance to extract features. The receptive field of a neutron is

defined by its spatial extension. Its local connectivity is restricted on the 2-D map (width and height) but includes in the depth the entire range.

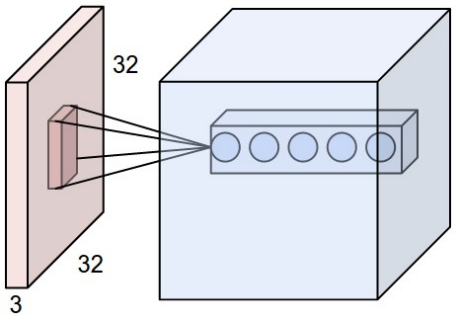


Figure 11: Each neuron in the convolutional layer is connected only to a local region in the input volume

To get the output after the convolution we can look at Fig 12. In this example a (3×3) Sobel filter is used. The convolution filter can be defined in values and size individually. Additional parameters can be set like zero-padding or stride. Zero-padding is used when the source image should be padded with zeros. Since not otherwise defined the filter will slide pixel by pixel and compute the filtering. If not every pixel (with its surrounding) should be filtered stride can be defined where we can set a value to skip a specific amount of pixels. In our example the output size of the map will be $(n-2) \times (n-2)$ where n is the size of the source map (we assume a quadratic source map). The convolution makes the most computational cost of a network. This results that, with each additional convolutional layer or increasing filter size the amount of parameters will be increased highly.

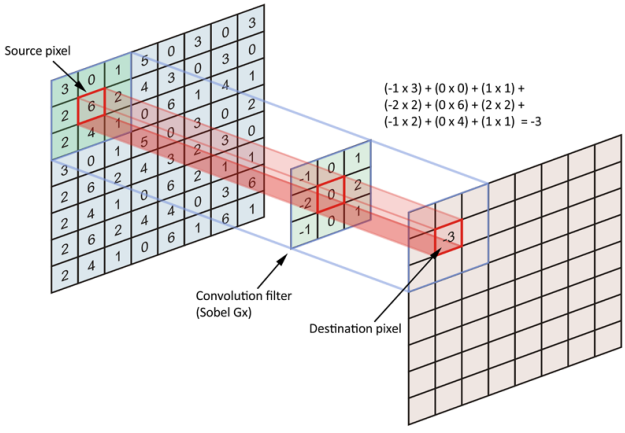


Figure 12: Example of a convolution with a 3x3 filter sliding through the map

Pooling layer

It is common to use a pooling layer to reduce the size of the layer (height and width). This is useful for efficiency and decreasing parameters. Another reason is to control overfitting. It is usually used between following convolutional layers. In theory pooling layer can be anything which reduces the size but practically most used is the max-pooling operation. This is in general a 2×2 filter which works for every part of the image independently. The depth size remains same as before. For the width and height it looks at a window size of 2×2 (for a 2×2 max-pooling filter) and takes the highest value of all four parameters. This process is done for the entire layer. A basic example how the output looks like after max-pooling operation is shown in Fig 13.

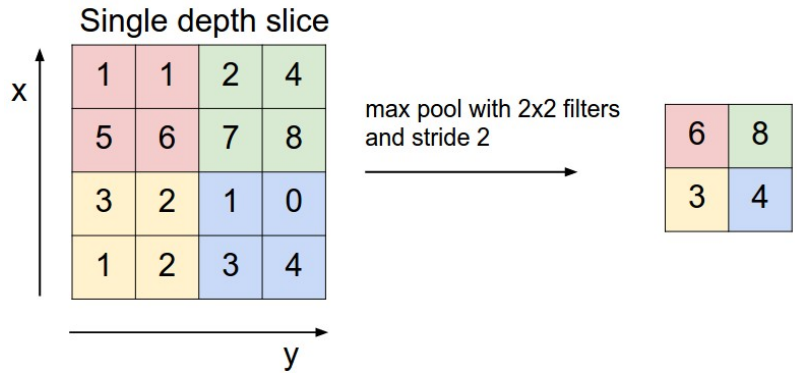


Figure 13: Input layer with size 4×4 is downsampled to 2×2 after max-pooling with filter size 2×2.

Fully-connected layer

A fully-connected layer is characterized that all neurons have fully connection to all activations in the previous layer as it is known from a regular neural network described in the previous sections. Nevertheless, the function form of a fully connected layer is identical with a convolutional layer. A fully-connected layer is used usually in the end of the network for downsampling it to the size of the input classes (especially for classification tasks).

Rectified linear unit (ReLu)

Rectified linear unit (ReLu) is an activation function used mostly after a convolutional layer and before a pooling layer. There are many other activation functions but in the last few years ReLu gained big popularity. The activation is simply thresholded at zero. This means only positive values are considered for the activation. The basic mathematical function for that is the following:

$$f(x) = max(0, x) \tag{3}$$

Fig 14 shows the ReLu in a graphical way.

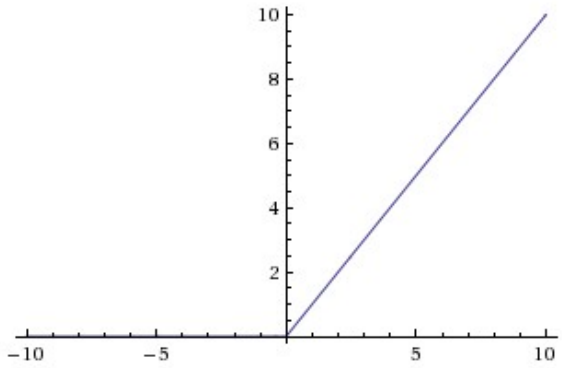


Figure 14: Rectified linear unit (ReLu) is a widely used activation function for CNN. Its threshold is at the value zero.

