Norwegian University
of Life Sciences

Master's Thesis 2019 30 ECTS
Faculty of Science and Technology

Multispectral Image Analysis of Spring Wheat Using UAV and Machine Learning

Lars Martin Bøe Lied
Environmental Physics and Renewable Energy

Abstract

Multispectral Image Analysis of Wheat Fields Using UAV and Machine Learning

by Lars Martin Bøe Lied

A need to increase efficiency of plant phenotyping has arisen due to global warming, food shortage, population growth etc. One way to do this, is by using image analysis. This can reduce amount of work by analyzing vast areas and get quick results, in comparison to manual labor.

In this thesis, a UAV (Unmanned Aerial Vehicle) attached with an RGB camera, a multispectral camera, a GPS and a light sensor, was flown over three fields of spring wheat. The multispectral camera used the bands; blue, red, green, near-infrared (NIR) and red edge (REG). In addition, the MTCI (MERIS Terrestrial Chlorophyll Index), EVI (Enhanced Vegetation Index) and NDVI (Normalized Difference Vegetation Index) indices were used. The UAV was flown over the three different fields in a grid pattern while collecting images. Afterwards, these images were stitched together into maps of the fields using Pix4D. Maps from two fields from the season before were also included. The image-values within each plot were then extracted using QGIS.

The data extracted was put into machine learning and deep learning algorithms to predict grain yield for each plot. The grain yield had then been measured manually before-hand after harvest. Using the machine learning algorithm Support Vector Regressor for predicting the grain yield, the following R^2 values for each field were achieved: 0.595, 0.582, 0.735, 0.63 and 0.917. The mean absolute errors (MAE) using the SVR for the different fields are between 5.6 % and 11.1 % of the average grain yield for the specific fields. The deep learning models using the two types ReLU and Selu, produced slightly worse results.

The images taken by the RGB camera were used for estimating plant height. This was done by creating a Digital Surface Model (DSM) to model the surface of the fields and a Digital Terrain Model (DTM) to model the terrain the fields were in. To estimate height for the fields, the DTM was subtracted from the DSM. These estimations of height were then compared to manually measured values. The best estimation achieved an R^2 value of 0.33.

Acknowledgements

I would like to thank my project advisors Ingunn Burud and Morten Lillemo for guidance in this thesis. I would also like to thank Sahame Shafiee who has advised me with the image analysis and the work in Pix4d.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to a rapid worldwide population growth and probable challenges in food supply due to climate change, food production must be increased. Therefore, new solutions in agriculture must be considered, and comprehensive research must be done [1]. The Norwegian government also decided to increase food production to match the predicted population growth of 20 % by 2032 [2].

We must however be careful with the approach, so we don't worsen the already fragile environment. As the director for food and agriculture in the UN, José Graziano da Silva stated in 2017;

> *Massive agriculture intensification is contributing to increased deforestation, water scarcity, soil depletion and the level of greenhouse gas emission.* [3]

He also stressed that while the farming of today with high input and being resource intensive has substantially increased food production, this has also come at a high cost for the environment. Another important aspect of this is soil degredation, both chemical and biological, due to fertilizers, pesticides, negative soil nutrient balance and so on. This could lead to about 5-6 million hectares worldwide irreversibly lost each year as a result of soil erosion, salinization and other degradation processes [4]. It is therefore necessary to increase food production without increasing emissions and the use of reagents. Due to cereals being a vital food source for both human and animal alike, this could be a great place to increase the effectiveness of something that is already in mass production. In this thesis, the focus will be on wheat. This is also the most widely grown crop in the world and provides approximately 20% of the food calories and protein for 4,5 billion people [5].

Since there is not an unlimited amount of rural area on earth, we need to increase the effectiveness of the wheat-fields already in use. Therefore, we need to analyse crops to find the most effective way of planting and which types to use. Manual analysis however takes time and is not very effective. Therefore, we can use UAVs (unmanned aerial vehicle) to fly

over the field with both multispectral and RGB cameras to help us with data collection, to make this process more efficient. This is an easy, cheap and non-destructive way of doing analysis. The results of these measurements could potentially help the farmer make decisions to increase yield. For plant breeders, UAV imaging can potentially save time by allowing estimation of plant height without having to do manual measurements.

Another aspect is the breeding of new and improved cultivars, capable of higher yield and more robustness. Automated plant phenotyping can help plant breeders to do more precise selection of such future cultivars. Due to a changing climate, this process should be as quick as possible to ensure the best cultivars for the climate of today and not the potentially different climate from several years ago.

The purpose of this thesis is to improve our understanding of data provided by the images taken by these drones over wheat-fields, for use in plant phenotyping. The correlation betweem data from the images with the amount of grain yield and plant height. To accomplish this, machine-learning and deep learning algorithms will be used. Further, will these algorithms be compared with respect to accuracy.

This thesis is a contribution to the on-going project vPheno (virtual phenomics) at NMBU started 01. May 2017. The main purpose of the vPheno project is to help breeders produce more robust cultivars in a shorter time period than today, which is about 15 years, using image analysis.

There were also pilot studies at NMBU in 2016 [6] and 2017 [7]. This thesis will explore a different way of extracting data from the images, as well as utilizing machine-learning and deep learning algorithms to further investigate the possibility of predicting the grain yield using the vegetation indices of different cultivars. The specific key questions for this thesis to answer will be:

1. Could we be able to accurately predict grain yield using data from the multispectral images?

2. Will computer estimated plant height from UAV images reflect manual plant height measurement?

Chapter 2

Theory

2.1 Plant Phenotyping

Plant phenotyping refers to the assessment of advanced traits in plants, such as growth, development, physiology, yield and several other parameters that form the basis for more complex traits [8].

The goal of any wheat breeding program is development of broadly adapted, durable, disease resistant, high yielding and stable wheat germplasm [9]. The traditional way of doing this in developing countries is by pedigree breeding [10], which typically takes 10-12 years before these new lines are used as parents for the next cycle of plants. In countries in the sub-Saharan Africa this could take as long as about 30 years for maize [10].

Due to traditional phenotyping being time consuming, labour intensive, costly and low-throughput, image analysis has caught interest in this field [11]. This could replace manual observations, thus reducing time consumption for breeding better plants.

2.2 Electromagnetic Radiation

Electromagnetic (EM) radiation is an energy-wave traveling in packets of energy called photons. There are different types of EM radiation, all with different wavelengths. These different types are set in a spectrum of EM radiation ranging from lowest to highest wavelength. Visible light is one small part of this spectrum, ranging from about 400-750 nm. When EM radiation hits an object, the light is transmitted, absorbed or reflected, depending on the characteristics of the object. By analyzing the specific reflectance or absorbance of an object, it is possible to identify and analyze this object [12].

Each component of plant cells and tissues has wavelength-specific absorbance, reflectance and transmittance properties. For Instance, Chlorophyll absorbs photons primarily in the blue and red spectral region of visible light and cellulose absorbs photons in the region

between 2200 and 2500 nm [13]. Imaging in different wavelengths is used for different types of plant phenotyping, such as estimating biomass, disease detection, water content and growth dynamics [13].

2.3 Spectral Indices

Spectral indices are combinations of two or more reflection values for different wavelengths [14]. These can be used to indicate the abundance of the feature of interest. In addition to vegetation indiceses, there are also indices to discover for example burned areas, man-made areas and water. There are some indices that cover a wider area of the wavelength-spectrum and some that cover a narrower area. The narrow bands are better for detecting small changes, and are not as prone to saturation as the broad [15].

2.3.1 Normalized difference vegetation index - NDVI

NDVI is used for analyzing vegetation. This is calculated by measuring the difference in reflection between near – infrared (NIR) and red light (RED). Chlorophyll, which indicates healthy vegetation, reflects more NIR and green light and absorbs more red light, which is why we see leaves as green. NDVI is calculated using this formula:

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

This results in a number between -1 and 1, where a high value indicates healthy vegetation and low values indicates low or no vegetation at all [16].

One weakness with NDVI, is that it quickly saturates in well vegetated areas. This is partly due to the saturation in the red channel, and partly due to that NDVI is ratio based [17].

2.3.2 Enhanced vegetation index - EVI

EVI is calculated similarly to NDVI. However, it corrects for some distortions in the reflected light caused by particles in the air and the ground cover below the vegetation [18]. The EVI value does not become saturated as quickly as the NDVI when viewing areas with a high amount of chlorophyll. EVI is calculated using the formula [19];

$$EVI = 2.5\frac{NIR - RED}{NIR + 6RED - 7.5BLUE + 1} \tag{2.2}$$

2.3.3 The MERIS terrestrial chlorophyll index - MTCI

This index value is another measure for chlorophyll content. This value integrates the red-edge value (REG) into the equation. This is the region where a sharp change in reflectance between wavelengths 690 and 750 nm takes place and characterizes the transition from chlorophyll absorption to leaf scattering [20]. MTCI is calculated by the formula;

$$MTCI = \frac{NIR - REG}{REG - RED} \tag{2.3}$$

MTCI values have been proved to give more accurate results for yield-prediction than using the NDVI [21].

2.4 Machine Learning

Machine learning is a data analytics technique that "learns" from experience on data. In supervised algorithms, this works by having the selected algorithm teach itself the pattern and the most significant parts of the data, using training data where both the datapoints and the result that corresponds with these datapoints are known. The algorithm runs through this training set to best be able to predict the results (Y) by using the data (X). After the model has been tuned on the training data, the model is put to the test by trying to predict a different set of data where the result, Y, has been removed. In this case the model will only look at the X-data. The resulted prediction makes an indication as to whether the model only works on known training data or if the model is generalized enough to be able to predict unknown samples. This could either be used for classifying unknown data or for regression. In machine learning, we have what is called the "No free lunch theorem", which states that there is no machine learning algorithm that is the best for every single case. It always depends on the problem at hand. Therefore, it is always necessary to try different methods to get the best results.

Machine learning is a huge research field, with advances being made all the time resulting in improved methods. Machine learning is used in a plethora of different fields, for instance driverless cars, finance and optimizing lines of production.

2.4.1 Support Vector Regression

Support vector regression, or SVR, is a type of machine learning algorithm used for regression. This is a subclass of the support vector machine (SVM) algorithm, which is used for classifying. SVM works by trying to separate the different classes with as large margin as possible using hyperplanes, with dimensions equal to that of the number of features minus one [22], see Fig 2.1. These hyperplanes will then later work as decision boundaries, where

datapoints on different sides of the plane will be assigned to different classes. By using hyperplanes, we introduce non-linearity to the algorithm, which is an advantage when we are trying to catch small subtleties which may not be a linear combination of the features. Support vectors are then made from the datapoints closest to the hyperplanes. An advantage with this method compared with other algorithms is that we only need to bring these support vectors when we are trying to predict unknown data, saving space. SVR works similarly to this, but the output is a continuous number instead of a class value.

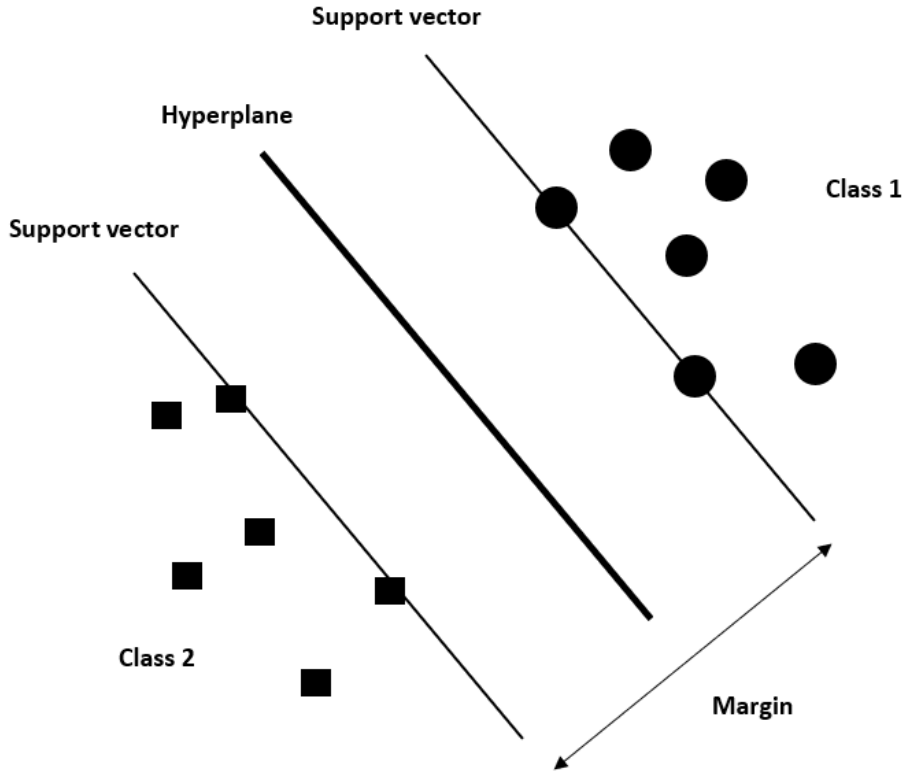


FIGURE 2.1: An illustration of seperating classes using hyperplanes in Support Vector Machines

SVR has several parameter values, two of the most important being; C and type of kernel. C signifies how sensitive the model should be to new training data. This signifies whether to try very hard to fit a new datapoint into the correct label or let there be more slack, and perhaps allow for some more errors in the training phase. Large C-values reduces the margin for error, while setting a lower C-value makes the model less strict [23]. The kernel decides how the samples are to be separated. By using a kernel, it is possible to transform linearly inseperable data to linearly seperable ones [24]. The kernel function is what is applied to the data points. This thesis will focus on the C-parameter and setting the kernel to the default value of "rbf".

2.4.2 Deep Learning

Generally, deep learning is much of the same as normal machine learning, only deeper. It contains more layers and thus more possibilities to catch small details in the data to explain further variation and making more advanced models, also known as neural networks. This also makes the calculations for these models take longer time. It is also no guarantee that the results will get better than a simpler model, so one should always compare different models of different complexities. This also refers to the "No free lunch" theorem. Contrary to what some might believe, a neural network is not a new technology that has been developed in recent years. It was first proposed in 1944 by Warren McCullough and Walter Pitts, from University of Chicago, as a loose model for the human brain [25]. However, it didn't get much traction until later when computational power increased [26].

Another disadvantage with neural networks, is that it handles somewhat like a black box. This means that you don't precisely know why it produced the output it did. This is illustrated in Fig 2.2 below, where the network is trying to guess which animal the picture is of. This can be a problem if you need to explain the details of why one decision is better than the other.

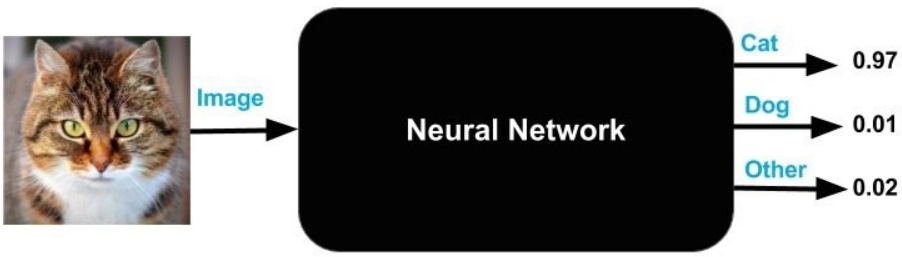


FIGURE 2.2: An illustration of Deep Learning acting as a black box when doing classifications [27].

Deep learning algorithms generally require more data to perform than machine learning algorithms. However, it is also better at getting more information from more data, illustrated in Fig 2.3 below.

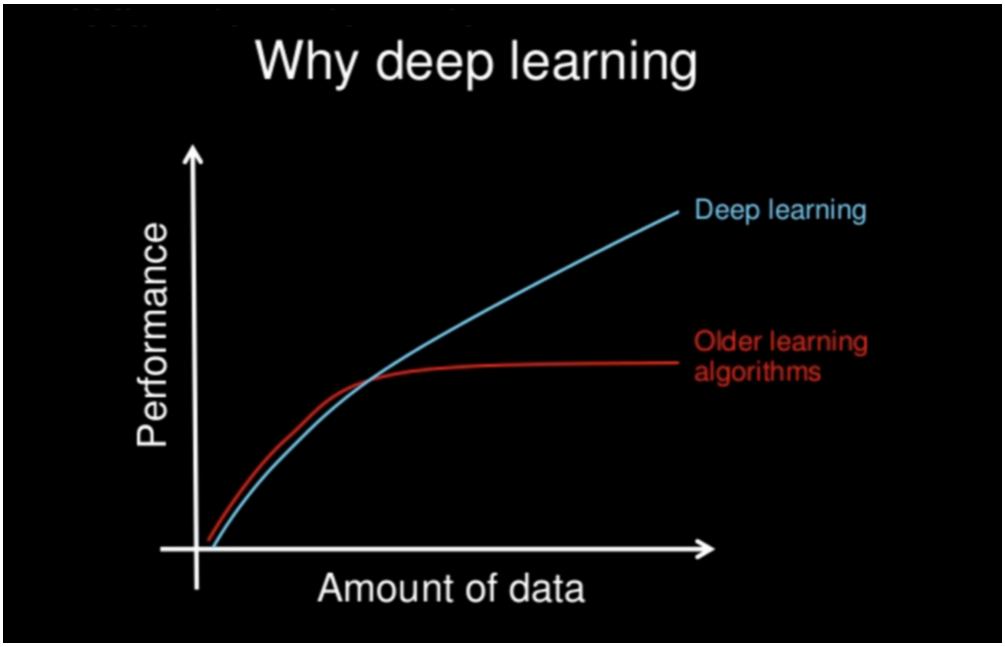


FIGURE 2.3: An illustration showing how Deep Learning handles more data in comparison to machine learning[26].

One important parameter to set in a neural network is the type of activation function. This depends primarily on the kind of problem at hand. This could depend on whether you have a two-class problem or a multi-class problem. It is also a matter of trying different ones to find the best, as stated in the No free lunch theorem. Another parameter for the deep learning algorithm is the number of epochs. This refers to the number of times to run through the data and learn the patterns for the prediction. This could result in overfitting the model and thus giving worse results on unseen test data. The ideal number of epochs is generally dependent on the problem and has to be tested.

Rectified Linear Units - ReLU

One activation function that is much used is ReLU, the Rectified Linear Units. While improving on earlier functions, it also has a potential of "dying" during training [28]. The ReLU activation function is show in Fig 2.4 below.

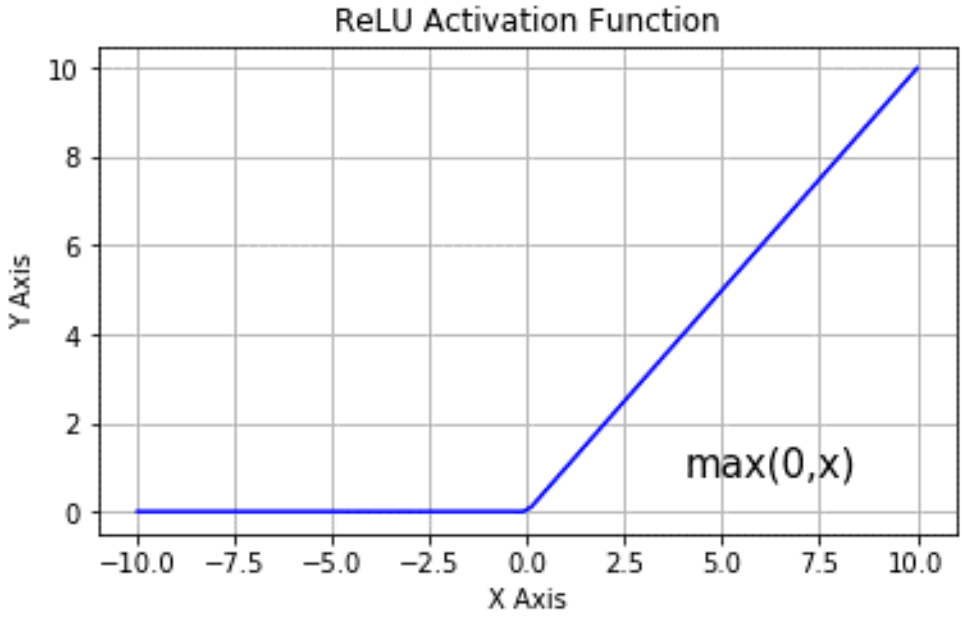


FIGURE 2.4: Graph showing the output of the ReLU activation function

The output is calculated as:

$$\phi(x) = max(0, x) \tag{2.4}$$

This makes it unlinear, and therefore suitable for neural networks. The fact that the output is zero for all negative values, could however cause problems in finding the best model [28].

Scaled Exponential Linear Units - SELU

The point that ReLU can "die" during training due to the output being zero is something SELU aims to combat, with this slightly different activation function;

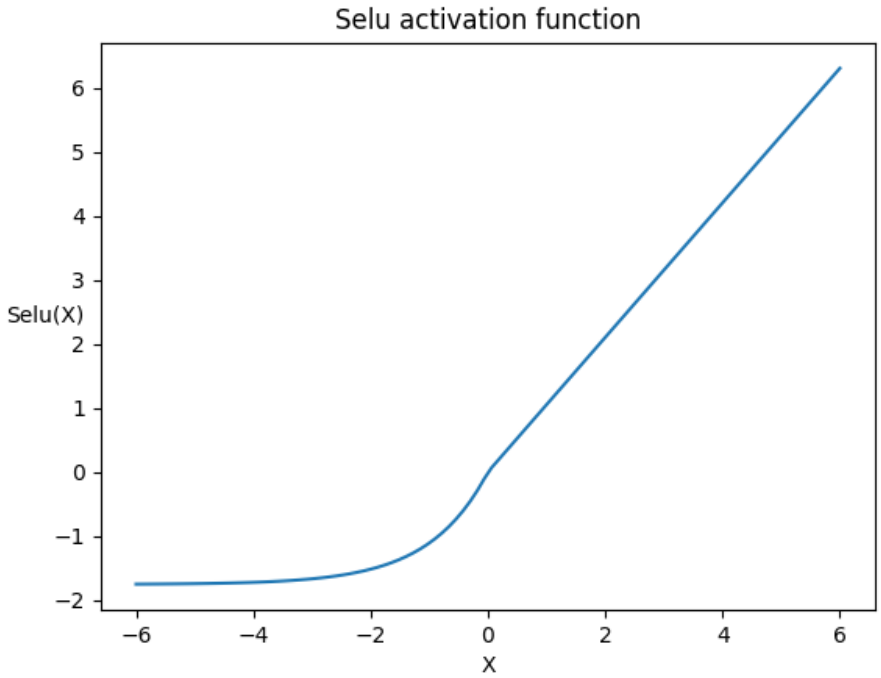


FIGURE 2.5: Graph showing the output of the SELU activation function

The output of this activation function is calculated by;

$$selu(x) = \lambda \begin{cases} x, & \text{if } x > 0 \\ \alpha e^x - \alpha, & \text{otherwise} \end{cases} \tag{2.5}$$

In eq. (2.5), λ and α are two fixed parameters, determined automatically based on the input features. This activation function makes the model move more toward the optimum and is not caught by the problem of the output being zero for ReLU [29].

2.4.3 Sequential Feature Selector

In machine learning or analysis in general, you always want to be dependent on as few features as possible. This increases the robustness of the model as well as the interpretability. One should always combine this with getting the accuracy of your prediction as high as the problem dictates. One way of getting the most important features is by using the sequential feature selector [30]. This algorithm needs a separate algorithm for modelling the data, such as the SVR, to work.

This algorithm can work in one of two ways, either forward (SFS) or backwards (SBS). In SFS the algorithm starts with using just one of the features and tries to model the data using the given model, for example SVR. It then picks the feature that provides the highest accuracy, or a set performance metric. This process repeats itself up to a set number of features that the user decides. SBS works backwards, meaning that it begins with all features and removes the one that gives the least reduction in performance, or potentially the highest increase in performance. This repeats itself down to a set number of features. It is also possible to include or remove features that have been previously picked, to make sure that all combinations have been covered. These variations are called Sequential Forward Floating Selection (SFFS) for the forward moving or Sequential Backward Floating Selection (SBFS) for the backward moving.

2.4.4 Scaling the Data

Numerical values put into a model can have very different range depending on the feature. If the values from one feature contain huge values while the values of another feature never exceeds 1, there is a chance that the larger values will get more attention from the models in use. Most models are also optimized for scaled data [31]. This is accomplished by centering the data around 0, meaning having the average value of that column of features to 0, and a standard deviation of 1. The formula used is the following:

$$z = \frac{x - \mu}{\sigma} \tag{2.6}$$

In eq. (2.6) z is the new standardized value, x is the original value, μ is the average of the feature column where x is present and σ is the standard deviation of the feature column.

Chapter 3

Methods

In this chapter the test site and methodology used in this thesis will be described. Firstly, the image capturing techniques and software will be explained. Afterwards,the methodology of data extraction and the process of converting images in to maps, and further into tables usable for analysis be explained.

3.1 Test Site

The test site used in this study is located at Vollebekk Research Farm, close to the Norwegian University of Life Sciences (NMBU) in Ås, Akershus(59°39'N 10°45'E). In this study, three spring wheat field trials were used. Two larger field trials called 18T1-T18 and 18BMLG1 and one smaller field called 18BMLROBOT. In this thesis 18T1-T18 will be called A-18, 18MLG1 will be called B-18 and 18BMLROBOT will be called C-18.

A-18 is divided into several small trials, and most of them are split into two replicates containing the same cultivars but with a different randomization for placement. A-18 contains 396 different cultivars with 600 plots in total and was planted 09. May 2018. The B-18 field is set up as seen in Fig 3.1 below, with 301 different cultivars and 528 plots in total. B-18 is set up in an augmented design, meaning that not all cultivars are repeated in the two different replicates. 225 of the cultivars are repeated in both the replicates while the rest are only present in one of the replicates. The randomization is done in an alpha-lattice design. B-18 was planted 10. May 2018. C-18 contains 24 historical wheat cultivars which were grown at two different fertilization levels of 75 kg N/ha and 150 kg N/ha. This field is set up in a split plot design. There are 4 replicates, resulting in 96 plots in total. The 24 different cultivars in each replicate is randomized in an alpha-lattice design. C-18 was planted 10. May 2018.

Data from two fields from 2017 were also included; 17BMLROBOT1 and 17CMLG1. 17CMLG1 will from now on be called A-17 and 17BMLROBOT1 will from now on be called C-17. A-17 contains 301 different cultivars planted 4. May 2017 in two replicates, with 560 plots in total, in the same way as B-18. Site C-17 is a smaller field, containing 24 historical spring wheat cultivars planted in the same arrangement as C-18 resulting in 96 plots in total. C-17 was planted 24. May 2017. See table 3.1 for names of fields and abbreviations used.

The field maps for all fields can be seen in Appendix B.

TABLE 3.1: Table of fields with materials planted, abbreviations and seasons

Name of field	Material	Abbreviation	Season
18T1-T18	Graminor breeding lines	A-18	2018
18BMLG1	MASBASIS spring wheat collection	B-18	2018
18BMLROBOT	Historical cultivars	C-18	2018
17CMLG1	MASBASIS spring wheat collection	A-17	2017
17BMLROBOT1	Historical cultivars	C-17	2017

Each plot in every field is given an identification number used in the analysis, as seen in Fig 3.1:.