There exist different pros and cons for using ReLu. It speeds up the convergence of the optimizer (compared to sigmoid/tanh activation function) and instead of expensive operations (exponentials, etc) it uses only a threshold. A negative point is that it is sensitive to the learning rate of the network. If the learning rate is set too high some neurons can be "*dead*". This means that they are never active across the entire training dataset. One attempt to fix this issue is to use leaky ReLu. Instead of the function being zero for $x < 0$ it will have a small negative slope.

Loss function

One of the most important functions which can tell something about the result quality of the training the network as well as for the validation is the loss function. It is generally speaking calculated with penalizing the prediction after comparing it with the true label and giving each penalty a weight. The sum of that for each training step can be outputted as a loss. If the network is set appropriate to the dataset the training loss should decrease

with each epoch (repeating training with the same training set multiple times). This indicates that the network learns from the training data. Fig 15 shows the influence of the learning rate (a hyperparameter that can be set and tells the network how fast it should adapt to new learned features) to the loss function. Since the goal is to let the loss function converge to zero, a high learning rate will converge in a higher value (local optimum) and a low learning rate will decrease slowly. It is important to find an appropriate learning rate by looking at the loss function.

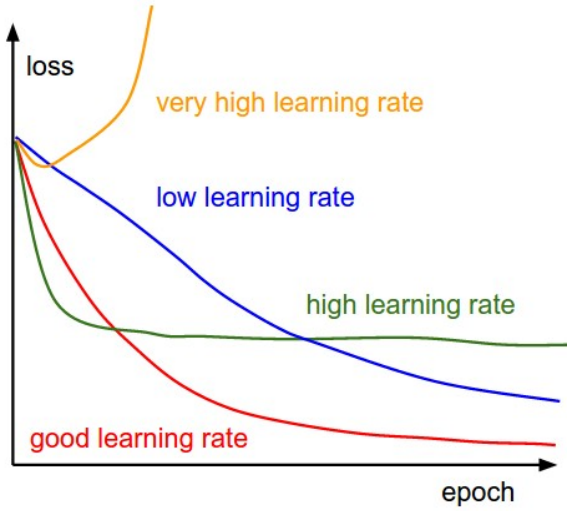


Figure 15: Curve of the loss function depending on the learning rate. With an optimal learning rate the loss function should decrease in the beginning faster and converge to zero

Although the most important concepts and parameters are explained in this chapter there are still other important concepts like hyperparameter tuning, regularization, dropout or optimizer which are not explained in detail in this project.

3 Methodology

In this section we will show our main product by going through the following topics:
The dataset we chose, the pre-processing steps we took, the network we built and the post-processing steps we applied.
As mentioned in the introduction, the main product is a deep learning network, built with tensorflow in python, for detecting deforestation in a specific time period. To realize this, we chose a specific region where deforestation takes place. With the sentinel-1 mission we get two different C-SAR images (see section 2.1.2) from the same area but different dates. To detect deforestation in between those two dates we need to get the ground truth, where deforestation actually took place. For doing that, we use the time slider function from Google Earth Pro to switch around both dates and generate manually the ground truth. After collecting the ground truth over the entire map we feed the training data with the label to the algorithm and validating it afterwards. The result is a binary map where each pixel is labeled either as deforested (value 1) or background (value 0).

3.1 Dataset

For the dataset we need to define some conditions:

- Needs to be in a forested area where deforestation took place recently (in the last few years and is still continuing).
- Since we will generate the ground truth by manually labeling in Google Earth Pro, the area on the satellite image in Google Earth Pro needs to be cloud free, so that we can label properly.
- Date difference must be enough big, to collect appropriate data of deforested areas.

Considering the conditions listed above, northwestern Oregon, (USA) near Portland, turns out to be well fitted. According to local newspaper (Oregonlife) (2018), in western Oregon has over 2000 km^2 forest cover disappeared since 2000. In Fig 16 is the exact location of the dataset shown. The dataset extends to a region around 900 km^2. The two different dates are March 2016 and June 2017. Those are based on the dates which the time slider in Google Earth Pro is able to show.
Both SAR images are level 1 - GRD products with dual polarization (VV + VH). For this project we use in both cases only the VH polarization and ignore the VV polarization. The acquisition mode for the images are Interferometric Wide Swath (IW), which is the over land default mode (see 2.1.2). Each level 1 - GRD image has a size between 1.5 - 2.0 GB.

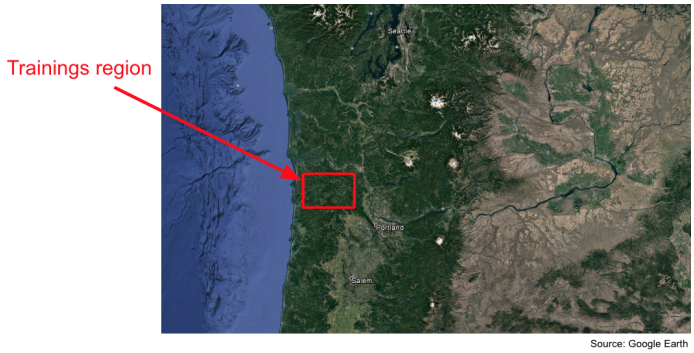


Figure 16: Dataset is northwestern Oregon, (USA) near Portland.