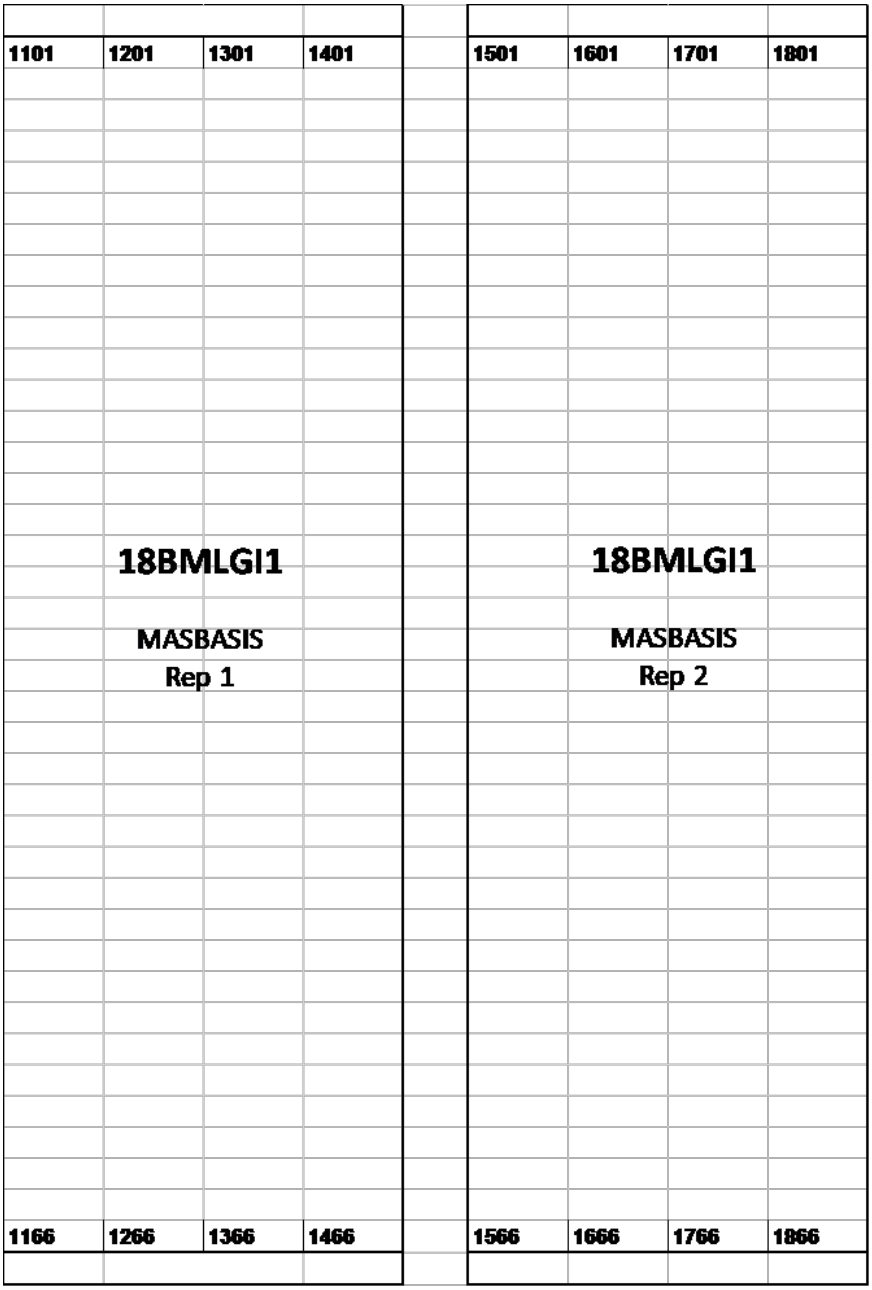


FIGURE 3.1: Figure showing the set up of the field map of B-18 with identification numbers for each plot

In each of the field trials, border rows were planted to decrease border-effects. Herbicides and fungicides were applied following standard agronomic practice. Manual measurements of days to maturity (DM) and days to heading (DH) were done. Plant height (PH) was measured once in the end of the season, as the plants reached their maximum height. This was done by measuring the distance from the soil surface to the base of the heads at two random places within each field trial plot. Grain yield (GY) for each plot was measured in $gram/m^2$.

3.2 Sampling of Images

The images used in this thesis were taken by the unmanned aerial vehicle (UAV) Phantom 4. It used a RedEdge-M multispectral camera from Micasense, a Downwelling Light Sensor (DLS) from Micasense, the included GPS module and the RGB camera that came with the Phantom 4. The image size produced by the RGB camera was 3000x4000 pixels. The two cameras were faced downward, and the irradiance sensor upward. The irradiance sensor was used to correct for varying sun-conditions during flights. The multispectral camera collected images in 5 different bands; red, green, blue, NIR and REG, with set wavelengths and bandwidth as seen in table 3.2. The specifications for the imager itself can be found in table 3.3. The camera also calculated images of NDVI values automatically.

TABLE 3.2: Table of the different bands in the multispectral camera used, with specifications [32]

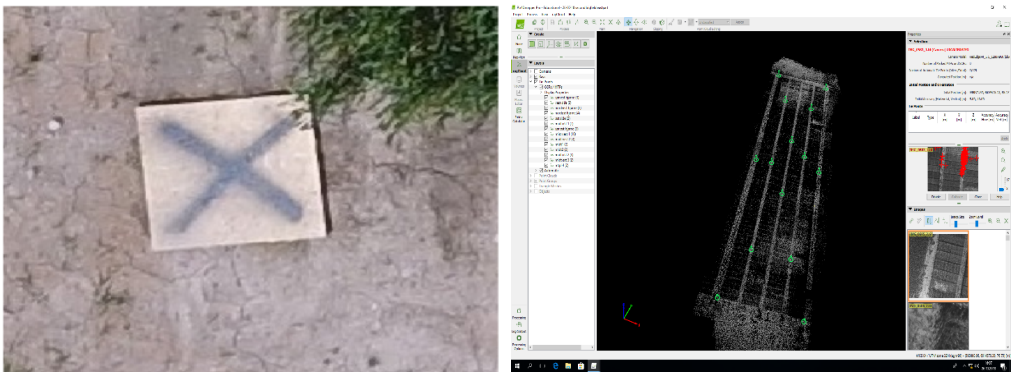
Band number	Band name	Center wavelength (nm)	Bandwidth (nm)
1	Blue	475	20
2	Green	560	20
3	Red	668	10
4	NIR	840	40
5	REG	717	10

TABLE 3.3: Specifications of imager used in multispectral camera [32]

Item	Value
Lens focal length	5.4 mm
Lens field of view	46 deg. HFOV
Imager size	4.8 mm x 3.6 mm
Imager resolution	1280x960 pixels

The drone was first set to fly automatically according to a set flight plan, but this proved to be too unreliable, so the drone ended up being flown manually. DJI GO 4 was the app used for the automatic flying. The two cameras were set to automatically take pictures every two seconds. The drone was flown with a speed of approximately 1-2 m/s, and a height of 11-12 meters above ground level. The pictures taken while the drone was ascending, and descending were removed from the files used in the analysis.

To help the software with producing maps from the images, ground control points (GCP's) were laid out in the fields, see fig 3.2a. These were placed in the corners and some spread out in the middle of the fields, see fig 3.2b. These GCP's did not have manually measured GPS coordinates, and were used more as markers to align the maps.



(A) One of the GCP's used (B) Placement of GCP's in Pix4D

FIGURE 3.2: One of the GCP's used, as well as an example of placement in the field

A calibration plate, or a Calibrated Reflectance Panel (CRP), was used as well to help calibrate the images for the daily light-levels. Used in addition to the DLS, the CRP can help confirm the information collected by the sensor and serve to increase confidence in the overall result [33]. The CRP was added by including a picture from each of the bands from the multispectral camera of the CRP just before flight, see fig 3.3. The CRP has a QR-code for imaging software readability, and a gray area of known reflectance. Both the use of the DLS and the CRP enables comparison of the images from different dates even though the light-levels could be different. Single clouds and shadows might still cause problems with the calibration.

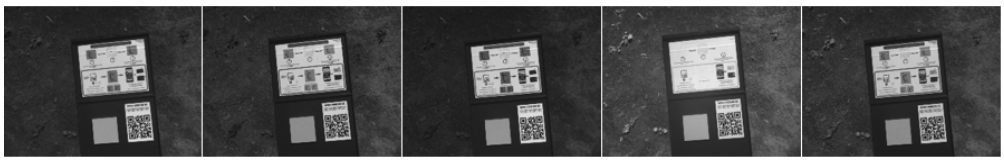
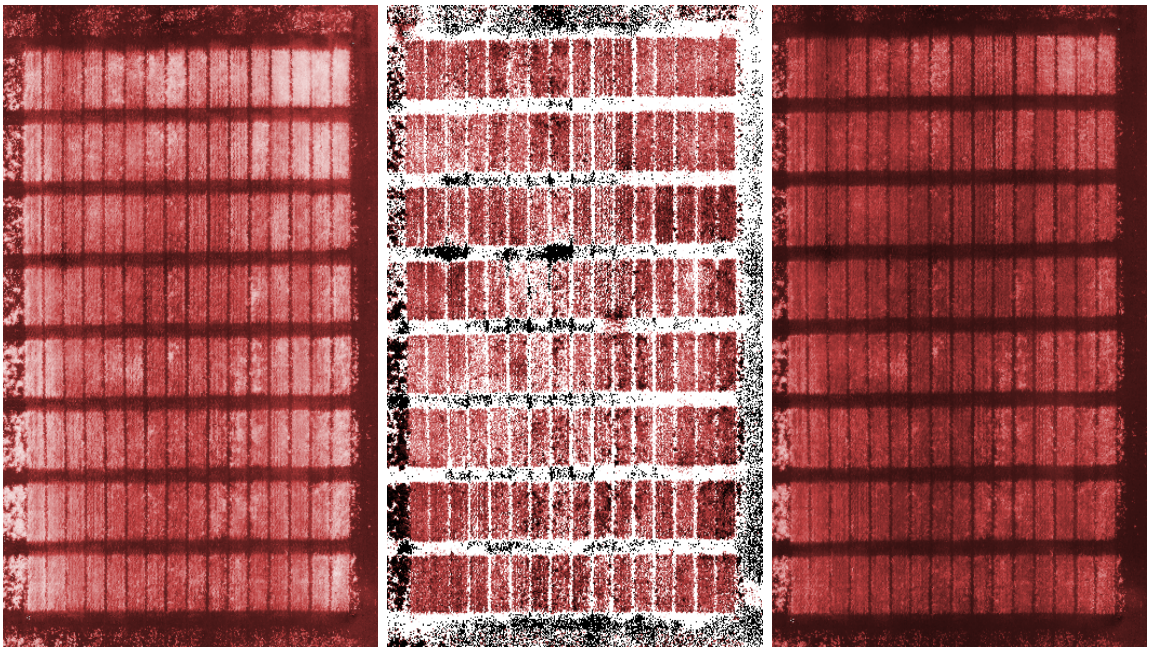


FIGURE 3.3: Calibrated Reflectance Panels shown in each band in the multispectral camera (2018 season)

3.3 Pix4D

The main image software used in this thesis is Pix4D, which is a commercial image processing software specialized in surveying, construction, mining, public safety and agriculture. The main purpose of Pix4D in this thesis is to stitch all images taken during flight into reflectance maps. The use of Pix4D was done according to Grindbakken (2018) [7].

Some examples of the maps created using MTCI, NDVI and EVI can be seen in Fig 3.4.



(A) Map using the NDVI index (B) Map using the MTCI index (C) Map using the EVI index

FIGURE 3.4: Maps created in Pix4D using the indices MTCI, EVI and NDVI. The maps have been colorized to better visualize differences. These maps are from the C-18 field on 11.07.18

The RGB-images formed the basis for the maps used in height estimation of the plants. For this purpose, Digital Surface Models (DSM) and Digital Terrain Models (DTM) using Pix4D were made. Ideally the DTM should be a 3D model for the ground, not including the plants, and the DSM should be a 3D model for the general top surface including the plants, with height values for each point. An estimation for the height of the plants could then be made by subtracting the DTM from the DSM.

3.4 Data Extraction

To get the necessary data from the generated reflectance maps generated by Pix4D, the software QGIS was used. QGIS is an Open Source Geographic Information System (GIS). QGIS can be used for several applications including vector analysis, sampling, geoprocessing and geometry [34]. In this thesis, QGIS was used for creating a custom grid matching the setup of the fields themselves and furthermore, extracting the values from the maps within these cells of the grid. The values were then extracted into separate Excel files and sheets corresponding to the specific field and date of flight. For the exact workflow see chapter 3.4.1.

3.4.1 Workflow in QGIS

This part will show the workflow used for QGIS in this thesis. Be aware that the version of QGIS in this thesis is 3.4 Madeira, and that changes in future versions might occur. The first part is to load the maps into QGIS. This is done by dragging each of the maps for each band on to the screen, this should be a TIFF file, see fig 3.5.

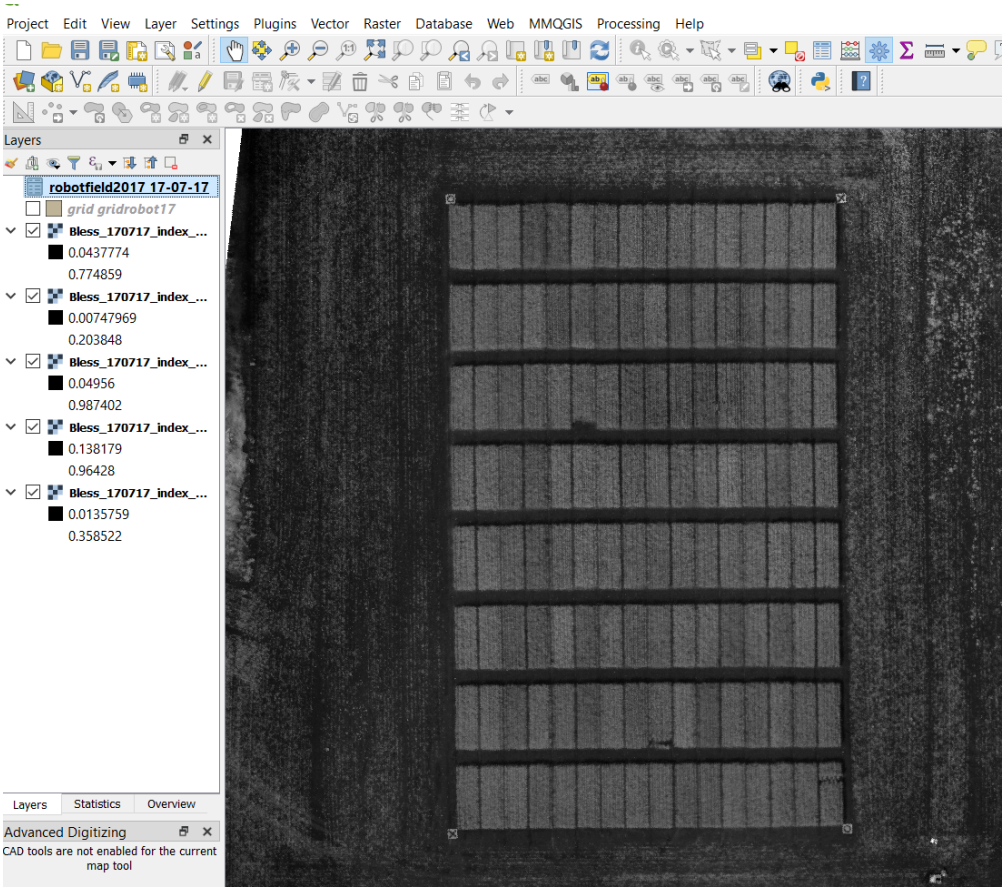
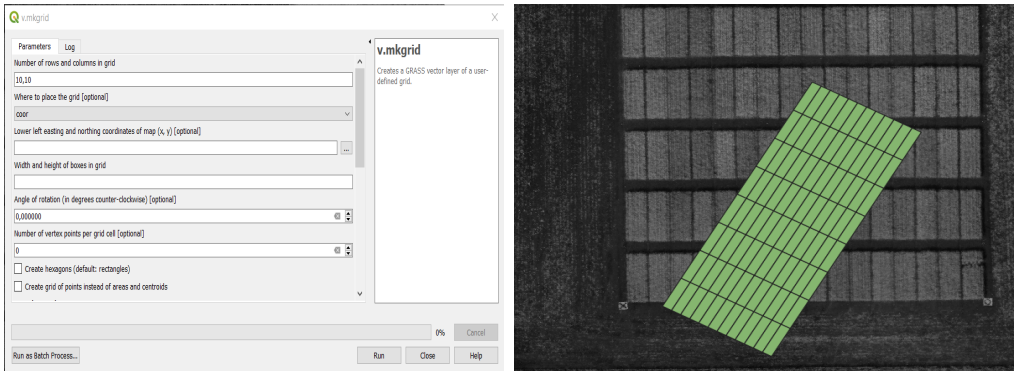


FIGURE 3.5: This is what it should approximately look like after inserting the maps into QGIS

Afterwards, make a grid using the v.mkgrid function, with algorithm ID of "grass7:v.mkgrid", under Processing → Toolbox . This creates a grid with user defined parameters, see Fig 3.6.



(A) Interface of the grid function (B) The output of the grid function

FIGURE 3.6: The use of the v.mkgrid function

A separate layer will then be made containing the grid. Right-click this layer and press "Toggle Editing". The grid can then be moved, rotated and changed further using the options on the top of the screen. However, before changes can be made, remember to press the "Select features" option to select which grid-cells you want to change, by having the wanted features within the area you make, see Fig 3.7.

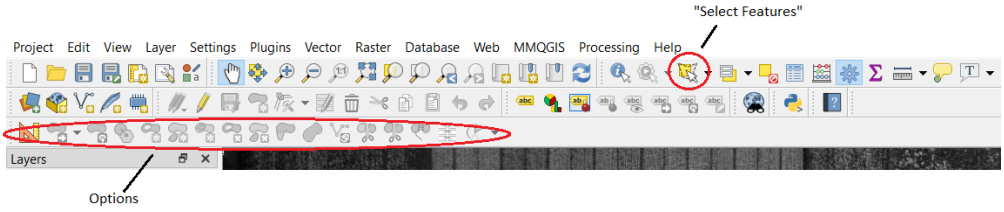


FIGURE 3.7: Most important tools to use on the layers

After moving each cell to the wanted position, it should look somewhat similar to Fig 3.8. The reason for four of the columns being left out, is that these four were border-columns, and therefore not being a part of the experiment. This also shows the advantages of using such a custom grid, enabling the user to focus on the important cells, as well as avoiding parts of the map where the image quality is bad.

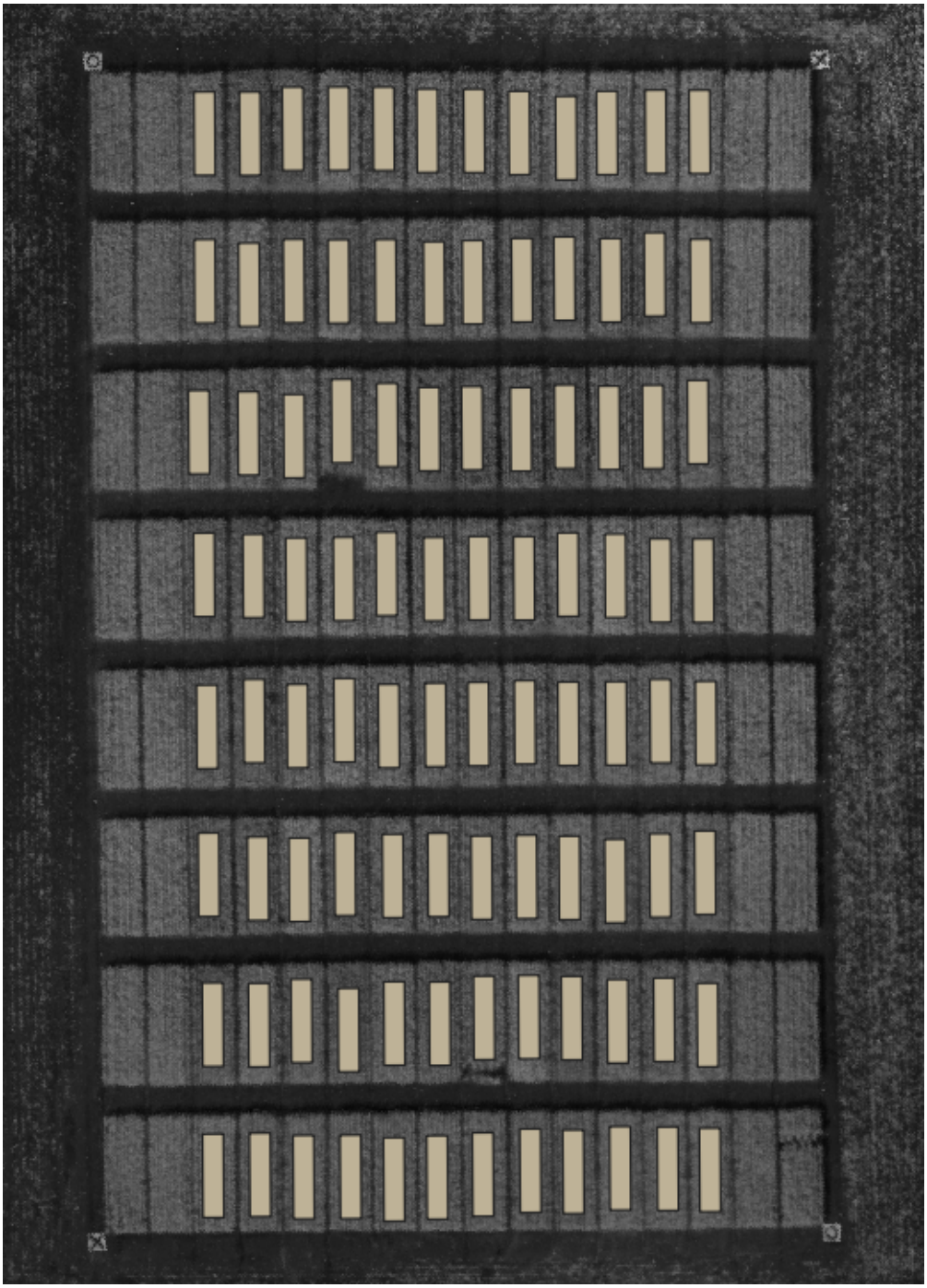


FIGURE 3.8: Here, the cells(features) are split correctly into the different plots in the field

The next step is to use the function "Zonal statistics", with algorithm ID of "qgis:zonalstatistics" in the same toolbox as the previous function. Input each of the layers/bands into the raster layer-parameter and leave Raster band as it is. Make sure that the parameter "Vector layer containing zones" contains the corrects grid that you want to use. Set the prefix parameter to the preferred name, for example the name of the band. Then, set the statistics that you want to calculate for each of the cells, and press "Run". If you want to look at the calculated

data, right-click the grid layer and press “Open attribute table”. To delete data from the table, double-click the grid layers and go to “Source Fields”, press “Toggle Editing” and select the wanted features and press “Delete Fields”.

It is possible to save grids to use them for other sets of maps. This is done by right clicking the grid, press “Export” – “Save features as”, then select “Geopackage” as the format, then set name and save path. Please note that when using a pre-existing grid, it is necessary to delete the already existing features, as described above.

The last step is to save the table as an Excel file. This is done by right-clicking the grid layer then press “Export” → “Save features as”, then select “MS Office Open XML spreadsheet [XLSX]” as format. Set file name, layer name and your respective CRS (Coordinate Reference System). The layer name will here correspond with the sheet-name in the Excel file. This is useful for having the same field in the same file but separating the data from each of the dates in each sheet.

3.5 Data Analysis

In this thesis, machine learning and deep learning will be used to predict the measured grain yield using the data from the multispectral images as well as the maturity date (MAT). The maturity date for C-17 was not measured. This prediction will then be used to see to which extent the grain yield can be predicted using this kind of image analysis.

For the machine learning algorithms used in this thesis, the Scikit-learn package was used [35]. This is an open source machine learning package that includes several tools for data mining and data analysis for the programming language Python. One of these packages is a support vector regression algorithm with the name; sklearn.svm.svr, which was used for predicting the grain yield for each plot in the field based on the data from the multispectral images. The Python-version used in this thesis is 3.6.

For the deep learning methods, Keras was used [36]. Keras is a deep learning library for Python, running on top of either TensorFlow or Theano. In this thesis, TensorFlow was used. Keras works as a high level API (Application Programming Interface), making the functionality in TensorFlow easier to implement and more user-friendly. From this library, all the deep learning algorithms were collected.

The sequential feature selector algorithm was collected from Mlxtend [37] , which is a Python library consisting of several tools.

For visualizing the development of the data analysis, the Python package matplotlib.pyplot was also heavily utilized. All features were standardized and scaled using the StandardScaler package from Scikit-learn.

For detecting outliers in the dataset, the SAS code in Appendix A was used. One of the outputs from the "PROC-MIXED" function in the SAS code is the residuals. This would then be used to remove outliers by checking for high residuals.

Chapter 4

Results

In this chapter the results of the mentioned methods will be presented. Firstly, the data extracted from the maps using QGIS will be presented followed by the results of the data from 2018 and 2017, respectively.

4.1 Data Extracted

The spectral bands and indices extracted using the QGIS software are shown in table 4.1. The camera used in 2017 did not have the blue band, therefore neither the blue nor EVI was collected as EVI requires the blue value to be calculated.

TABLE 4.1: The spectral indices used in 2017 and 2018 in the multispectral camera

2017	2018
Red	Red
Green	Green
NIR	NIR
REG	REG
NDVI	NDVI
MTCI	MTCI
	Blue
	EVI

The statistics calculated for each cell in QGIS were the median, mean and the standard deviation. The median value was chosen as the input for the data analysis since the median should be more prone to outlying values in the images than the mean. The usable data for each field were captured on the dates shown in the following table:

TABLE 4.2: Table showing the dates for each map used for each field, as well as the sowing dates and ranges in heading and maturity dates within each of the field trials.

A-18	B-18	C-18	A-17	C-17
26.06.18	13.07.18	02.07.18	14.07.17	14.06.17
02.07.18	26.07.18	05.07.18	17.07.17	19.06.17
19.07.18		11.07.18	20.07.17	29.06.17
		19.07.18	01.08.17	03.07.17
		24.07.18		14.07.17
				17.07.17
				14.08.17
Sowing: 09.05.18	Sowing: 10.05.18	Sowing: 10.05.18	Sowing: 04.05.2017	Sowing: 24.05.2018
Heading: NA	Heading: 17.06-04.07.18	Heading: 19.06-26.06.18	Heading: 27.06-09.07.17	Heading: 11.07-18.07.17
Maturity: 22.07-02.08.18	Maturity: 24.07-05.08.18	Maturity: 20.07-29.07.18	Maturity: 07.08.-20.08.17	Maturity: NA

The different dates for each datapoint were treated as different features. To get some information on the development of the MTCI values, the change between the MTCI values for each separate date were also added as features. This resulted in quite wide initial tables with many features, with number of features per field being approximately the number of separate bands times the number of dates for each field. The number of the initial features and the number of samples used for each field can be found in table 4.3. The number of samples refers to the number of plots in each field.

TABLE 4.3: Table showing the number of features and samples extracted from each field.

Field	Number of samples	Number of features
A-18	599	27
B-18	526	18
C-18	96	45
A-17	560	28
C-17	96	48

4.2 Data Analysis

The results from the SAS code did not find any datapoints with significant residuals. Therefore, none of the samples were removed due to the residuals. One sample was previously removed due to having an obvious wrong value of grain yield of -9.

The momentary goal in this analysis is to lower the number of features as well as get as high a prediction accuracy as possible. First, the C-parameter in the SVR algorithm must be set to a value best suited for these data. This is done by using the SVR with different C's and comparing the results with both the training data and the test data. The graph showing the result of this for field A-18 can be seen on Fig 4.1. The scoring for this is done by comparing the R^2.

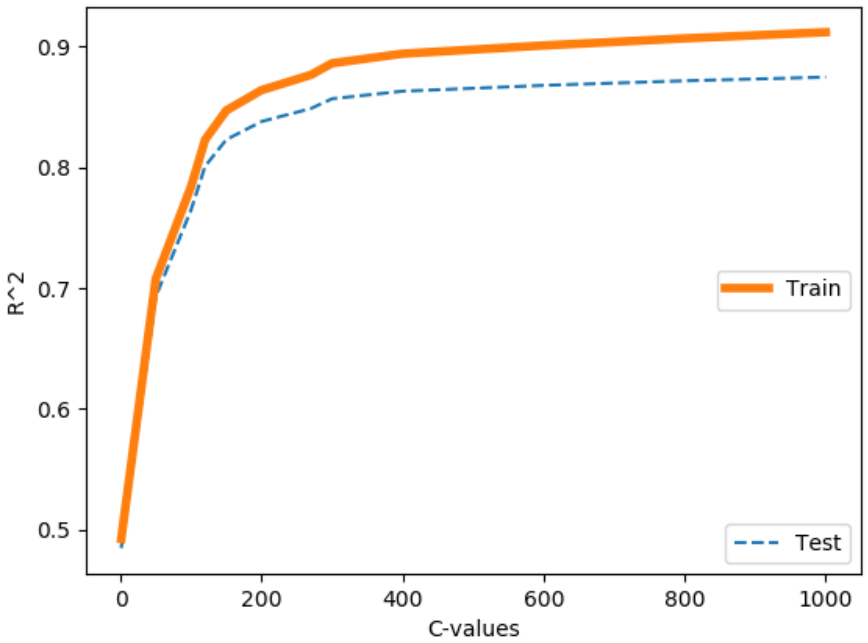
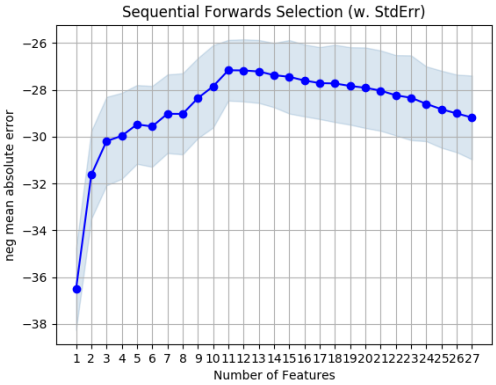


FIGURE 4.1: Figure showing development of the R^2 values for predicting grain yield with different C-values for field A-18

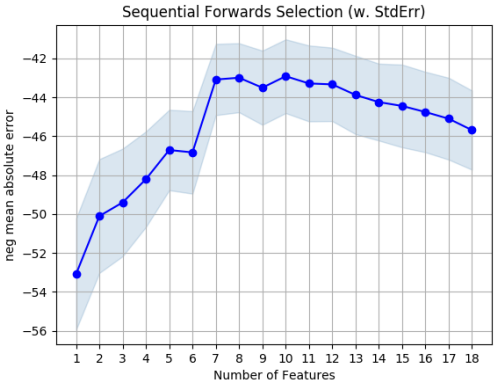
It is clear from Fig 4.1 that the R^2 value doesn't change much after about a C-value of about 350. With higher C's, the difference between test and train accuracy also increase, which is not preferable. Therefore, the C-value used in this field was set to 350. The same analysis was made for the other fields as well with similar results.