Ground truth

As mentioned before we generate the ground truth manually by labeling polygons where deforestation took places between the chosen two dates (March 2016 and June 2017). This is done by switching between the time slider feature in Google Earth Pro. When an area is discovered where deforestation took place from March 2016 to June 2017 it should be labeled as deforested. The feature "Add a polygon" in Google Earth Pro can be used for this to define polygons. How this process looks like is shown in Fig 17 where three polygons are labeled as deforested.

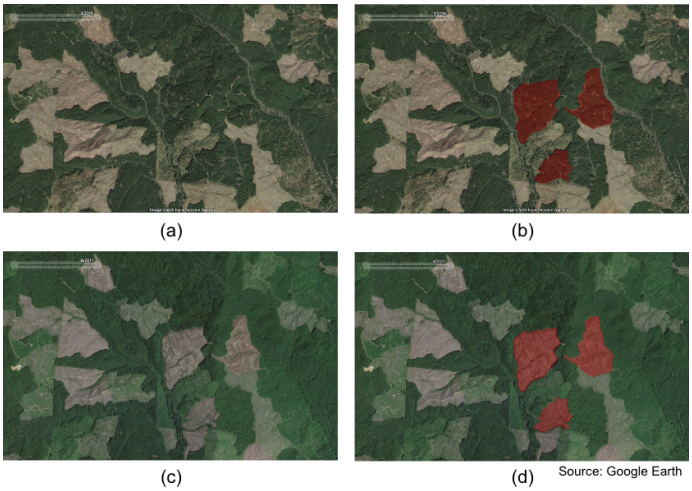


Figure 17: An example of how the ground truth is generated. (a) Satellite image of a small region in March 2016 and (c) the same region in June 2017. (b) and (d) polygons labeled as deforested (in red) overlay to the satellite image.

By continuing with this process over the entire dataset which is 5250 $\times$ 1680 pixel big, we get a binary map with labels deforested (value 1) or background (value 0). One problem

that occurs after labeling is that only 2.8% of the pixels in the ground truth map have actually the label deforested. The remaining 97.2% are background (see Fig 18).

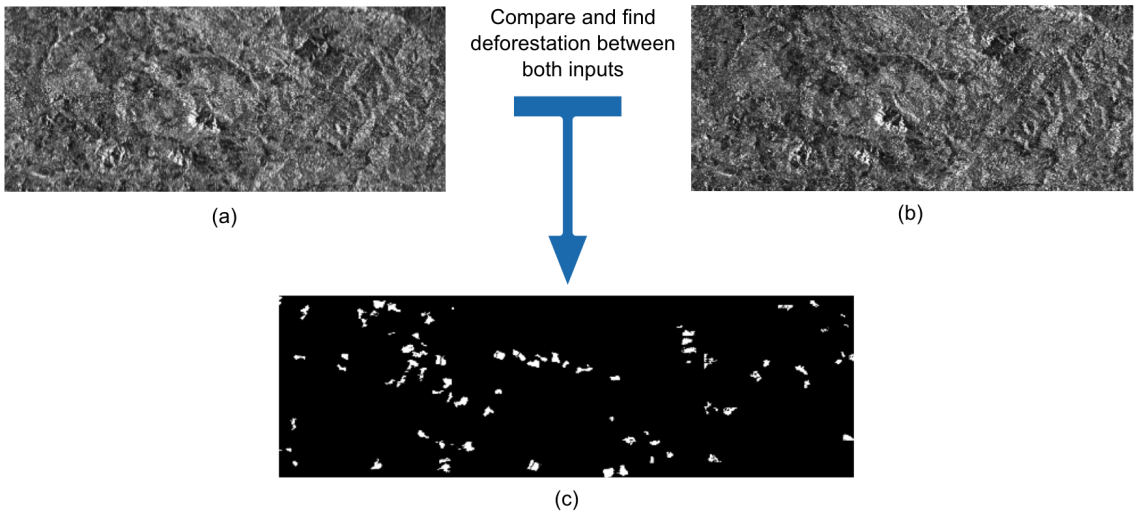


Figure 18: (a) SAR image of March 2016; (b) SAR image of June 2017; (c) Ground truth label (white: deforested; black: background)

3.2 Pre-Processing

As mentioned in section 2.1.1 the raw sentinel-1 data are in SAR geometry. We need two main pre-processing steps before we can use it. This is important because we are generating ground truth based on Google Earth projection system. For that reason, it is important to have the same geometry for the SAR images because we need to overlay the ground truth with the SAR images for feeding it to the network in a later stage. For all the pre-processing steps we use SNAP (Sentinel Application Platform) [ESA (2018d)]. It is a toolbox for processing sentinel-1 images with in-build functions.

Radiometric calibration

The process described in this section is based on Nuno Miranda (2015).