The data for each field were put into the SFFS algorithm to find the most important features, using the SVR algorithm as the model with the set C-value. The graphs showing the development of the prediction using different features can be seen in the graphs in Fig 4.2. The score metric used onward is the mean absolute error (MAE). This number is hard

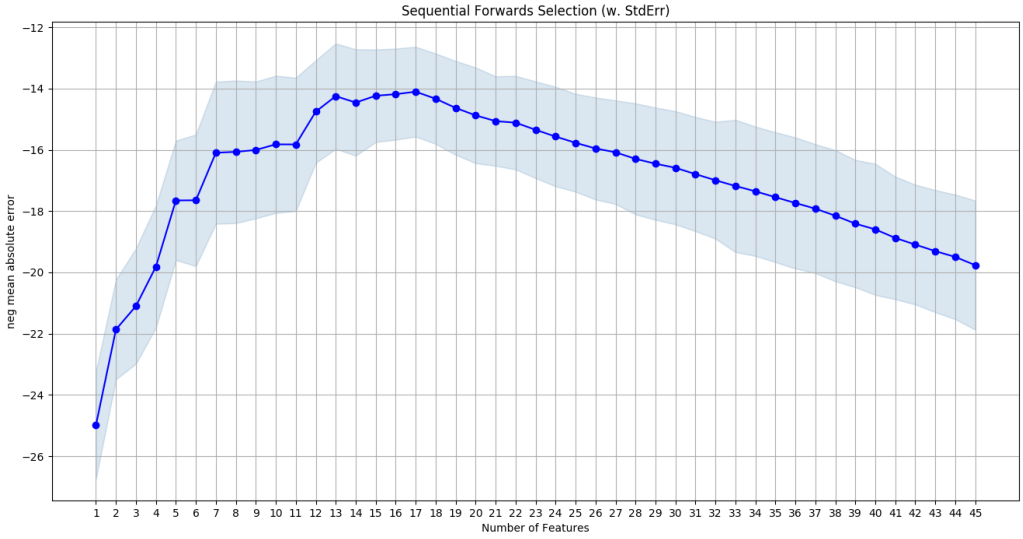
to compare directly between fields, since the average grain yield differs by a large margin between each field. A comparison done by setting the MAE for each field as a percentage of the average grain yield for the given field is done later this chapter in table 4.8.



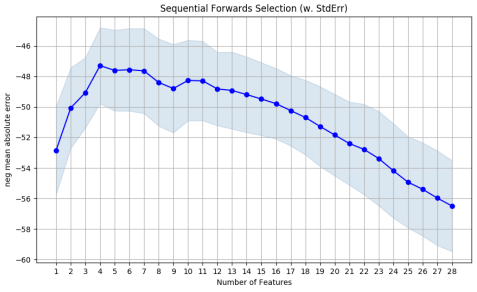
(A) Figure showing the development of the MAE using different features for the A-18 field



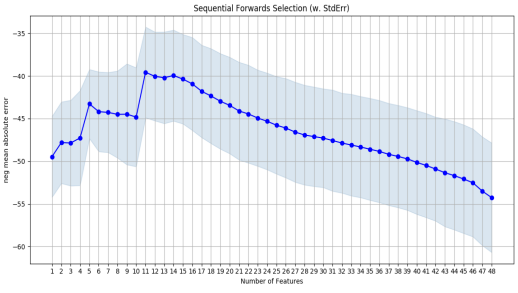
(B) Figure showing the development of the MAE using different features for the B-18 field



(C) Figure showing the development of the MAE using different features for the C-18 field



(D) Figure showing the development of the MAE using different features for the A-17 field



(E) Figure showing the development of the MAE using different features for the C-17 field

FIGURE 4.2: Development of the MAE using different features for each field

It is clear from these graphs in Fig 4.2 that some features are redundant and makes the prediction worse. This is likely due to an overfit of the model, meaning that all these features create more noise in the prediction rather than useful information. The features used for making the best prediction along with the MAE and R^2 achieved while using these features can be seen in table 4.4. The average measured grain yield for each field is also presented to be able to evaluate the MAE. In table 4.4, the features are not ordered by importance. The R^2 and MAE values are calculated using 100 different random splits and taking the average over these. These predictions were of "unseen" data that were not part of the training for the set model.

TABLE 4.4: Set of features giving the best prediction for each field. R^2 and MAE using these features along with the average grain yield for each field is also shown.

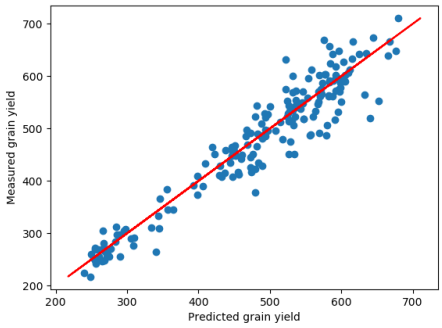
Index	A-18	B-18	C-18	A-17	C-17
0	26-06-18_bluemedian	13-07-18_bluemedian	02-07-18_greenmedian	17-07-17_greenmedian	14-06-17_ndvimedian
1	26-06-18_redmedian	13-07-18_greenmedian	02-07-18_ndvimedian	20-07-17_redmedian	29-06-17_greenmedian
2	26-06-18_mtci	13-07-18_ndvimedian	02-07-18_evi	20-07-17_mtci	29-06-17_nirmedian
3	MAT	13-07-18_rededgemedian	05.07.18_bluemedian	01-08-17_ndvimedian	03-07-17_greenmedian
4	02-07-18_mtci	13-07-18_evi	05.07.18_mtci		03-07-17_rededgemedian
5	02-07-18_evi	26-07-18_ndvimedian	11.07.18_bluemedian		17-07-17_rededgemedian
6	19-07-18_bluemedian	26-07-18_mtci	11.07.18_greenmedian		14-08-17_ndvimedian
7	19-07-18_greenmedian	MAT	11.07.18_rededgemedian		14-08-17_nirmedian
8	19-07-18_redmedian		19.07.18_redmedian		14-08-17_rededgemedian
9	19-07-18_nirmedian		19.07.18_mtci		14-08-17_mtci
10	19-07-18_mtci		24.07.18_ndvimedian		03-07-17_mtci - 14-06-17_mtci
11			24.07.18_evi		
12			MAT		
R^2	0.917	0.63	0.735	0.582	0.595
MAE	26.6	44.3	14.4	47.9	47.1
Grain Yield	473	456	193	432	585

It could be relevant to see the results without maturity date (MAT) included, as this is something one must measure manually. When MAT is removed from the features used for the A-18, B-18 and C-18 fields, the results are as follows:

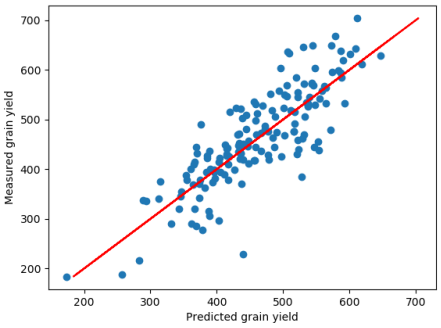
TABLE 4.5: Table showing results when removing MAT as a feature

	A-18	B-18	C-18
R^2	0.915	0.6	0.7
MAE	27.2	46.6	18.9
Avg. Grain yield	473	456	193

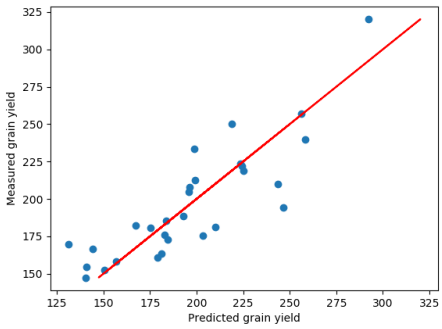
For further visualization, see the graphs in Fig 4.3, showing the predicted grain yield against the measured values. This is just the prediction using one random split, so these graphs may change using different splits. Refer to table 4.4 for more accurate evaluation. These graphs show the prediction of "unseen" data, which were not part of the training for that model.



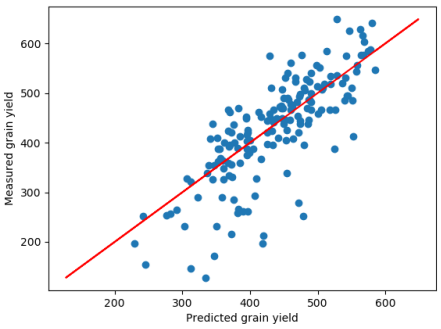
(A) Predicted against measured grain yield for A-18 using features from table 4.4



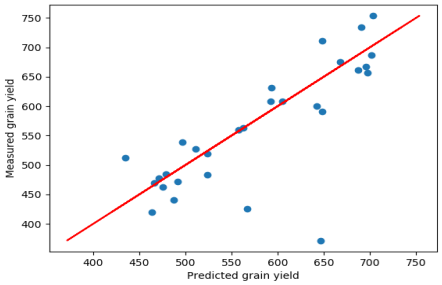
(B) Predicted against measured grain yield for B-18 using features from table 4.4



(C) Predicted against measured grain yield for C-18 using features from table 4.4



(D) Predicted against measured grain yield for A-17 using features from table 4.4



(E) Predicted against measured grain yield for C-17 using features from table 4.4

FIGURE 4.3: Predicted against measured grain yield for B-18 using features from table 4.4

4.3 Deep Learning

The grain yield was also predicted using deep leaning algorithms; ReLU and Selu, each with two different complexities described by the number of layers and the number of neurons.

In this section, two sets of features will be used for each field. The first set contains all the features, and the second contains the features found in table 4.4, using the SFFS algorithm. The results from these will then be compared to find the best set of features for each field.

For each of these sets of features, two different models will be used, and two variations of each model using different depths. The two different models used are ReLU and Selu. The first variation contains 2 dense layers and 8 neurons, and the second contains 5 dense layers and 32 neurons. This is used to see if a deeper and more complex model can detect more variation and make better predictions than the shallow one. From now, these different variants will be referred to as Relu, deep Relu, Selu and deep Selu.

The data was first split up into a training set containing 70 % of the data and a test set containing 30 % of the data. To create the best deep model, the ideal number of epochs must be set. This was done by plotting the results of a K-fold cross validation using 10 folds. This was applied to the training set for all models. See appendix C for the code used in the K-fold cross validation, as well as the code for plotting the graph. The results of one of these runs using the deep Selu model on the A-17 field can be seen in Fig 4.4. The metric used for scoring the accuracy of the models will from now on be the mean absolute error (MAE). This number is difficult to compare between fields as the grain yield differs substantially between fields. A comparison of the MAE as a percentage of the average grain yield of the specific field is therefore done in the end of this part.

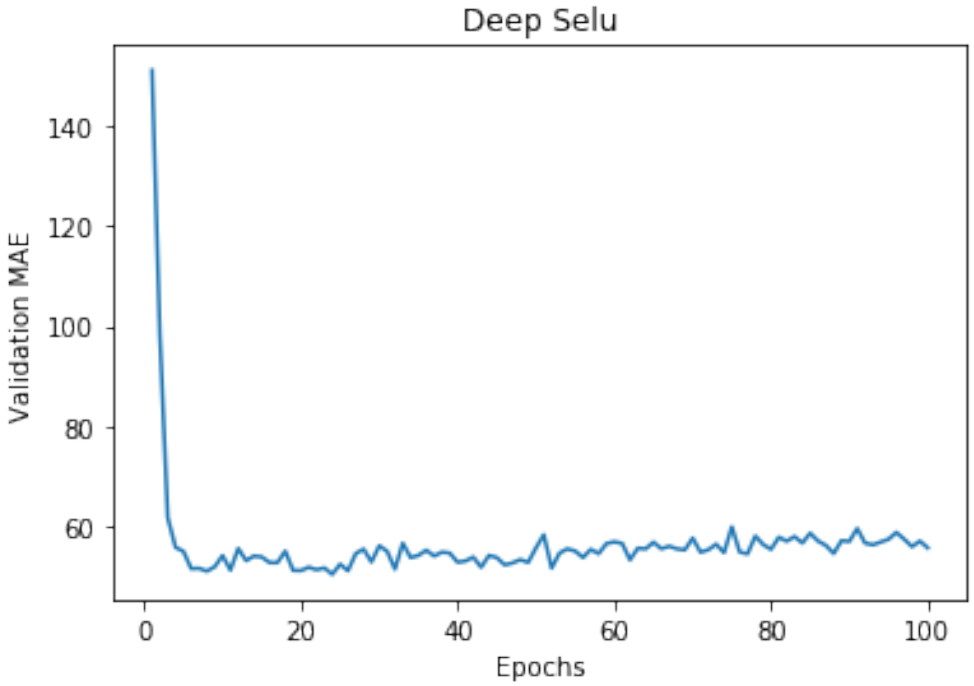


FIGURE 4.4: Development of the MAE for field A-17 using different epochs, using all features.

The ideal number of epochs were extracted from these graphs for each model-variation and each field by using the number of epochs that gives the lowest validation MAE. These models were then trained with the set number of epochs and the two sets of features on the training data. These results can be seen in table 4.6.

TABLE 4.6: The training result using the best number of epochs with the various models and fields. The average grain yield is also shown to evaluate the MAE.

		all features (MAE)	from sfs (MAE)	Difference (MAE)	Avg. Grain yield
A-18	relu:	27.8	28.2	-0.4	473
	deep relu	30.7	74.5	-43.8	
	Selu	28.3	27.9	0.4	
	deep selu	27.4	27.2	0.2	
B-18	relu	49.2	48.7	0.5	456
	deep relu	91	47.5	43.5	
	selu	45.5	43.6	1.9	
	deep selu	42.9	45.4	-2.5	
C-18	relu	44.2	36.2	8	193
	deep relu	28.8	75.2	-46.4	
	selu	24	22.5	1.5	
	deep selu	16.1	15.1	1	
A-17	relu	135.3	93.9	41.4	432
	deep relu	95.3	131	-35.7	
	selu	48.7	47.8	0.9	
	deep selu	50.6	47.5	3.1	
C-17	relu	186.2	173.4	12.8	585
	deep relu	128.8	63.1	65.7	
	selu	89.7	52.7	37	
	deep selu	90.1	42	48.1	
Sum:		1280.6	1143.4	137.2	
Average:		64.03	57.17	6.86	

For further visualization, see fig 4.5 where the results from table 4.6 are plotted.

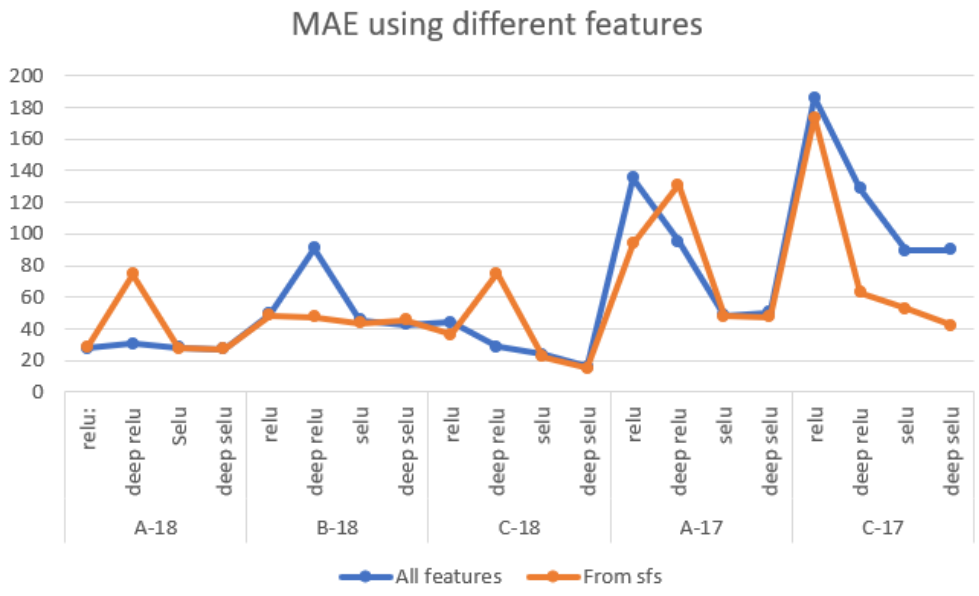


FIGURE 4.5: The MAE from table 4.6 plotted

The best models for each field were applied to the unseen test data. The result from these predictions can be seen in table 4.7.

TABLE 4.7: Results using the best model on each field applied on unseen test data. With "SFS" in the Feature-column meaning the features found by the SFFS algorithm.

	Model	Features	Epochs	MAE
A-18	Deep Selu	SFS	49	28.8
B-18	Deep Selu	All	43	49.3
C-18	Deep Selu	SFS	72	18.6
A-17	Deep Selu	SFS	20	55.6
C-17	Deep Selu	SFS	35	55.7

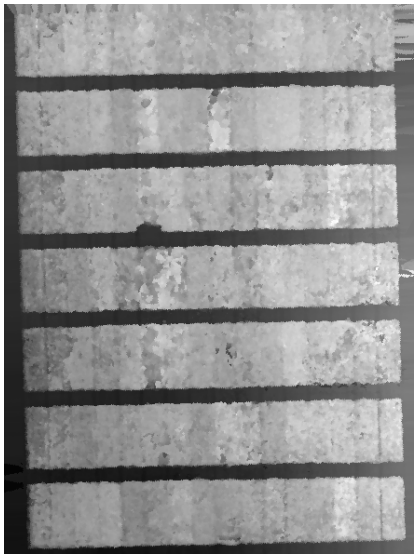
A comparison between the results using SVR and the deep models on test data can be seen in table 4.8.

TABLE 4.8: A comparison between results using the SVR and the deep model on unseen test data, showing MAE and the MAE as a percentage of the average grain yield for the specific field.

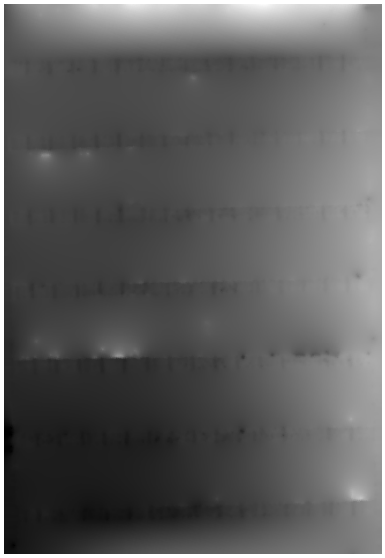
	Deep model (MAE)	SVR (MAE)	Perc. Of avg. (SVR)	Perc. Of avg. (Deep)
A-18	28.8	26.6	5.60	5.62
B-18	49.3	44.3	9.71	10.81
C-18	18.6	14.4	7.46	9.64
A-17	55.6	47.9	11.09	12.87
C-17	55.7	47.1	8.05	9.52

4.4 Plant Height Estimation

For the estimation of the plant height, DTM and DSM maps were extracted from Pix4D. The plant height was only measured once during the season, namely in the end when the growth period was over. Therefore, it was necessary to pick a date of flight late in the season to use for extracting the DTM and DSM. The maps resulting in one of the best estimations can be seen in Fig 4.6. These maps are from the 2017 season, with date 01.08.17 from the C-17 field.



(A) DSM map from C-17 taken 01.08.17.



(B) DTM map from C-17 taken 01.08.17.

FIGURE 4.6: DSM and DTM map from 01.08.17

Subtracting the DTM from the DSM resulted in the following map;

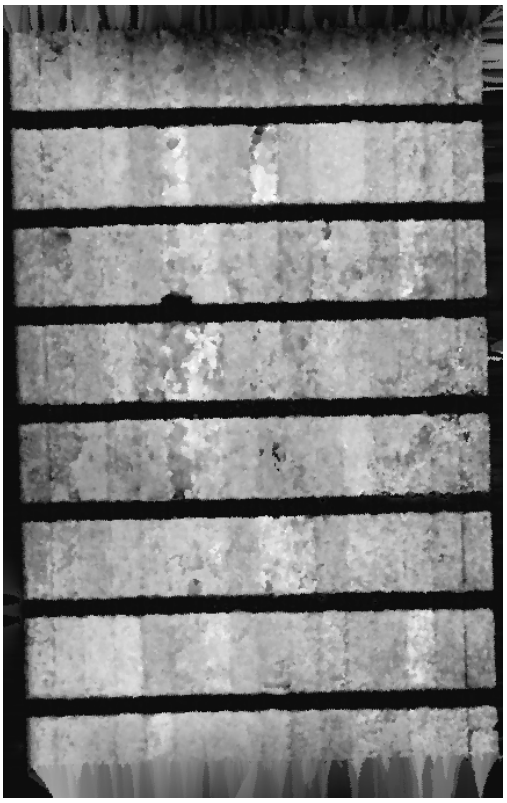


FIGURE 4.7: Map showing DSM minus DTM for C-17.

The values of the pixels in the map in fig 4.7 should give an estimation of the height in each separate plot. The same approach as explained in chapter 3.4.1 was chosen to extract the values within each plot. Instead of extracting the median value, the mean value within each

plot was chosen. Some values were removed due to being outliers, stemming from the image quality. Plotting the resulting height estimations against the measured values resulted in the following figure; This resulted in an R^2 value of 0.33 with MAE of 6.8 cm. The average

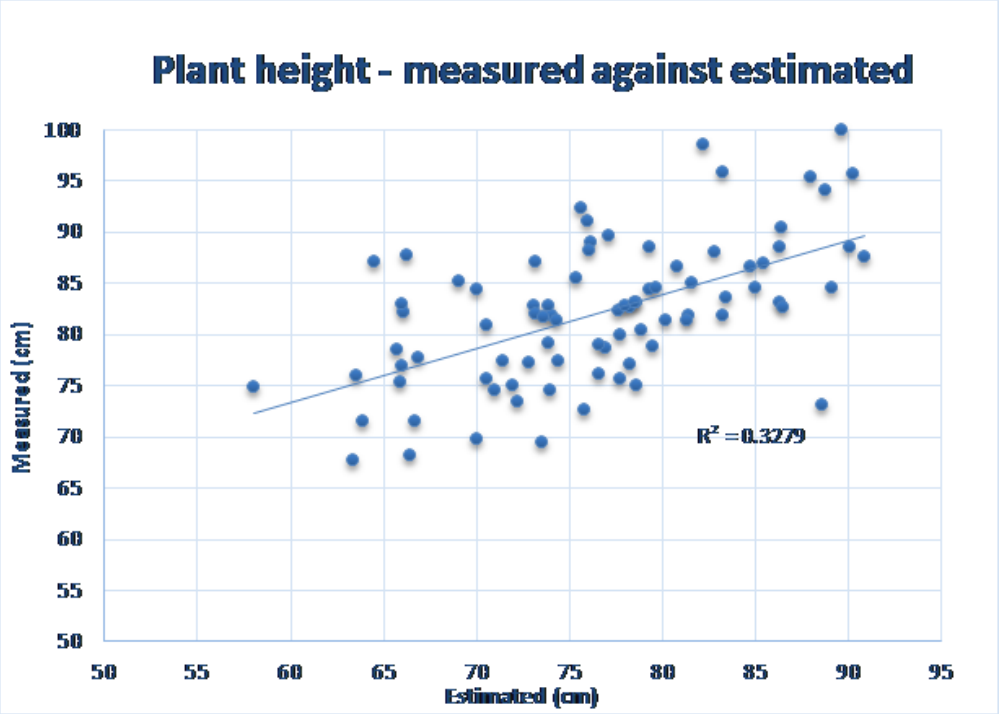


FIGURE 4.8: Estimated plant height plotted against measured plant height, resulting in R^2 value of 0.33.

measured plant height was 82 cm for comparison with the MAE, thus getting within 8.3% of the average value.

Chapter 5

Discussion

In this chapter the different parts of this thesis, ranging from image capturing to results from the data analysis will be discussed. Some comparisons with similar projects will also be shown.

5.1 Image Acquisition

Due to problems with the automatic flying with the drone, manual flight was used during all flights during the 2018 season. This resulted in not being able to guarantee that the images were taken with enough coverage, in comparison to automatic flying, thus worsening the image and map quality used for the analysis. Several maps created were also deemed unusable and were not included in the analysis, thus reducing the pool of data. This could also worsen the amount of relevant information in the maps for predicting grain yield. This could be one of the reasons for the high R^2 value for the A-18 field, since this field generally had good coverage and image quality, in comparison to the B-18 field which barely had two usable dates of image data. The drone was also unstable in flight sometimes, resulting in sudden drops in altitude and wiggling back and forth. There could also have been placed more GCP's to improve the stitching of images into maps.

5.2 Data Extraction

Even though the process of using QGIS to extract data from the maps is precise, it is quite tedious and time consuming. This will especially be the case if even more fields and dates are included. A possible solution to this is using deep learning for image segmentation to extract each plot automatically. This could be done using Convolutional Neural Networks (CNN), as described by Ankit [38], thereby cutting the time used to extract the data and being able to handle potentially more fields and dates.

5.3 Data Analysis

The models used for prediction in this thesis were not very specialized to this exact task. This is especially true for the deep learning algorithms, as they used only dense sequential layers. This could be part of the reason why the deep learning models gave a lower accuracy than SVR in this thesis. Another part is that for each field, we have a maximum of only a few hundred samples, but a deep learning algorithm generally requires more data.

More than one machine learning algorithm should also have been used. In line with the "No free lunch" theorem, several models should have been tested and compared to find the best result. Another model which could have been used is the Random Forest Regressor from Scikit-learn [39].

One weakness with the setup of the predictions using the deep learning was how the test set was handled. 30 % of the data was held back in the beginning, and this was the only set used for testing. However, with the SVR, 100 different splits were used, and an average of the result was made. However, during the training of the deep learning model, there were constantly new validation splits of the training data. This only used 70 % of the data and could probably have been done in a better way. One positive part of this is that the test data used was completely unseen during training and therefore one can be sure that none of this information leaked into the training of the model and while finding the best number of epochs.

This thesis presents an easy to use, yet powerful method of applying the Sequential Feature Selector (SFS) and Support Vector Regression (SVR) to analyze the data. This makes it easy to analyze unlinear data, finding the most important features and making a prediction on unseen data. The use of these tools could improve results found by Hassan [40], where multispectral image analysis was used on fields of bread wheat, as they used only linear regression and just looked at correlation rather than predictions. The same applies to [41], where spectral reflectance was used for indirect selection of grain yield. This type of approach could also have helped [42], where hyperspectral data was used to predict grain yield. Here, the SFS could have been implemented alongside the SVR or another preferred model, to find the most important bands from the hyperspectral camera.

5.4 Prediction of Grain Yield

The resulting R^2 values using the SVR to predict grain yield ended up being 0.917, 0.63, 0.735, 0.582 and 0.595 for A-18, B-18, C-18, A-17 and C-17, respectively. These numbers are consistent with other researchers results [43],[44] for predicting grain yield of maize and rice using image analysis. These articles refer to resulting R^2 values ranging from 0.71 to 0.92.

The reason for the prediction for the A-18 field being much better than the other fields could be a result of several factors. One reason could be the image quality generally being good for most flights over that field. This field was much easier to work with in Pix4D than the B-18 field, which could be due to the field being more level, problems with the drone or other coincidences.

Note that even though removing the only manual measurement (MAT) from the prediction did give a drop in performance, see table 4.5, it was not by a substantial amount. This means that only using spectral data can produce good predictions of grain yield.