The first pre-processing step is called radiometric calibration to convert the radar reflectivity into physical units. The radar reflectivity contains a real and an imaginary part. L_1 products of sentinel-1 provides so called Calibration Annotation Data Set (CADS) which contains a Look Up Table (LUT) A_σ to transform the radar reflectivity into radar cross-section σ^0 where the area normalization is aligned with ground range plane. The Earth model used is the ellipsoid inflated with an average height such that the normalization

factor can be simplified to $sin(\alpha)$ where α is the local incidence angle of the Earth model used. To calculate the radar cross-section σ^0 we use the following equation:

$$\sigma^0 = \frac{DN^2}{A_{dn}^2 \cdot K} \cdot \frac{1}{G_{eap}^2} \cdot \left(\frac{R}{R_{ref}}\right)^3 \cdot sin(\alpha) \tag{4}$$

where

- $\frac{1}{G_{eap}^2}$ is the elevation antenna pattern (EAP) correction (2-way)
- $\left(\frac{R}{R_{ref}}\right)^3$ is the range spreading loss (RSL) correction
- A_{dn} is the product final scaling from internal GRD to final GRD
- α is the local incidence angle
- K is an absolute calibration constant.
- DN is the pixel amplitude directly taken from the measurement file

In the case of sentinel-1, the EAP and RSL corrections are by default applied such that the above formula simplifies to:

$$\sigma^0 = \frac{DN^2}{A_{dn}^2 \cdot K} \cdot sin(\alpha) \tag{5}$$

As defined in the product specification we can also simply use the LUT with the following relation:

$$\sigma^0 = \frac{DN^2}{A_\sigma} \tag{6}$$

Geometric terrain correction

For the geometric terrain correction, as the second pre-processing step, we use Range-Doppler-Terrain correction. It is a necessary step to remove effects of side looking geometry of SAR images to allow geometric overlays with data from different sensors and geometries [Gens (2006)].

To generate the terrain corrected image we use in SNAP the previously generated radar cross-section σ^0 in the calibration. The digital elevation model (DEM) used is the SRTM (Shuttle Radar Topography Mission) with a spatial horizontal resolution of 3" [Survey (2018)]. Resampling method used is bilinear interpolation with a pixel spacing of $10m \times 10m$. The map projection is WGS84.

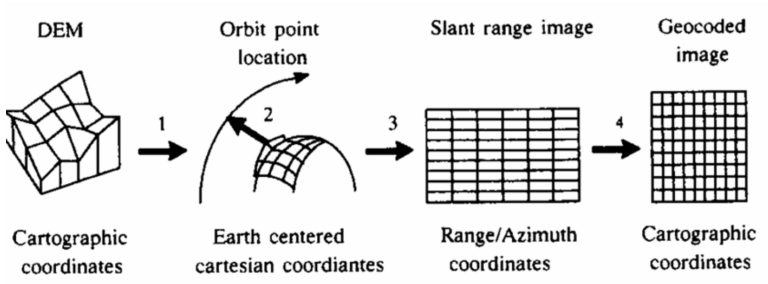


Figure 19: Process of terrain correction (backward geocoding) [T.Bayer (1991)]

Transformation of distribution function

Before we feed the dataset to the algorithm we do a last pre-processing step of transforming the distribution of the SAR images from Chi-squared to Gaussian. Instead of doing some heavy computations for transformation we use a simple approach:

$$f(X) = X^{\frac{8}{27}} \tag{7}$$

where X is the SAR image with chi-square distribution. By applying equation (7) to the SAR images we get the transformation shown in Fig 20 from the blue to the green distribution. The value range (which is the intensity channel) changes approximately from $[0; 8.7]$ to $[0; 1.9]$.

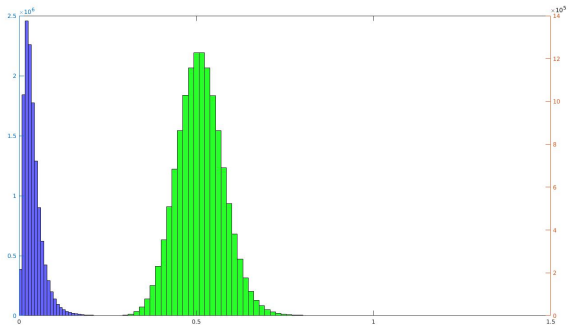


Figure 20: blue: Distribution of the dataset (Chi-squared distribution).; green: Transformed distribution of the dataset to Gaussian approximation.

3.3 Network Architecture

Input data

To feed the network we use a pixelwise approach. We take a 31×31 patch as an input. The label of each patch is the ground truth of the center patch pixel. In other words, to learn from a pixel we take its local area or surrounding context(extend of 15 pixel in each direction). The patch size is set based on the approximate size of deforested pixel clusters. For the following patch we move to the following pixel (sliding window approach). Fig 21 illustrates an example how those patches are generated. We can also see that, on the image boarders we need a padding of the same size as the extend (15 pixel). This padding is done by mirroring on the edges. Since both SAR images (March 2016 and June 2017) have the exact same pixel size and are co-registered, we can take the same patch in each image. As the first input layer for the network we treat both patches as two channels. Instead of training each patch one by one we define a batch size (amount of patches for each training step) of 200. In this case our 4-D input layer has the size $200 \times 31 \times 31 \times 2$. For the training set we feed also the labels for each patch. This is a 2-D layer of the size 200×2 where for each batch either the first element is set to 1 (if label equals deforested) or the second element is set to 1 (if label equals background).

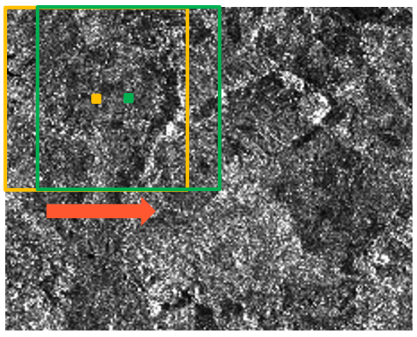


Figure 21: Subpart of the input SAR dataset. Yellow rectangle shows a 31×31 pixel batch where the label of the batch is defined by the ground truth of the middle pixel (yellow dot). The green rectangle shows the sliding window approach defining the next batch by sliding one pixel to the right. Red arrow is the moving direction.

As already mentioned our dataset is highly unbalanced. Only 2.8% of the data is labeled as deforested. For the fact that we have anyway a small training set, the total amount of data for this class is limited. To counteract this unbalance we feed the deforestation labeled batches multiple times to reach at least 20 % of the class deforested for training.

Hidden layers

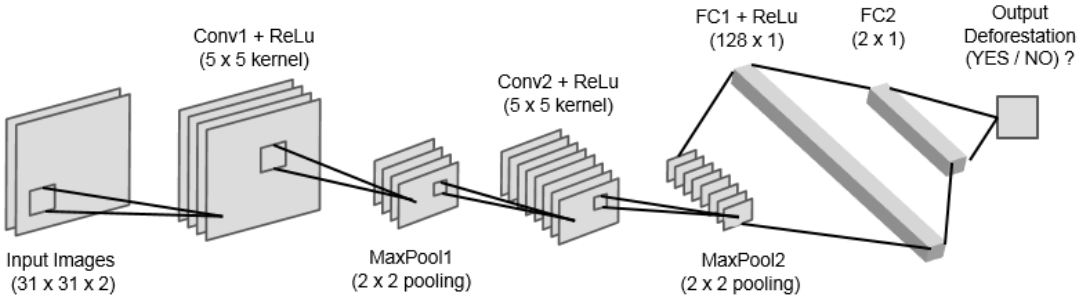


Figure 22: The network is build with five hidden layers. Two convolutional layers (both followed with max-pooling layer) and in the end one fully connected layer. The second fully connected layer is the output.

In Fig 22 we can see the network architecture. After each convolution we apply ReLu to the data before proceeding with max-pooling operator. In both cases a 2×2 max-pooling is used. The kernel size for the convolutional layer is 5×5. 16 filters are used in Conv1. The layer size in MaxPool1 is $200 \times 16 \times 16 \times 16$. For Conv2 are 32 filters used. The layer size in MaxPool2 is therefore $200 \times 36 \times 8 \times 8$. After flatten it to a size of 200×2304 we get the size of $200 \times 128 \times 1$ in FC1, respectively $200 \times 2 \times 1$ in FC2.
With the last layer we compute a probability for each of the two classes (deforestation or background). The predicted class for each batch is the class with the higher probability.

The weights and biases for the network are initialized randomly with a mean value of 0 and a standard deviation of 0.05.

Loss function and Optimization
Once we get the output of the network we need to calculate a loss to adjust the weights in the network. This is done by a function which actually calculates the loss of each training step by comparing the distance between the prediction and the label:

$$loss = \frac{1}{N} \sum_{i=0}^{N} \|y_{gt} - y_{pred}\| \tag{8}$$

where N is the batch size of 200, y_{gt} is the ground truth label and y_{pred} predicted label.
To train the algorithm we use an optimizer called gradient descent optimizer with a learning rate of 0.01.

Cross Validation

We use 75% of the dataset for training and the other 25% for validation. In this case we get a prediction map of the data used for validation. To estimate the performance of the network we use a 4-fold cross validation which means we actually split the entire data in four different parts (see Fig 23), where for each step three of them are used for training and one for validation. In this case we run the process four different times and for each time we get performance measures of the validation as well as a predicted map. To calculate the overall performance we take the mean over all four results. With this method we can generate a predicted map over the entire dataset by putting the prediction maps of each validation dataset together. This can be also used to qualitatively estimate the performance of the network.

Figure 23: Dataset divided in four different subsets. Each subset has the full length of the input image and equal width.

Since the dataset has a total width of 1680 pixel, we decide to divide the parts in the width direction. Each part has a total width of 420 pixel. This means that the training set has the shape of 1260 $\times$ 5250 pixel whereas the validation set has the shape of 420 $\times$ 5250 pixel.

3.4 Post-Processing

As the last step in the process we use morphological operators to improve the result performance. The reason is to smooth the edges of the deforested clusters and fill some small holes in the cluster (since prediction is per pixel and the dataset is limited, it can have some scattered pixels misclassified. To do so, we use two different morphological operators: First, opening operation (which is an erosion followed by a dilatation). A disk shaped structuring element of 5 × 5 pixel is used. On the resulting map closing operation is applied (which is a dilatation followed by an erosion). For this operation a disk shaped structuring element of 29 × 29 pixel is used. In Fig 24 we can see both structuring elements.

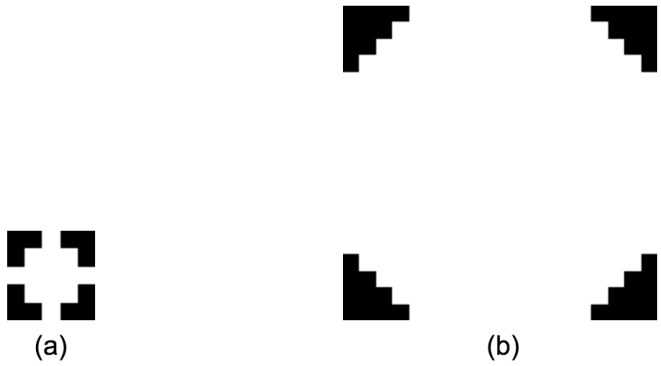
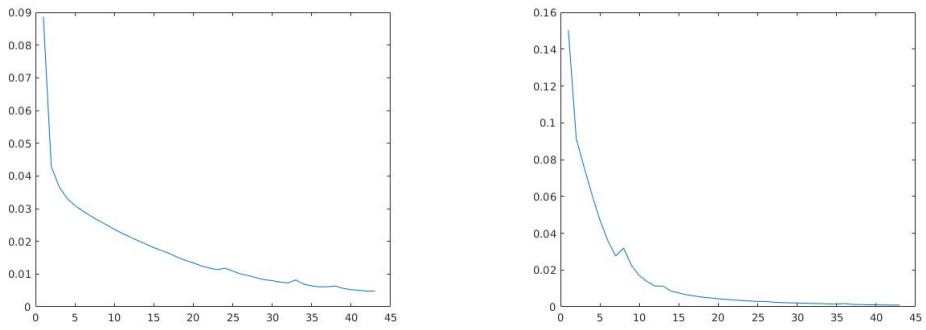


Figure 24: (a) disk shaped structuring element (a) for opening operator (5 × 5 pixel) and (b) for closing operator (29 × 29 pixel)

4 Result

4.1 Cross Validation

As discussed in the previous chapter (how we split the 4-folds cross validation and put the prediction results together) we want to look deeper in the resulting map. We use two different constellation of the input data for this process. The first approach is to train the network with the highly unbalanced dataset. For the second approach we balance the training data with around 20 % deforested training samples. For the training process we stop after 50 epochs which turn out to be a good compromise between training time and performance. As we can see in Fig 25 the training loss for the balanced dataset converges much faster at around 10-15 epochs whereas in the highly unbalanced approach the convergence starts somewhere around epoch 40.



(a) (b)

Figure 25: Mean training loss after cross validation for (a) highly unbalanced training set and (b) balanced training set with 20 % deforested batches.

To show that the more balanced approach is much better we can look at Table 3 where the values can be compared for both approaches.

Training dataset	**Accuracy**	**Recall**	**Precision**	**F1-Score**
Highly unbalanced (2.8 % deforested)	0.9816	0.7583	0.4980	0.5933
Balanced (20 % deforested)	0.9848	0.6949	0.7458	0.7148

Table 3: Comparison of performance between unbalanced and balanced approach

In the table above we can see that despited a small reduction in recall an improvement in accuracy, precision and f1-score can be achieved. The precision increased significantly

from 0.498 to 0.746 which is an increase of around 50 %. For the f1-score we are able to increase it by 20 %. For this reason we chose the balanced dataset to proceed.

Fig 26 shows the ground truth of the entire dataset. Each pixel colored in white is labeled as deforested where everything else in black is labeled as background.

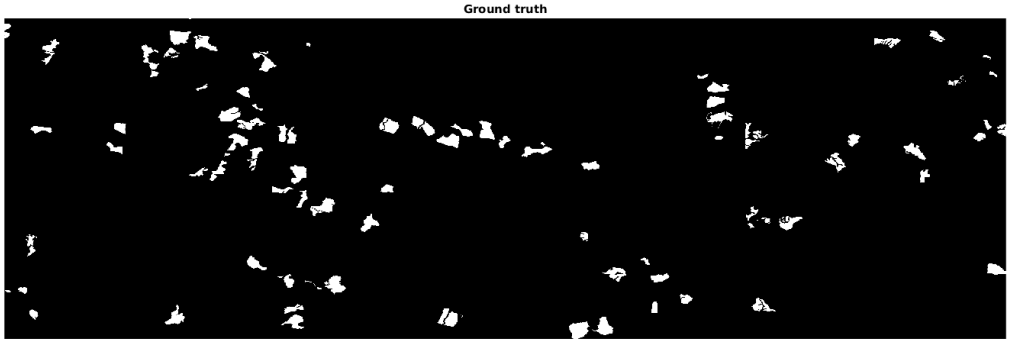


Figure 26: Ground truth labels of the dataset.

To see how well the performance values are qualitatively we can give a look to the prediction map. This is generated by training with balanced dataset (using same pixels with class deforested multiple times in the same epoch with different constellations). For the prediction map we put all 4-folds cross validation together and Fig 27 shows the prediction map after cross validation.

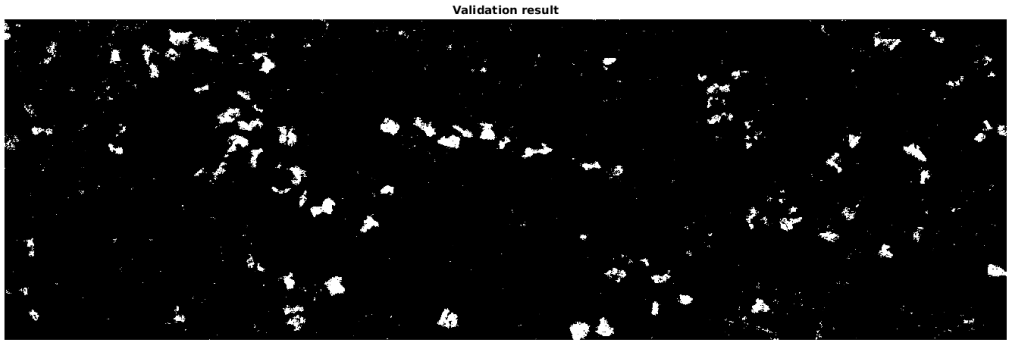


Figure 27: Prediction map after 4-fold cross validation where white pixels are predicted as deforested and black as background.

By comparing the ground truth with the prediction we can easily see that in the first look the prediction of the deforested clusters are performed well. Almost all deforested clusters can be found with its shape. A lot of smaller clusters of false positive are in the prediction included as well as bigger false negative clusters which we will give a look into it in a later step.

4.2 Post-Processing

Instead of stopping at this point we try to improve our result by using post-processing steps. We decide to use morphological operators like opening and closing to get rid of those small false positive clusters which usually have the size of a few pixels. Another point to target with those processing steps is that in some of the true positive clusters (actually deforested areas) are smaller pixels which are predicted as background. This can also be eliminated. The resulting post-processed map is shown in Fig 28.

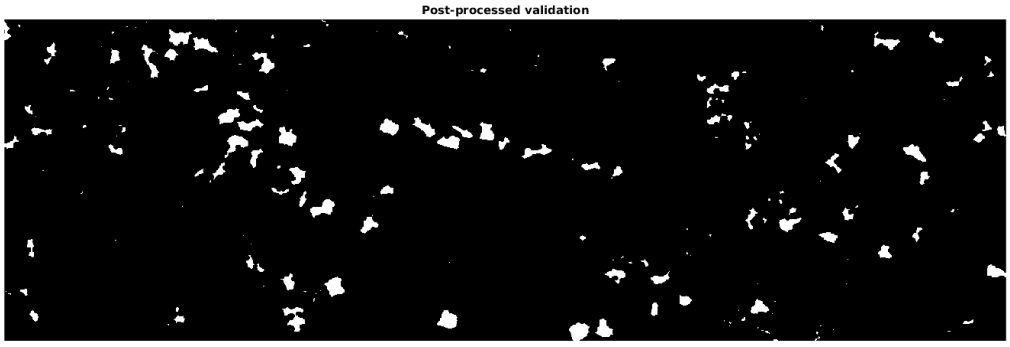


Figure 28: Post processed map after morphological operators

If we compare the post processed map to the predicted map we can see visually an improvement based on the ground truth. A lot of the smaller false positive clusters are removed, bigger deforested clusters do not have holes anymore and the boarders of those clusters are sharpened. The improvement can be also seen quantitatively by comparing the performance of the predicted map and the post-processed map in Table 4.

	Accuracy	**Recall**	**Precision**	**F1-Score**
Prediction result	0.985	0.695	0.746	0.715
Post processed result	0.987	0.793	0.745	0.768

Table 4: Comparison of performance between prediction and post processed result

Even though the improvement is not significantly high visually it leads to a much better result in the end where the post processed map contains clearly those deforested clusters with a better shape regards to the ground truth.
We can look in Fig 29 how the post processing improves the result in a small area of the dataset. While the predicted map contains some holes in the deforested clusters those can be filled with the morphological operators and the edges of those clusters are sharpened.

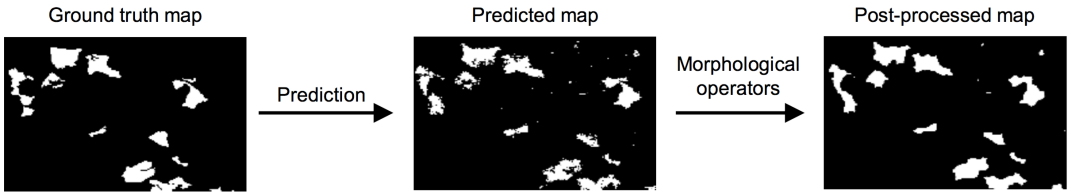


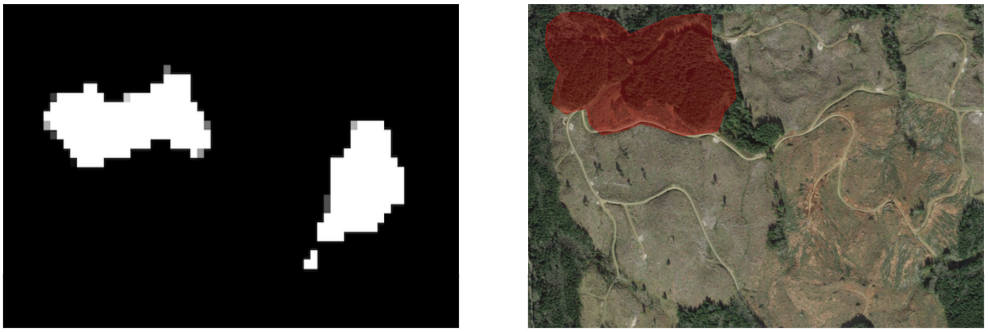
Figure 29: An example from a subpart how post-processing steps improves the result

4.3 Discussion

The overall result is satisfying. But although most of the labeled deforested clusters can be found, the quantitative results can still be improved a lot by using more training dataset. With our limited dataset, the algorithm is restricted to only limited pixels with deforestation. This is also the reason why pixel wise approach is used for training (where training for each pixel with its surrounding takes place), instead of using semantic segmentation with patches. Since we have a highly unbalanced dataset we would need a lot more patches with at least 20% of the pixels labeled as deforested. Because of that limitation it makes less sense.

Another point to mention is that when we check the post processed prediction result we see big clusters where deforestation is predicted with an unique shape which can be identified in Google Earth where deforestation took place but not in the time period of our images. If we look at the Fig 30 we can see that the labeled polygon in Google Earth (red) is predicted in the post-processed result correctly. The result of this subpart contains another big cluster which is predicted as deforested but has in fact the label background. If we now look at the Google Earth image we can see that in this specific area deforestation already took place. By zooming in Google Earth as near as possible to this area we can see that some of the trees are still somewhere at the ground and not collected yet. That same problem is detected in different areas of the training region. With this observation and the fact that we do not know the observation date accuracy of the date given in Google Earths time

slider, we can set up an assumption: For most of the big true negative clusters with similar shape to recently deforested parts (before March 2016) the misclassification is due to the fact that the dates of the sentinel images and Google Earth can have a small shift which leads to errors in labeling for the ground truth.



(a) (b)

Figure 30: (a) Post processed prediction result for a subpart; (b) Google Earth image from the same subpart shown for March 2016.

Another point important to mention is the quality of the ground truth labeling. Since this is done manually and in some cases where we can not clearly define if it is deforested or not (e.g. deforestation just started somewhere around the particular date and is still continuing) the decision of labeling can be affected with subjective view. To get a more objective ground truth label it is possible to let the labeling be checked from another person before using it for training.

Due to the time consuming manual labeling process and limited time for this project, we are not able to generate more training data. Although this can be one of the most important points to improve the overall result and probably do not even need any morphological operators.

The actual shape of the predicted deforested areas does not exactly match with the ground truth which lowers the performance. The reason for that is also the manual labeling process in Google Earth where the exact border of the deforested area is in most case not clear to determine. Thus, wrong predictions on the borders should not be seen as misclassification of the network rather as vagueness of the labeling.

5 Conclusion

To sum it up, we were able to show that with sentinel-1 SAR images deforestation can be detected. We achieved a f1-score of 0.77 after cross validation. Even though the value does not seem to be stunning, by looking at the post processed prediction result we can clearly see on the prediction map for the entire dataset, that almost all deforested areas in the training region could be detected. To increase the performance we need to generate more training data, match the date for the ground truth exactly with the sentinel-1 images and generate the ground truth with higher precision and objectivity. We are convinced that by solving this points the performance can be improved a lot.

By extending the training dataset to multiple locations we can use this application to detect deforestation anywhere on the globe. Since we build our network for detecting deforestation in a time period, our implementation can be used for monitoring purpose in different areas around the world. With a revisiting time for sentinel-1 satellites of just a few days and the fact that SAR imagery are not affected from cloud or any weather conditions, we can use continuous, up to date images to discover deforestation in an early stage and counteract, if needed.
Especially for governmental organizations or companies which are strongly interested in solving illegal deforestation, but where field monitoring is not possible, this can be of great importance.

6 Outlook

There are several ways to improve this project. Following examples are some hints about what can be adapted to this project.

One interesting point would be generating more training data from different geographical regions and generating ground truth data with other sources than only Google Earth, to make it more reliable and see how the network improves.

If the interest is about deforestation in general and not in a specific period, we can modify the network to detect generally deforestation for a specific region. Such would expect just one input image and the network would predict any areas where deforestation took place.

We can also go further and try to generate subclasses depending of the deforestation stage or how much biomass has been destroyed in each image pixel.

Since we normally also have multi spectral data from the sentinel-2 satellites of the same region, we could combine those with the SAR images to train the network. In this case we need to handle the cloud problematic in multi spectral images and some adjustments of the network for the input layer needs to be done.

We could also think a bit out of the topic and use this network for any other environmental change detection besides forest. Since our input expects only images from the same region with different dates and ground truth labeling, this can be adapted without big effort. Some possibilities would be other big global problematics like snowmelt in polar regions, increasing urbanization in settlement areas or decreasing agricultural use.

References

Bibliography

Alberto Moreira, Pau Prats-Iraola et al. (2013). "A Tutorial on Synthetic Aperture Radar". In: *IEEE Geoscience and Remote Sensing Magazine*.

ESA, (European Space Agency) (2018a). *Copernicus mission overview*. http://www.esa.int/ Our_Activities/Observing_the_Earth/Copernicus/Overview4.

– (2018b). *SAR image geocoding*. https://earth.esa.int/web/guest/missions/esa-operational-eo-missions/ers/instruments/sar/applications/radar-courses/content-2/-/asset_publisher/ qIBc6NYRXfnG/content/radar-course-2-sar-image-geocoding.

– (2018c). *Sentinel 1 - Mission*. https://sentinel.esa.int/web/sentinel/missions/sentinel-1.

– (2018d). *Sentinel Application Platform (SNAP)*. http://step.esa.int/main/toolboxes/snap/.

Gens, Rüdiger (2006). *Terrain Correction*. Tech. rep. University of Alaska Fairbanks (UAF).

Harris, Geospatial Solutions (2018). *Calibrate to Sigma Nought*. http://www.harrisgeospatial.com/docs/CalibratingSigmaNought.html.

Karpathy, Andrej (2018). *CS231n Convolutional Neural Networks for Visual Recognition*. http://cs231n.github.io/.

MathWorks (2018). *MathWorks - Deep Learning*. https://ch.mathworks.com/solutions/deep-learning/convolutional-neural-network.html.

Nuno Miranda, P.J. Meadows (2015). *Radiometric Calibration of S-1 Level-1 Products Generated by the S-1 IPF*. Tech. rep. ESA (European Space Agency).

(Oregonlife), Rob Davis (2018). *Deforestation a problem in Oregon despite replanting*. http://www.oregonlive.com/environment/index.ssf/2015/09/oregon_lost_500000_acres_of_fo.html.

Survey, U.S. Geological (2018). *NASA Shuttle Radar Topography Mission Global 3 arc second SRTMGL3S*.
https://lta.cr.usgs.gov/srtmgl3s.html.

T.Bayer R.Winter, et al. (1991). "Terrain influences in SAR backscatter and attempts to their correction". In: *IEEE Geoscience and Remote Sensing Magazine* 29.3, pp. 451–462.

List of Figures

List of Tables