Another problem is to generalize these models to predict yield for new fields, which have not been included in training. All predictions done in this thesis have been a result of training the models on the field that they are meant to predict, although not on the same plots. One then has to create a model general enough to be able to look past field specific traits and only focus on the most general of traits that can explain grain yield. This would have to be done by using several years' worth of data from several fields and make a model using all these data. Then one could hopefully be able to create a robust enough model where one could input entirely new data from different fields and still get a good prediction.

When training the different models, it was clear that the ReLU and Selu used much more time than the SVR. The training and setting the epoch-parameter for ReLU and Selu took around 20 minutes, while training and setting the C-parameter for SVR took around 2 minutes. The SFS algorithm took around 8 minutes. This further the incentive of using the machine learning instead of deep learning to be more time efficient as well as the higher accuracy of predictions.

5.5 Importance of Features

An important part of any research handling complex data with several variables and features, is to be able to find the important parts of the data without being distracted by possible noise. If only a handful of the features is worth prioritizing, this could make the work of optimizing plant phenotyping an easier task.

To analyze the importance of different features in this thesis, it is necessary to inspect table 4.2 and 4.4 from chapter 4. To begin with, the most important features as seen in table 4.4, are spread out over the entire season, and it is not easy to see a distinct time of the season that is the most important. The data used in this thesis also lacks enough datapoints from June, which shows the early growth period and possibly contains important information for predicting yield.

The number of times each feature is present for each field in table 4.4 can be seen in table 5.1 below, where nan-values indicate that it is not present in the list of most important features for that field.

TABLE 5.1: Table showing number of times each feature is present in the table for important features(table 4.4) for each field, where nan means not present.

Index	a18	b18	c18	a17	c17
blue	2	1	2	nan	nan
evi	1	1	2	nan	nan
green	1	1	2	1	2
mat	1	1	1	nan	nan
mtci	3	1	2	1	2
nir	1	nan	nan	nan	2
red	2	nan	nan	1	nan
ndvi	nan	2	2	1	2
rededge	nan	1	2	nan	3
mitci-diff	nan	nan	nan	nan	1

The difference between indices used in 2018 and 2017 is the inclusion of the blue band in 2018, and thereby the EVI index. There is a clear improvement in prediction from 2017 to 2018, which could partly be due to the addition of the blue band. It is also clear from table 5.1, that these two features are important for predictions as they are present at least once for all the fields from 2018. The MTCI index also proves to be important as it is featured for all the fields, the same as the green band. The NDVI index still seems to contain important information even though it is said to be quickly saturated. The MTCI and REG are the only features being present three times for the same field and would definitely seem to be important. The maturity date (MAT) is also present where it was recorded, except for A-17 (MAT was not measured for C-17) and gives a visible increase in prediction as seen when comparing table 4.5 and 4.4. The feature being the least represented is the difference in MTCI. This could be due to the fact that the models can handle the information from these by using the initial MTCI values themselves.

5.6 Variations Within The Fields

With the way the data extracted from QGIS were used, the results in this thesis are prone to be affected by the variations of external factors within the fields other than just the different types of cultivars. This could be moisture and other variations in the soil making the grain yield vary. This could especially be the case for the 2018 season as the temperature was very high that summer, and there could therefore be big differences in soil conditions within each field. These extra variations within each field could make it easier for the model to predict grain yield. This could have been handled by using a different function in the SAS code, removing the variations in data due to external factors, but this was not done. However, the results still show that the variations causing the various grain yields can be picked up by the multispectral data.

5.7 Estimating Plant Height

According to the results using the DSM minus DTM, it is not easy to get an accurate height estimation of these types of plants. One change that could help this approach is to fly over the field with the camera before the growing season. Then it would be possible to create the DTM more accurately since there are no plants present, and then use this map as the DTM for all other height estimations. The first flight in 2018 was done several weeks into the growing period. This resulted in the plants being quite tall at the time of flight. The idea of flying before the growth season was meant to be done in the 2018 season, but due to a misunderstanding of the position of the field, this was not done.

Another possible source for the error from the estimation is the manual measurements of the height. This was only done once in the end of the season at the point where the plants were no longer growing. This meant that when doing the analysis, it was necessary to find a date of flight close to the date of measurement and being totally reliant on that map. It could have been possible to use different maps late in the season from after the growth had stopped. The manual measuring of the height was not very accurate, with it being more of an average using a couple of points with a measuring stick. This could have resulted in an uncertainty of +- 5cm, which could have made it difficult for the estimation to be precise.

When comparing to the results from Grindbakken [7] which was also a part of the vPheno project, the results from this thesis were an improvement. Grindbakken reports a result of the correlation between the computer estimated and the measured heights of -0.376. We achieved an R^2 value of 0.328 and MAE of 6.8 cm, where the average measured height was 82 cm. Another research article using the DSM approach on a wheat field [45], got an average difference of 4.66 cm between the estimated and measured plant heights. This is comparative

to the result in this thesis. When comparing to the results achieved by Hassan [46] of R^2 values up to 0.96, who also used methods using DSM's, the results in this thesis are much worse. Hassan also used some different treatments on the data to achieve these results.

Finally, it might not be necessary to get a very high accuracy for this estimation. This depends of what the purpose is, whether to get some idea of the height or a more precise estimation. This is something that must be further agreed upon.

5.8 The Drone

One problem that became very apparent during image capturing was problems with the drone. The drone was quite unstable, both with regards to flight but also with GPS-signals. This is probably mostly due to the extra equipment that was attached to the drone. This included the extra GPS, the light sensor and the multispectral camera. The extra weight and the potential blocking of the GPS signals from the drone itself made the procedure of capturing the images more cumbersome than it should have been. One possible solution to this is to use a larger drone capable of carrying more weight and more attachments without being affected as much with possible conflicting signals. The automatic flying was tried without any of the extra equipment, and that worked well.

5.9 Weather Conditions

The temperatures during May, June and July of 2018 in Norway were very high, with the average temperature in May being 4.2 degrees Celsius above the normal temperature for that month. This resulted in the hottest May recorded in Norway in modern times [47]. This extreme heat affects the growth of the plants and could affect the ability to predict grain yield in comparison to colder months. The summer of 2017 was generally a cold summer, which could result in the opposite effects than during the 2018 season. This is a major reason why this type of research requires several years of trials to produce the most robust results, as the weather changes from year to year.

5.10 Future Directions

For future directions in this field, I would advise making sure the data collection is as efficient as possible. A large part of this point would be to have a drone that is capable of flying automatically with the extra equipment. This would both be more time-efficient and help improve image quality. By improving image quality, the results of the analysis could see a big improvement.

A different room for improvement is to make the data extraction more efficient as mentioned earlier in part 5.2. Then there could be made a more automated pipeline for extracting and analysing the data, using a more specialized deep learning method.

In the case of estimating plant height, this should be repeated to achieve the optimal way of estimating. The results in this topic from different researchers varies significantly, and there is a need to find the best methodology. These estimations could also vary greatly depending on the flight-pattern and further use of the drone.

Finally, there is a need for repeating this research during several seasons to gather as much data as possible to create a reliable model for prediction of grain yield.

Chapter 6

Conclusion

Two questions were asked in the introduction to this thesis, first being predicting grain yield using multispectral images and the other question of estimating plant height. These questions will here be evaluated based on the results presented in this thesis. For this thesis a UAV with a multispectral and an RGB camera was flown over three wheat-fields to extract data. Data from two fields of wheat from 2017 was also included.

The results given by the tools used in this thesis shows that the grain yield can be predicted by a large amount by the multispectral data. This is shown by the R^2 value being between approximately 0.6 and 0.9 for the different fields. The mean absolute error ranges between 5.6% and 11.1% of the average grain yield for the specific fields.

The second question of being able to estimate plant height using the RGB camera achieved less satisfactory results. The R^2 value between the estimated and the measured values being approximately 0.33, with mean absolute error of 6.8 cm.

This thesis presents an opportunity to directly predict grain yield using multispectral data and presents one option of estimating plant height. By using the same outline, plant breeders could use the methods presented in this thesis to predict grain yield, find the most important features for this prediction and use these results to improve efficiency of plant breeding. The same machine learning algorithms can also easily be expanded by adding different features that the user finds important.

As a conclusion; this thesis gives optimistic results, but it is necessary to produce more data and improve methodology to create robust models for predicting grain yield and estimating plant height. It is also necessary to remove variations in the data due to external factors, such as soil conditions.

Bibliography

[1] Food and Agriculture Organization of the UN. *2018 THE STATE OF FOOD SECURITY AND NUTRITION IN THE WORLD*. 2018. URL: http://www.fao.org/state-of-food-security-nutrition/en/.

[2] Standing Committee on Business and Industry of the Storting. *Innstilling fra næringskomiteen om landbruks og matpolitikken*. 2012. URL: https://www.stortinget.no/globalassets/pdf/innstillinger/stortinget/2011-2012/inns-201112-234.pdf.

[3] Food and Agriculture Organization of the UN. *Increasing food production without damaging the environment*. May 2017. URL: http://www.fao.org/news/story/en/item/889671/icode/].

[4] Adil Aly Atef Hamdy. *Land Degradation, Agriculture Productivity And Food Security*. 2014.

[5] The Wheat Initiative. *Breakout session P1.1 National Food Security – The Wheat Initiative – an International Research Initiative for Wheat Improvement*. Nov. 2012. URL: http://www.fao.org/docs/eims/upload/306175/Briefing%20Paper%20(3)-Wheat%20Initative%20-%20H%C3%A9l%C3%A8ne%20Lucas.pdf.

[6] Ingunn Burud Eivind Bleken. "Exploring robots and UAVs as phenotyping tools in plant breeding". Master's thesis. Realtek, 2017.

[7] Ole Kristian Grindbakken. "Phenotyping studies of spring wheat by multispectral image analysis". Master's thesis. Realtek, 2018.

[8] Danfen Huang Lei Li Qin Zhang. "A Review of Image Techniques for Plant Phenotyping". In: *PubMed* (Oct. 2014).

[9] G Velu and Ravi Prakash. *Phenotyping in Wheat Breeding*. Jan. 2014. DOI: 10.1007/978-1-4614-8320-5_2.

[10] Gary N. Atlin, Jill E. Cairns, and Biswanath Das. "Rapid breeding and varietal replacement are critical to adaptation of cropping systems in the developing world to climate change". In: *Global Food Security* 12 (2017), pp. 31–37. ISSN: 2211-9124. DOI: https://doi.org/10.1016/j.gfs.2017.01.008. URL: http://www.sciencedirect.com/science/article/pii/S2211912416300931.

[11] Unseok Lee et al. "An automated, high-throughput plant phenotyping system using machine learning-based plant segmentation and image analysis". In: *PLOS ONE* 13.4 (Apr. 2018), pp. 1–17. DOI: 10.1371/journal.pone.0196615. URL: https://doi.org/10.1371/journal.pone.0196615.

[12] Humboldt State University. *Spectral Reflectance*. 2018. URL: http://gsp.humboldt.edu/OLM/Courses/GSP_216_Online/lesson2-1/reflectance.html.

[13] Lei Li, Qin Zhang, and Danfeng Huang. "A Review of Imaging Techniques for Plant Phenotyping". In: *Sensors (Basel, Switzerland)* 14 (Nov. 2014), pp. 20078–20111. DOI: 10.3390/s141120078.

[14] Harris Geospatial Solutions. *Spectral Indices*. May 2019. URL: https://www.harrisgeospatial.com/docs/SpectralIndices.html.

[15] Harris Geospatial Solutions. *Narrowband Greenness*. 2019. URL: http://www.harrisgeospatial.com/docs/narrowbandgreenness.html.

[16] GISGeography. *What is NDVI (Normalized Difference Vegetation Index)?* Feb. 2018. URL: https://gisgeography.com/ndvi-normalized-difference-vegetation-index/.

[17] Z.-X Wang, C Liu, and A Huete. "From AVHRR-NDVI to MODIS-EVI: Advances in vegetation index research". In: *Acta Ecologica Sinica* 23 (Jan. 2003), pp. 979–987.

[18] NASA Earth Observatory. *Measuring Vegetation (NDVI and EVI)*. Aug. 2000. URL: https://earthobservatory.nasa.gov/features/MeasuringVegetation/measuring_vegetation_4.php.

[19] Athos Agapiou, Diofantos G. Hadjimitsis, and Dimitrios D. Alexakis. "Evaluation of Broadband and Narrowband Vegetation Indices for the Identification of Archaeological Crop Marks". In: *Remote Sensing* 4.12 (2012), pp. 3892–3919. ISSN: 2072-4292. DOI: 10.3390/rs4123892. URL: http://www.mdpi.com/2072-4292/4/12/3892.

[20] J. Delegido et al. "A red-edge spectral index for remote sensing estimation of green LAI over agroecosystems". In: *European Journal of Agronomy* 46 (2013), pp. 42–52. ISSN: 1161-0301. DOI: https://doi.org/10.1016/j.eja.2012.12.001. URL: http://www.sciencedirect.com/science/article/pii/S1161030112001542.

[21] Su Zhang and Liangyun Liu. "The potential of the MERIS Terrestrial Chlorophyll Index for crop yield prediction". In: *Remote Sensing Letters* 5.8 (2014), pp. 733–742. DOI: 10.1080/2150704X.2014.963734. eprint: https://doi.org/10.1080/2150704X.2014.963734. URL: https://doi.org/10.1080/2150704X.2014.963734.

[22] Rohith Gandhi. *Support Vector Machine — Introduction to Machine Learning Algorithms*. June 2018. URL: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[23] Pushkar Mandot. *What is the Significance of C value in Support Vector Machine?* Sept. 2017. URL: https://medium.com/@pushkarmandot/what-is-the-significance-of-c-value-in-support-vector-machine-28224e852c5a.

[24] Tejumade Afonja. *Kernel Functions*. Jan. 2017. URL: https://towardsdatascience.com/kernel-function-6f1d2be6091.

[25] Larry Hardesty. *Explained: Neural networks*. Apr. 4, 2017. URL: https://www.csail.mit.edu/news/explained-neural-networks.

[26] Niklas Donges. *Pros and Cons of Neural Networks*. Apr. 2018. URL: https://towardsdatascience.com/hype-disadvantages-of-neural-networks-6af04904ba5b.

[27] Satya Mallick. *Neural Networks: A 30,000 Feet View for Beginners*. May 2, 2017. URL: https://www.learnopencv.com/neural-networks-a-30000-feet-view-for-beginners/.

[28] Isaac Changhau. *Activation Functions in Neural Networks*. May 22, 2017. URL: https://isaacchanghau.github.io/post/activation_functions/.

[29] Elior Cohen. *SELU — Make FNNs Great Again (SNN)*. July 21, 2017. URL: https://towardsdatascience.com/selu-make-fnns-great-again-snn-8d61526802a9.

[30] Sebastian Raschka. *Sequential Feature Selector*. 2018. URL: http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/#example-2-toggling-between-sfs-sbs-sffs-and-sbfs.

[31] Sebastian Raschka. *About Feature Scaling and Normalization – and the effect of standardization for machine learning algorithms*. July 11, 2014. URL: https://sebastianraschka.com/Articles/2014_about_feature_scaling.html.

[32] MicaSense Inc. *RedEdge-M User Manual*. Nov. 2017.

[33] MicaSense. *Light Sensors: the Basics (DLS and Sunshine Sensor)*. Aug. 30, 2017. URL: https://support.micasense.com/hc/en-us/articles/115002782008-Light-Sensors-the-Basics-DLS-and-Sunshine-Sensor-.

[34] QGIS. *QGIS User Guide. Features*. 2019.

[35] Scikit-learn. *scikit-learn Machine Learning in Python*. 2019. URL: https://scikit-learn.org/stable/.

[36] Keras. *Keras: The Python Deep Learning library*. 2019. URL: https://keras.io/.

[37] Sebastian Raschka. *Sequential Feature Selector*. 2019. URL: http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/.

[38] Utkarsh Ankith. *Semantic Segmentation of Aerial images Using Deep Learning*. Feb. 4, 2019. URL: https://towardsdatascience.com/semantic-segmentation-of-aerial-images-using-deep-learning-90fdf4ad780.

[39] Scikit-learn. *RandomForestRegressor*. 2019. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

[40] Muhammad Hassan et al. "Time-Series Multispectral Indices from Unmanned Aerial Vehicle Imagery Reveal Senescence Rate in Bread Wheat". In: *Remote Sensing* 10 (May 2018). DOI: 10.3390/rs10060809.

[41] Shiferaw A. Gizaw, Kimberly Garland-Campbell, and Arron H. Carter. "Use of spectral reflectance for indirect selection of yield potential and stability in Pacific Northwest winter wheat". In: *Field Crops Research* 196 (2016), pp. 199–206. ISSN: 0378-4290. DOI: https://doi.org/10.1016/j.fcr.2016.06.022. URL: http://www.sciencedirect.com/science/article/pii/S0378429016302088.

[42] Osval Montesinos-López et al. "Predicting grain yield using canopy hyperspectral reflectance in wheat breeding data". In: *Plant Methods* 13 (Jan. 2017), pp. 1–23. DOI: 10.1186/s13007âĂŠ016âĂŠ0154âĂŠ2.

[43] Adeline Ngie and Faruk Ahmed. "Estimation of Maize grain yield using multispectral satellite data sets (SPOT 5) and the random forest algorithm". In: *South African Journal of Geomatics* 7 (Feb. 2018), p. 11. DOI: 10.4314/sajg.v7i1.2.

[44] X. Zhou et al. "Predicting grain yield in rice using multi-temporal vegetation indices from UAV-based multispectral and digital imagery". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 130 (2017), pp. 246–255. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2017.05.003. URL: http://www.sciencedirect.com/science/article/pii/S0924271616302337.

[45] Nusret Demir et al. "Automated Measurement of Plant Height of Wheat Genotypes Using a DSM Derived from UAV Imagery". In: vol. 2. Mar. 2018, p. 5163. DOI: 10.3390/ecrs-2-05163.

[46] Muhammad Hassan et al. "Accuracy assessment of plant height using an unmanned aerial vehicle for quantitative genomic analysis in bread wheat". In: *Plant Methods* 15 (Apr. 2019). DOI: 10.1186/s13007-019-0419-7.

[47] Per Vidar Raunholm Roy Hilmar Svendsen. *Varmeste mai maaned i moderne tid*. June 1, 2018. URL: https://www.yr.no/artikkel/varmeste-mai-maned-i-moderne-tid-1.14064858.

Appendix A

SAS Code

```
proc import datafile='c:\sas\2018\18bmlgi1.csv'  out=dataset replace;
delimiter=',';

proc print;

proc mixed covtest data=dataset;
class Line Rep Block col;
model GrainYield = Line /outp=resids;
random rep  block (rep) Col/s;
lsmeans  Line ;
ods output LSMeans=lsm;


proc export data=resids outfile='c:\sas\2018\residuals.csv' replace;
delimiter=',';

proc export data=lsm outfile='c:\sas\2018\lsmeans.csv' replace;
delimiter=',';

run;
```

FIGURE A.1: SAS code for B-18 and A-17

```
proc import datafile='c:\sas\2018\graminor.csv'  out=dataset replace;
delimiter=',';

proc print;

proc mixed covtest data=dataset;
class Line Nursery Rep col;
model GrainYield = Line /outp=resids;
random Nursery rep(nursery)  Col/s;
lsmeans  Line ;
ods output LSMeans=lsm;

proc export data=resids outfile='c:\sas\2018\residuals.csv' replace;
delimiter=',';

proc export data=lsm outfile='c:\sas\2018\lsmeans.csv' replace;
delimiter=',';

run;
```

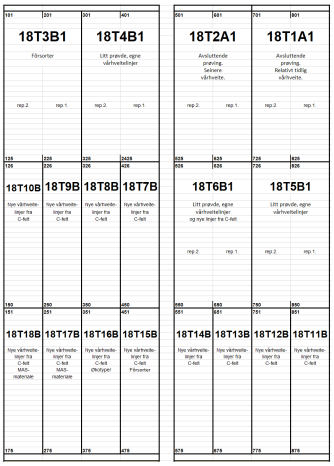
FIGURE A.2: SAS code for A-18

```
proc import datafile='c:\sas\2018\18bmlrobot.csv'  out=feltdata replace;
delimiter=',';

proc print;

proc mixed covtest data=feltdata;
class Entry N_level Rep Block Col;
model GrainYield = entry N_level entry*N_level /outp=resids;
random  N_level*rep block(N_level*rep) Col /s;
lsmeans  entry N_level entry*N_level ;
ods output LSMeans=lsm;

proc export data=resids outfile='c:\sas\2018\residuals.csv' replace;
delimiter=',';

proc export data=lsm outfile='c:\sas\2018\lsmeans.csv' replace;
delimiter=',';

run;
```

FIGURE A.3: SAS code for C-18 and C-17

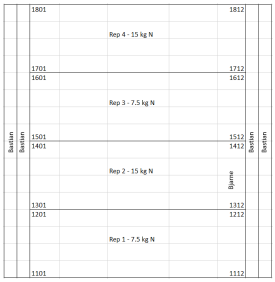
Appendix B

Field Maps

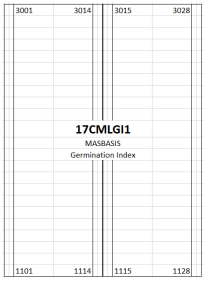


(A) Field map for A-18

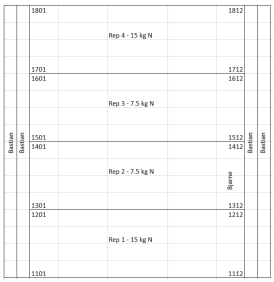
(B) Field map for B-18



(C) Field map for C-18



(D) Field map for A-17



(E) Field map for C-17

FIGURE B.1: Field maps for all fields used

Appendix C

Deep Learning Code

Deep Learning-keras

May 15, 2019

In [0]:
```
import keras
keras.__version__
import pandas as pd
import copy
import numpy as np
from sklearn.model_selection import train_test_split

import matplotlib.pyplot as plt
%matplotlib inline

from keras.models import Sequential
from keras.layers import Dense

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.model_selection import cross_val_score
```

In [0]:
```
!pip install -q xlrd
```

In [0]:
```
from google.colab import drive
drive.mount('/content/drive')
```

In [0]:
```
dfferdig = pd.read_excel("drive/My Drive/DAT300/fullgraminor2018.xlsx", sheet_name=0)

dfferdig.set_index('26-06-18Unnamed: 0', inplace=True)
```

In [0]:
```
features = ['26-06-18_bluemedian', '26-06-18_redmedian', '26-06-18_mtci', 'MAT',
       '02-07-18_mtci', '02-07-18_evi', '19-07-18_bluemedian',
       '19-07-18_greenmedian', '19-07-18_redmedian', '19-07-18_nirmedian',
       '19-07-18_mtci']


df_x = dfferdig[features]
df_y = dfferdig['26-06-18GrainYield']
```

In [0]:
```
df_x.shape
```

In [0]:
```
scaler= StandardScaler()
X_train, X_test, y_train, y_test = train_test_split(
```

```
        df_x, df_y, test_size=0.3, random_state=1)
    X_train.iloc[:, :] = scaler.fit_transform(X_train)
    X_test.iloc[:, :] = scaler.transform(X_test)
```

In [0]:
```
def build_model(act, lay, nur):

    # Input:
    #     act: which activations function, denoted by 'name_of_activation' i.e. 'relu'
    #     lay: integer, number of layers
    #     nur: number of neurons within each layer. possibly we would want a different
    #          but for now we keep the same number of neurons for all layers

    model = Sequential()
    model.add(Dense(nur, activation=act, input_shape=(X_train.shape[1],)))
    n=1
    while n <= lay:
      model.add(Dense(nur, activation=act))
      n += 1

    model.add(Dense(1, activation=act))
    model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])

    return model
```

In [0]:
```
def K_fold(X_train, y_train, model_activation, model_layers, model_neurons, k, num_epo
  """Custom function for K-fold Cross validation for both keras
  Inspired by the Colab notebook 'DAT300 - 05 - regression example.ipynb'"""

  num_val_samples = X_train.shape[0] // k
  all_scores = np.zeros(k)
  all_mae_histories = []

  for i in range(k):
      print('processing fold #', i+1)

      # Prepare the validation data: data from partition # k
      val_data = X_train[i * num_val_samples: (i + 1) * num_val_samples]
      val_targets = y_train[i * num_val_samples: (i + 1) * num_val_samples]

      # Prepare the training data: data from all other partitions
      partial_train_data = np.concatenate(
          [X_train[:i * num_val_samples],
           X_train[(i + 1) * num_val_samples:]],
          axis=0)

      partial_train_targets = np.concatenate(
          [y_train[:i * num_val_samples],
           y_train[(i + 1) * num_val_samples:]],
```

```
                    axis=0)

                # Build the Keras model (already compiled)
                model = build_model(model_activation, model_layers, model_neurons)

                # Train the model (in silent mode, verbose=0)
                history = model.fit(partial_train_data, partial_train_targets,
                                    validation_data=(val_data, val_targets),
                                    epochs=num_epochs, batch_size=1, verbose=0)

                # Evaluate the model on the validation data
                val_mse, val_mae = model.evaluate(val_data, val_targets, verbose=0)
                all_scores[i] = val_mae

                mae_history = history.history['val_mean_absolute_error']
                all_mae_histories.append(mae_history)

            average_mae_history = [
                np.mean([x[i] for x in all_mae_histories]) for i in range(num_epochs)]
            print(min(average_mae_history))
            if history:
              return average_mae_history
            else:
              return all_scores
```

```
In [0]: def plot_epochs(mae_history, title):
            plt.plot(range(1, len(mae_history) + 1), mae_history)
            plt.title(title)
            plt.xlabel('Epochs')
            plt.ylabel('Validation MAE')
            plt.show()
```

0.1 result table

```
In [0]: K_fold_results = pd.DataFrame({'Model': ['ReLU','Deep ReLU','selu', 'Deep selu', 'LinR
                                  'Validation MAE': np.nan,
                                  'Hyperparameters': np.nan})
        K_fold_results.set_index('Model', inplace=True)
        K_fold_results = K_fold_results[['Validation MAE', 'Hyperparameters']]

        K_fold_results
```

0.2 ReLU

```
In [0]: relu_layers = 2
        relu_neurons = 8
        relu_history = K_fold(X_train=X_train, y_train=y_train, model_activation='relu', model_
```

```
print(min(relu_history))
plot_epochs(relu_history, 'ReLU')
```

0.3 Deep ReLU

```
In [0]: deep_relu_layers = 5
        deep_relu_neurons = 32
        deep_relu_history = K_fold(X_train=X_train, y_train=y_train, model_activation='relu', r

        plot_epochs(deep_relu_history, 'Deep ReLU')
```

0.4 Selu

```
In [0]: selu_layers = 2
        selu_neurons = 8
        selu_history = K_fold(X_train=X_train, y_train=y_train, model_activation='selu', model_

        plot_epochs(selu_history, 'Selu')
```

0.5 deep selu

```
In [0]: deep_selu_layers = 5
        deep_selu_neurons = 32
        deep_selu_history = K_fold(X_train=X_train, y_train=y_train, model_activation='selu', r

        plot_epochs(deep_selu_history, 'Deep Selu')
```

0.6 Linear regression

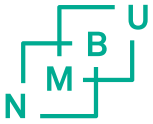
```
In [0]: # Linear
        LinReg_lin_score = cross_val_score(LinearRegression(), X_train, y_train, cv=5, scoring=
        K_fold_results.loc['LinReg', :] = [abs(np.mean(LinReg_lin_score)), ' ']

        # Quadratic
        X_quad = PolynomialFeatures(degree=2).fit_transform(X_train)
        LinReg_quad_score = cross_val_score(LinearRegression(), X_quad, y_train, cv=5, scoring=
        K_fold_results.loc['LinReg Quadratic', :] = [abs(np.mean(LinReg_quad_score)), ' ']

        # Cubic
        X_cubic = PolynomialFeatures(degree=3).fit_transform(X_train)
        LinReg_cub_score = cross_val_score(LinearRegression(), X_cubic, y_train, cv=5, scoring=
        K_fold_results.loc['LinReg Cubic', :] = [abs(np.mean(LinReg_cub_score)), ' ']
```

```
In [0]: K_fold_results
```

0.7 Setting best models

```
In [0]: bestdeep = build_model(act='selu', lay=5, nur=32)#set the parameters for the best mode
        bestdeep.fit(X_train, y_train, epochs=49, batch_size=1, verbose=0)
```

```
        #bestdeep.predict(X_test)
        bestdeep.evaluate(X_test, y_test, batch_size=1)

In [0]: bestlin = PolynomialFeatures(degree=3).fit_transform(X_test)
        lin_score = cross_val_score(LinearRegression(), bestlin, y_test, cv=5, scoring='neg_mea
        K_fold_results.loc['LinReg Cubic', :] = [abs(np.mean(lin_score)), ' ']

In [0]: K_fold_results
```



Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway