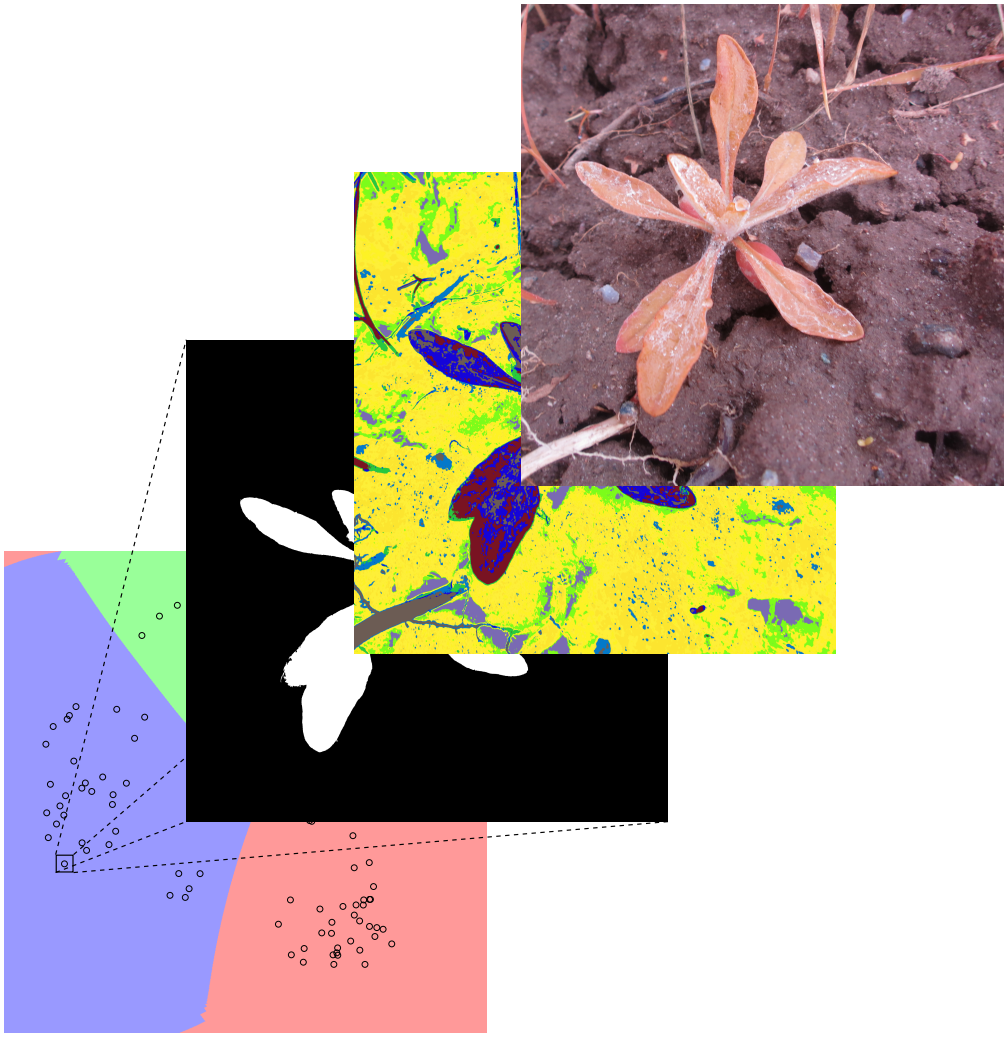
CHALMERS
UNIVERSITY OF TECHNOLOGY



Weed classification and measurement

From computer vision to machine learning

Master's thesis in Complex Adaptive System

Victor Nilsson

Department of Mathematical Siences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017:NN

Weed classification and measurement

From computer vision to machine learning

VICTOR NILSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Mathematical Statistics
Chalmers University of Technology
Gothenburg, Sweden 2017

Weed classification and measurement
From computer vision to machine learning
VICTOR NILSSON

Supervisor: Mats Rudemo, Department of Mathematical Sciences
Examiner: Mats Rudemo, Department of Mathematical Sciences

Supervisor @ farmdrones: Therese von Hackwitz

,

Master's Thesis 2017:NN
Department of Mathematical Sciences
Division of Mathematical Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A schematic view of the computer vision to machine learning process.

Typeset in LaTeX
Printed by [Name of printing company]
Gothenburg, Sweden 2016

Weed classification and measurement
From computer vision to machine learning
VICTOR NILSSON
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Keywords: lorem, ipsum, dolor, sit, amet, consectetur, adipisicing, elit, sed, do.

Acknowledgements

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Victor Nilsson, Gothenburg, June 2017

Contents

List of Figures

List of Tables

1

Introduction

"Boot up"

Since the dawn of computers, humans have had access to computational power previously far beyond reach. In the beginning this was used to numerically solve Ordinary Differential Equations, (ODE), that had no exact explicit solution. With mathematical computational, the computer was able to retrieve solutions to problems previously far beyond human reach. Although, with the immense computational power that the computer possessed it still lacked something that made man superior still. Intuition, creativity and being able to reason about results given prior knowledge for similar problems. As the computer had to be programmed and do exactly as programmed, it was as good as the code giving it instructions. This made the computer static, the program was never able to learn from previous problems.

This limitation is something that is progressively being erased today. Machine learning is a field of computer science, in which one studies algorithms that learn from data to make statistical predictions on new data. In this thesis, different machine learning algorithms are discussed and compared on an application in image recognition.

1.1 Background

Over-fertilization of crops and heavy use of herbicides in weed control introduces chemicals into the ecosystem. Reducing the amount of these substances is therefore of a problem requiring attention. In order to make best use of these products, local measurements of the required chemicals can be used for optimal distribution. For this to work, information about the system, as well as good analytic tools is needed to interpret the current state. Using cameras to acquire images over the fields will enable a database to be used in machine learning algorithms that will enable extraction of information over the system.

1.2 Aim

The primary goal of this thesis is to be able to extract and classify different kinds of weeds that are located in a farming field. Given an image of a section of a field, information about the field is in form of location and density of the weeds. This information will be crucial when determining the state of the field and what is

the proper course of action in order to both maximize the yeild and growth of the intended plants.

1.3 Limitations senare

There exists different kinds of limitations within the framework of this project. The first limitations is due to small amount of data. In order to give machine learning algorithms a chance to do its work, a large amount of data is required to make proper assumptions on the dataset. For the initial stages of the project, a dataset of 8 different kinds of weeds are provided with 27 images each. On this dataset, the different algorithms are tested and evaluated.

1.4 Temporär information

Name	Command
Chapter	\chapter{*Chapter name*}
Section	\section{*Section name*}
Subsection	\subsection{*Subsection name*}
Subsubsection	\subsubsection{*Subsubsection name*}
Paragraph	\paragraph{*Paragraph name*}
Subparagraph	\paragraph{*Subparagraph name*}

2

Machine Learning

"Practice makes perfect"

Machine learning is currently a buzzword, as of now there is a lot of hype around it and is maybe used when not necessary. In general, machine learning could be said to be creation of algrotihms which finds patterns in training data, without explicitly told exactly which, in order to make predictions on a new, but similar dataset. In the following chapter, different categories of machine learning tasks will be discussed and some algorithms for these problems. In the end of the chapter, some algorithms which goes outside the scope of this thesis application is also discussed.

2.1 Machine Learning

As described above, applied machine learning uses an algorithm, which is trained on some data and the result is applied to other data. The machine learning can be divided into different classes depending on how the format on the data, and how the training is on the algorithm is done.

2.1.1 Supervised Learning

The intended outcome of a supervised learning algorithm is often very clear. As the name suggsets, the training process could be said to be observed by a teacher, which rewards the algorithms based on a payoff. If the algorithms behaves correctly for some training instance it will be encouraged to do the same for similar data, but if the algorithms makes misstakes it is penalized and should change its behaviour. Examples of these kinds of systems are classification and regression, where the answers are known during the training process, e.g. a system which should learn to distinguish images of cats and dogs, the supervising teacher knows the correct class of the training image and tells the algorithms whether it labels the image correctly or not.

2.1.2 Unsupervised Learning

As opposed to supervised learning, in unsupervised learning the algorithms is supposed to find its own answers. This might sound strange but it can be used to find structure in data without knowing what to look for. This is useful for finding hidden patterns and can be used in a classification sequence preceding a supervised learning part. These kinds of learning methods can be used in e.g. clustering or feature learning.

2.1.3 Reinforcement Learning

Reinforced machine learning problems are quite different from un-/supervised learning problems, as the output of an input is often not evaluatable. To give an example of such a system could be a robot which is supposed to navigate in its surounding with a goal of moving from point A to point B. THe inpouts of the algorithm could be different kinds of sensor values and the output is the soutgoing signal to its actuators. There is not a obvious output at each pass of sensor values to the actuators, since there are an infinite number of ways to get between two points. The payoff of the algorithms should therefore be based on how well the entire task is performed. Another example is a chess playing algorithms, based on the state of the board there might be an optimal move for that exact setup, but a good chess player needs to be able to consider several succesive moves at each state. Thus, the algorithm should be evaluated based on whether it won the game or not instead on its individual moves.

2.1.4 Deep Learning

Deep learning is techniqually a subset of the other learning techniques, but what is special about it is the complexity of the model. It is usually a combination of several un-/supervised learning algorithms layered on top of each other, making the learning process learn in many steps, thus deep learning. The advantages of such an algorithms is its ability to learn complex and abstract patterns, but the disadvantage is that is requires an enormous amount of data. E.g. in a image classifier, it can learn what parts of an image is relevant an require less preprocessing than other models.

2.2 Unsupervised Learning

In unsupervied learning we want to find order in the data we are studying. This kind of learning is often called clustering which will be apparent when describing the following algorithm.

2.2.1 k-means clustering

If one would plot the data in a graph, there would probably be some regions that are more dense than others. A clustering algorithm would try to group the data in the regions together without knowing if they are related or not, thus unsupervised learning. A simple and effective algorithm to group the data in k clusters is to select k means which represents the regions. A data point belongs to group c if the cluster mean m_c is the closest one, using euclidead distance, $d_i^2 = \sum_j (x_j - m_{i,j})^2$. This is called the k-means clustering.

The algorithm is training by updating these centroids m_i until the algorithm has converged. The clusters are initialized by assigning k training data points as the

different centroids. The distance for each training points are calculated to the centroids and the current group S_i which is closest to the centroid m_i is assigned to the points. The centroids are updated according to,

$$m_i = \frac{1}{\#S_i} \sum_{x_j \in S_i} x_j, \tag{2.1}$$

where $\#S_i$ is the number of training points in group i. This is a great algorithm for grouping clusters that are linearlty separable. What this mean is that the clusters are separated by a hyperplane, but this is what limits this algorithm, if the data is better separable by, say a circle, then this clearly fails. One way to tackle this, while keeping the founding algorithm, is by introducing another distance measure. The change of distance functions, also called the kernel trick, could be introduced here, but it still poses another problem, namely the choise of distance measure. The selection of parameters chosen in a machine learning algorithm is a large factor of its performance, and the choise of the distance function here is dependant on the way the data is clustered. If we do not really know how the data is clustered, then there is no real way to define a distance function to separate them, instead we will take the approach of applying non-linear transformations on the data and separate the transformed data, together with the orignal, linearly. This is generally not how you would like to thing as you are basically introducing redundancies, but hopefully making the data more complex gives us the opportunity to retain the simpler model.

2.2.2 Ev mixture models

2.3 Supervised Learning for multiclass classification

2.3.1 Quadratic Discriminant Analysis

Our first multiclass classification will be the so called Quadratic Discriminant. For this classifier we will model our classes with a class conditional distribution, $P(X|K = k)$. This distribution tells us what is the probability of seeing the studied data X given that we are in the class k. We will also assume that the features from plants in a class vary as a multvariate Gaussian distribution, i.e.

$$P(X|C = c) = \frac{1}{(2\pi)^D |\Sigma_c|^{1/2}} exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_c)\right), \tag{2.2}$$

where $\boldsymbol{x} = x_1, \ldots, x_D$ are the features of the studied plant X, and $\boldsymbol{\mu}_c = \{\mu_{c,1}, \ldots, \mu_{c,D}\}$

$$\Sigma_c = \begin{bmatrix} \sigma_{c,11} & \sigma_{c,12} & \sigma_{c,13} & \ldots & \sigma_{c,1D} \\ \sigma_{c,21} & \sigma_{c,22} & \sigma_{c,23} & \ldots & \sigma_{c,2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{c,D1} & \sigma_{c,D2} & \sigma_{c,D3} & \ldots & \sigma_{c,DD} \end{bmatrix}$$

,

are the mean and covariance of the feature distribution of plant c and D is the number of features that describes the plant. $\sigma_{c,ab}$ is calculated as $E[X_a X_b] - \mu_a \mu_b$. To get the reversed conditional probability, i.e. the probability for beloning to class c given the input x we use bayes theorem,

$$P(C = c|X) = \frac{P(C = c)P(X|C = c)}{\sum_i P(C = i)P(P(X|C = i))}. \tag{2.3}$$

The denominator will be omitted as it occurs for all classes and we are only interested in the relative probabilities between these. We will rank each class after taking the logarithm of each conditional probability and end up with the expression,

$$R_c = -\frac{1}{2} ln|\Sigma_c| - \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_c) + \ln P(C = c), \tag{2.4}$$

and the selected class will be the one with the largest rank.

2.4 Feed forward nerual networks

Something svm requirements. Here we will discuss a method to let the model determine its parameters and decision boundary using a combination of a simple activation function.

2.4.1 Logistic regression

2.5 linear regression

Model class $y = f(x; \omega)$, where f uses numerical parameters ω to map input vector x to predicted output y. During the learning, reduce the discrepency of target t to y.
for regression, minimize a energy function $\frac{1}{2}(y - t)^2$, the closer the output y is to the target, the smaller the energy. Classification problems, where the output is a descreet number curresponding to the class, can be translated from a regression model, where the selected class is the closest class from the continous output. This makes classification algorithms easier to train as we have a continous space and the involving functions are easier to differentiate.

2.5.0.1 Linear boundary

Imageine that we want to sort a data set of two features that can be separated into two groups using a line. The coefficients for this line are $\boldsymbol{\omega}^T = [\omega_0, \omega_1, \omega_2]$ and to determine if a data points belongs to either groups, we simply determine the sign of the line parameters multiplied with the feature vector,

$$y = \boldsymbol{\omega}^T \boldsymbol{x} = \omega_0 + \omega_1 x_1 + \omega_2 x_2 \begin{cases} Group\,1 & if \quad \geq 0 \\ Group\,2 & if \quad < 0 \end{cases}. \tag{2.5}$$

If we we denote group 1 as 1 and group 2 as 0 we can use a step function,

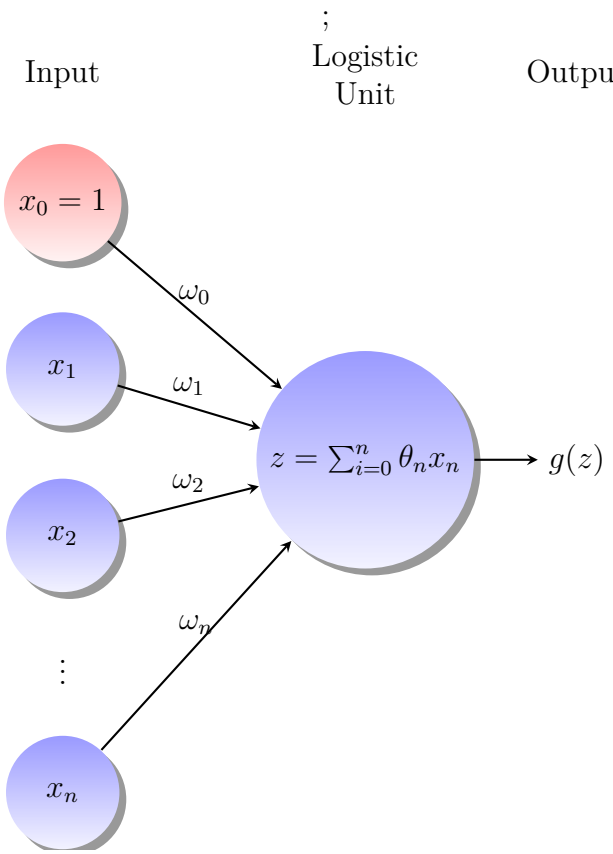


Figure 2.1: A logistic unit combines the inputs via the weights on the connection from the input to the unit. These weighted inputs are summed and are inserted to the activation function whose result is the output.

$$y = H(\boldsymbol{\omega}^T \boldsymbol{x}) \; where H(z) = \begin{cases} 1 & if \quad z \geq 0 \\ 0 & if \quad x < 0 \end{cases} \tag{2.6}$$

so to make the model make predictions on this data set we only need to set the parameters of the line.

More on the activation function

To train this model we will be in need of a continous function as we are going to use its derivative to optimize its parameters. The function we will choose will have to be symmetric around $(0, g(0) = 1/2)$ so the model is not biased to one class or the other, and also in the range $[0, 1]$. A good candidate is the logistic function,

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.7}$$

which is the function that gave this model its name. One way to visualize this model is to use one so called logistic unit and its representation can be seen in Figure. 4.1, where we have generalized the model to use n inputs. The logistic unit outputs a value depending on the inputs its function it is called an activation function, since a low value means lw activity and the opposite for large values.

Unlike the discriminant classifier, we will use our data to update the parameters of the model rather than creating them. Instead the parameters will initially be randomsized and updated to better fit the model to the data. The model will be evaluated using a cost function which penalizes the model for making the wrong predictions. This cost function increases with increasing errors and the objective is to select parameters which minimizes this function.

2.5.0.2 Gradient descent

One of the simplest methods to minimize a function $f(\boldsymbol{x})$ is to always change the parameters in a neighbourhood which decreases the function the most. The direction that decreses the function the fastest is in the opposite direction of the gradient, i.e. $-\nabla f(\boldsymbol{x})$. So if we continously let the parameters update towards the negative gradient it should eventually find a local minima. One requirement for this is that we are using a so called learning rate, α, which is sufficiently small, if it is not then the parameters might be updated to much and 'overshoot' the minimum and start increasining again. The final form of the algorithm is,

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \alpha \nabla f(\boldsymbol{x}_n), \tag{2.8}$$

until convergence.

2.5.1 Multiclass Optimization

mclasses Until now, we have only discussed the logistic regression for comparing two classes. To generalize the model we will use a method called one-vs-all. As the name suggest we will, in turn, compare one class vs the rest, which is then done for all the different classes. The class which gets the largest rank from the activation function is then chosen as the correct one, and this is one other reason why we want a continous activation function, as we can compare how well the data fits the class by the result of the function. This means we are basically training n different models and we can model this using several logistic units (Figure. 2.2), this is actually what we call a single layer neural networks, but more on this later.

2.5.2 Non-linear Boundary

Previously we have only discussed how to optimize lines separating m classes, but we would like to be able to create more complex boundries using non-linear functions. One way to create more complex boundries would be to include quadratic terms in the actication function, i.e. $z_m = \sum_i^n \omega_{m,i} x_i + \sum_i^n \sum_j^n \omega_{m,i,j} x_i x_j$. We can train such a model in the same way as the linear activation parameter function, but the limitation still lies in the choise of parameterization function, one might not be satisfied with the quadratic term and would like to include higher order terms. This sounds like a tedious task, and an algorithm which creates this parameterization function automatically seems like something to strive for.

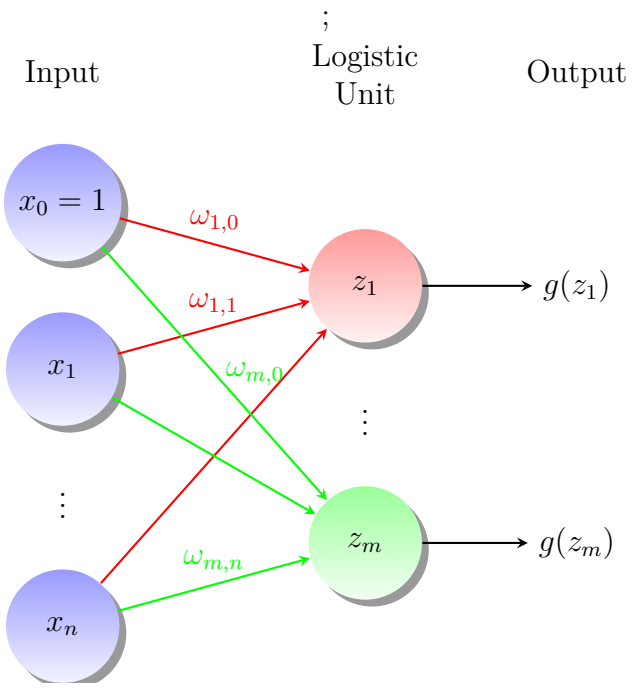


Figure 2.2: Instead of only being connected to one logistic unit, the inputs are connected to all m units representing each class. Each logistic unit j are connected to the inputs by the weights ω_j and after the summation, the unit with the largest activation "wins".

2.5.3 feed forward neural networks

What we have done with the logistic regression is that we have tried to approximate a function which is able to separate the data from the different classes. We could actually first train a logistic regression model to approximate a function which we then use for training another logistic regression model which then seperates the data. This means we could use the first model to get a parameterization function which is used for the activation of the next one. So what we can do is create a network of regression models as in Figure. **??**.
Using one layer of regression sparates the data using straight lines, but as this could then be used as input to another layer it can be shown that using two layers can separate data using in a convex set. So using only a two layered network of logistic units we can separate data using complex boundires, but this is still not as general as we would want, we might want non convex sets to separate data. It can be shown

Reference here

that using a two layered network can approximate any function with arbritary accuracy. This is what we wanted to begin with, to approximate a function which is the activation function to a logistic regression model. Thus, train a model which approximates a activation function using a two layered neural network connected to yet another layer can create general decision boundires. So using a three layered neural network can separate data and is thus sufficiently deep for most machine learning tasks, but there is actually reasons why one would use other number of layers which we will discuss later.

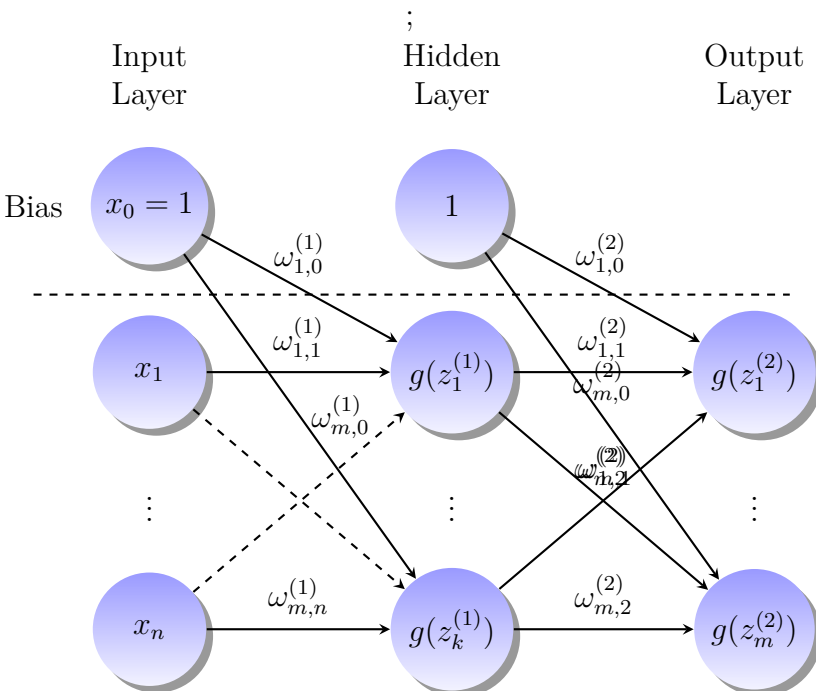


Figure 2.3: Instead of only being connected to one logistic unit, the inputs are connected to all m units representing each class. Each logistic unit j are connected to the inputs by the weights ω_j and after the summation, the unit with the largest activation "wins".

2.5.4 backpropagation

We have yet to talk about how to these layers of logistic units or, as we will call it from now on, the neural network is trained. One could train one layer at the time, first train the first layer to approximate a linear separating function, which is fed to the layer which uses this function to approximate an arbitrary function, that is then used for the last layer. This process does not really work, since we do not know from the beginning which function we want to approximate, and if we did, we would not need this method at all. So we would need a method to update the layers simuntanously, depending on how the changes affect the other networks. What we really need to do is see how well the last layer classify the data, compute the result using an error or cost function and update this layer using gradient descent as before. The difference is that we want this update to go back to the previous layer as well, so we somehow get the error gradient for those units. They can then be updated also using the gradient descent, and send information backwards to the previous layer. This process continous until we have updated the input layer. The method that does this is called backpropagation, or error backpropagation, and it is not hard to imagine where it got its name, since the error is propagated through the network starting at the output, ending at the input.

equations

2.6 Preprocessing

Now we have equipped us with the tools necessary to classify data to different groups, but we have a large part still before we can start applying machine learning to a real example. When we recieve data from some sensor, e.g. a camera, it is usually not in a format which we want to classify, we have to preprocess the data to fit our needs. Almost always we have to clean the data, any equipment will produce some kinds of noise, and depending on the amount of data we have for the training, this could be a huge problem as the model might learn the noise and not the actual trend behind it.

Another preprocessing step could be to extract informative features, this could be features that we know are a part of distinguishing between different classes. Different features could be such as sizes or shapes of objects. This part of the preprocessing is called dimension reduction, and this part is used to try to separate only the essential information about the different classes. The preprocessing steps used in this thesis will be explained in the following chapters, and how we combine these will be discussed in Chapter. **??**.

2.7 Convolutional neural networks

The amount of preprocessing required is decreased with the amount of data availiable. If one would have a plethora of data one could teach a model what features to extract. This could be done using an unsupervised learning algorithm prior to the supervised one, or one can combine these to one large supervised learning algorithm. This is one of the reasons why one would considera a deeper network than a three layered neural networks. It is at this transition one start to usewhat is called deep learning, when the network, not only classifies the data, but also learn what features to consider for the classification. Using larger and larger networks increases the training time as the backpropagation needs to go through many layers, and it will take many more iterations to learn the features for the later parts of the networks. But given enough data, the network can learn effective features, albeit very abstract ones, that can't be explained in words.

For certain problems, there are reasons to use special kinds of arcitecthures of the network (the network above is called a fully-connected feed forward neural networks, due to that each layer is completely connected to the previous one and the data always goes from one layer to the next). E.g. when working with sequiental data there could be a reason to keep some information from the previous time step. Instead of increasing the number of inputs to include all of the previous time step, or even the one before it and further, one could keep the activations of the units from the previous timestep as an input to the unit as well as the new data. This kind of network is called a Recurrent Nerual Network (RNN), this kind of network works well when the input data has some temporal dimensions, such as music or stock market. Though, this thesis involvs information in images, which requires another type of network, namely Convolutional Neural Networks (CNN). A convolution is a

way to reduce information in a neighburhood of a pixel to a single number, so it is a way dimension reduction and the mathematics behind convoltuions can be found in Appendix. **??**. A convolution on an image could extract information about for example large changes in the images such as separation between different objects. So a convolutional neural networks begins with several layers of convolutions to learn special features in images, which is then followed by a normal fully-connected nerual network to separate the learnt features. The downside of this network is it requires a large quantity of quality images, atleast of the size ≈ 10000 images for each class, which is many times more images than availiable for this project, though, te use of this could be the future of similar projects.

3

Computer Vision

"A picture is worth a thousand words"

B4 machine learning, need to clean data. How extensive depends on the machine learning algorithm and the data. Often some kind of noise removal.

- Basic information in images
- Noise removal, e.g. smooting (introduce filters)
- Image segmentation (color represenations)
- image processing, (morphological operations)

Before considering using any machine learning algorithms for classification on images, there is often a need for some image preprocessing, except in some cases which will be discussed later in Chapter.**??**. In order to give the algorithm as informative data as possible we want to extract the relevant parts of an image. To do this we need to consider what kind of information is present in images and how to use it.

3.1 Images

Images is a way to display large amount of data visually. An image is often represented using a one or many 2-dimensional matrices. The building stone of an image is a pixel, which is a vector of size, d, where d is the number of matrices used by the image or also called *channels*. The values of the pixels are usually called the *intensity* of the corresponding channel and are limited to a certain set of values, which could be either continous or discreet. The channels are usually represented using different colors and more often then not ny red, green and blue colors, more on this can be read in Appendix. C.

3.1.1 Information in images

The number of channels used by an image is called the *depth* of the image, and is as mentioned above, often 3. It is also mentioned that the intensities in these channels are limited to a set and most file formats uses 8 bits to express these values, meaning each pixel contains 24 bits.[1] This means that each pixel can contain any of $(2^8)^3 = 16777216$ different combinations of colors. The dimensions of the matrices of the image is usually of the order 10^3, meaning that there are a humongous different

[1] Sometimes a fourth channel, the *alpha* channel, is present in an image and thus a pixel contains 32 bits of information, but these extra bits do not carry any color information and will be omitted in this thesis.

combinations of images. Surely there must be a way to classify the message of an image by the use of differnet layers of information.

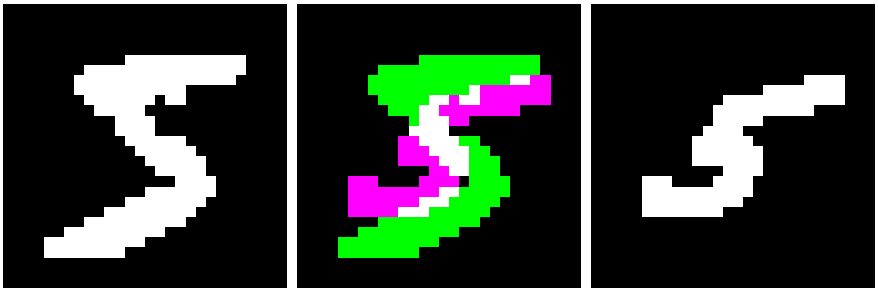


Figure 3.1: Above we see two binary images of size 28×28 that represents the digit 5. Individually there is no ambiguity of their illustrations, but studying their overlap in the middle picture they might as well come from different classes as the conjoined sections are far smaller than the secluded parts.

A image with a message does not carry pixels independant of each other, in Cref-pic:overlap we can see two different images both containing enough information to convey a digit 5. So the images must contain more information than what is given by the individual pixels, the position of these must also play a role. Therefore we need to distinguish different parts of the image before we can be sure of its representaion. These parts will be divided into three categories,

- **Global**: all pixels must be considered simuntanously to carry a message.
- **Regional**: Only a subsection of the image is important for classification.
- **Local**: The individual pixel information is important.

and a combination of the regional and local parts will often be used. As we saw earlier in the example with the digits, the regional of the white pixels were the important ones to see what the image showed and its regional and not global, since a linear transformation of the white pixels would not change the characteristics.

3.2 Image nosie removal

gonzales 10.3.4

3.3 Image segmentation

In order to extract useful information from an image, we need to segment the image into the parts we want to extract information for, e.g. if we want to study a plant in a field we need to remove the background dirt.

3.3.1 Histogram-based thresholding

If the part of the image is fundamentally different from the rest of the image, e.g. a white cloud on a blue sky, we can use a technique of thresholding. In thresholding, we represent the image using only one intensity, i.e. only one channel, and creates a

threshold value, T_I. Using this threshold we create a binary image where 1 represents the distinct part of the image and 0 the indistinct part of the image using,

$$I_S(x,y) = \begin{cases} 1 & I(x,y) \geq T_I \\ 0 & I(x,y) < T_I \end{cases} \tag{3.1}$$

Choosing this threshold might be trivial for some images, e.g. images which have a bright and a dim region, but sometimes the intensities are very close or even overlapping. So in order to make the different part easily separable one would need to represent the image intensities using a non-trivial transformation from the different inputs. E.g. separating a green plant from a brown background could use the color information to separate these, but both parts might have some overlapping of the color channels. Instead we can transform the three channels, Red, Green, and Blue to another index which separates the color information, which is done by converting the RGB color to HSI color space. Details of this transformation can be found in Appendix. **??**. The Hue index of this color space carries is the only index carrying information of which color it represents, which makes it an excelent choise for separating objects of different colors. To select the threshold we will then use a method called adaptive global theshold. The histogram of the intensities is first plotted, which should show two decently separated sections and a value between the two regions peaks should be selected as an initial estimate of the thresold.
Using Equation. (3.1) we will get two different groups and the mean values of the intensities in respective group should be calculated. The threshold is then updated to be the mean of the means, which will procedurally produce better and better estimates for the separating threshold, and the procedure will stop at convergence or when the threshold updates less than a targeted change δT. An example can be seen in Figure. 3.2.

3.4 Image processing

More often than not, the image thresholding is not enough to completely separate the object from the background. Often there is other objects that are of similar color that came as a side product of the thresholding and also parts of the object might not perfectly separated. This part of the chapter will deal with the post processing of the image thresholding. Here we will use tools from a mathematical concept called morphological operations, where the details can be found in Appendix. **??**. These operations will also be useful in the next chapter where we will extract useful information of the object obtained after this post-process.

3.4.1 Separated parts connection

The first part of the post-processing will be to connect separated parts of the object, and the concept is to make all the objects found using the thresolding slightly larger, after which the parts of the desired should be connected. We apply the morphological operation, dilation, to the previously aquired binary image using a $3x3$ structure element,

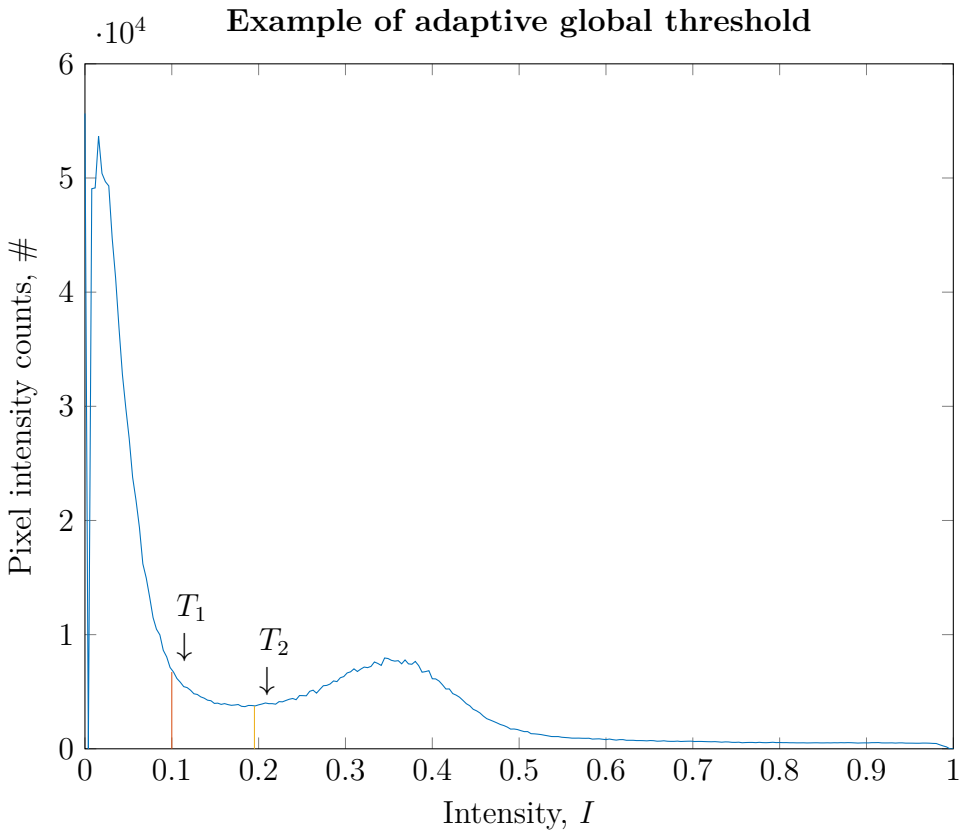


Figure 3.2: The initial threshold guess, T_1, is too far to the left which is adjusted to T_2 after one iteration of the adaptive global thresholding algorithm.

$$SE = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.2}$$

bild på hur det kan gå till

3.4.2 Connected components

When we have connected the different parts of the object we want to extract the object by itself. We will do this by using the so called connected components, the general idea is to find different parts of the binary image which are connected with each other, and then the component which occupies the largest portion of the image should be our desired object. The idea is to search through the whole image, checking whether two neighbouring pixels belong to the foreground, and then set them as the same connected component. The binary image should then contain different clusters belonging to different groups.

bild som visar olika färger på olika grupper

3.4.3 Hole filling

We started this post-processing by connecting separated parts of the object, and then we extracted the object from other objects, but the result might still not represent the whole object as there might be some small sections that was not captured by the dilation process. WE would like to will these holes to get a complete solid object. THis procedure is quite simple, we start by inverting the image, i.e. the white pixels are now black and vise versa. we then extract the connected compoenents. All the holes in the object should now be in different groups, and since we know that all the holes are inside the object, none of them are at the border of the image. So to remove all the connected components which does not represent the holes have pixels at the boundries of the image. Removing these will make sure only the holes are remaining, and to get the final result we add the images together.

ännu en bild

3.4.4 object shrinkage

We started this process by making the object slightly larger by the dilation operation, and to finalize it we will reverse this operation to remove the border that does not belomng to the object. So as the last part of the post-processing part is to apply the opposite of the dilation, namely the erosion, using the same structure element from (3.2).

4

Information Extraction

"Ball is in your court"

As discussed in section. **??** a large part of machine learning is process incomming data to a fromat which is suited for learning. In this chapter we will extract features which gives descriptions to different objects. This is the second part of the preprocessing following image preprocessing, in which it is assumed that the object at hand has been completely segmented from the background and a binary image has been produced.

4.1 Descriptive Object Features

4.1.1 Size dependent features

One great start when classifying objects is to estimate its size, if the object is roughly the same size as a house then a classifier that is not proficient in differentiating bushes from trees, then the size could be the deciding factor. To use size as a feature, there is a need to calibrate lengths in an image, since we need to know what scale we are using. If a proper scale is not know then these features do not give any relevant information, but the algorithms for calculating different size features can be combined to create size independent features such as shape, which will be seen later.

4.1.1.1 Area

The area of the object is the most straight forward feature, since we only have to count the number of pixels occupied by the object times the area of a single pixel. Unless there is a calibrating length in the image there is no way to accuratelly calculate the area of the pixels, but if the images are taken from the same angle and height we can set it to one.

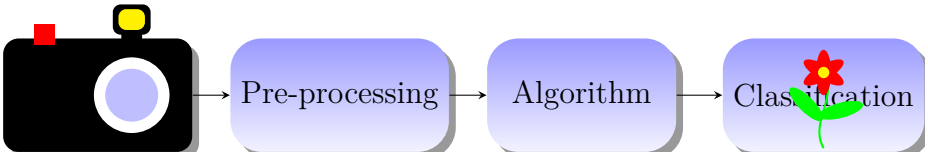


Figure 4.1: A logistic unit combines the inputs via the weights on the connection from the input to the unit. These weighted inputs are summed and are inserted to the activation function whose result is the output.

4.1.1.2 Area Perimeter

The area of the object only determines how much space it occupies, but it does not contain any information about its shape, so to include some spatially aware features, the perimeter is included. The perimeter is simply the number of pixels that is in the set of the object pixels and also in a neighbourhood of the background.

4.1.1.3 Convex HUll

Another feature that determines the shape of the object is the convex hull, which is the smallest convex set of pixels that includes the entire object. A convex set is a set where a line between each set of elements is included in the set. So the convex hull is the same as the area plus the background "valleys" of the object.

4.1.1.4 Convex Hull Perimeter

Again, we include the perimeter of the previous feature. Since the convex hull fills the valleys of the object, the convex hull perimeter is always at maximum as large as the area perimeter

4.1.1.5 Thickness

Lastly, we have a feature called thickness. The thickness is the number of times we have to apply the operation thinning described in appendix. **??** until convergence.

4.1.2 Size independent features

4.1.2.1 Mean Color

Since images are represented using the three color channels, red, green, and blue, this feature is actually three. The mean of each channel is simply all the object pixels color value summed, and divided by the number of pixel, or here even by the area.

4.1.2.2 Standard devisation color

The standard deviation of the colors can also be included as some objects might have wider color ranges.

4.1.2.3 formfactor

$$elog = 4\pi \frac{area}{perim^2} \tag{4.1}$$

4.1.2.4 Elognatedness

$$elog = \frac{area}{thickness^2} \tag{4.2}$$

4.1.2.5 convexities

$$convexity = \frac{convHullPerim}{Perim} \tag{4.3}$$

4.1.2.6 solidities

$$solidities = \frac{area}{convexity} \tag{4.4}$$

5

Methods

"Actions speak louder than words"

5.1 Data aquisition

- Data
  - Aquisition
  - Instruments
  - Preprocessing
- Methods
  - Feature selection
  - Algorithm evaluation

This project classifies data on two different scales, one that is quantitative and the other qualitative. These different datasets are labeled differently and will be processed in different manners. The qualitative data set has been provided as a test set from a previous masters thesis by mAndersson [**m_nadersson**]. This set has been completely labeled and the images of the different plants are of the highest quality, containing a small amount of noise. It is on this set the supervised learning will be applied as the whole process from pre-processing to features is streamlined.
h
First part of the project is data acquisition, and currently there are two different desirable approaches, both with different advantages and disadvantages.

One of the approaches would be to grow the crops and weeds oneself in a controlled environment, alongside with a "wild" grown field of unspecified weeds directly taken from a real field. This approach would give me complete control over the data, when it should be extracted but would yield a small database as I don't have neither the time nor the place nor knowledge go grow an entire field.

The other would be to get in contact with a real farmer, taking continuous pictures from a real field. This could give direct feedback of the state of the field, as expert knowledge is in the vicinity. Although, having a "real life" dataset might seems desirable, I might have too little control of the environment, e.g. the farmer might be busy, the fields might be under some treatment which will alter the plants, and there might be some other uncontrollable factors.

Regardless of the approach, the database will be obtained from the fields using a Canon S110 camera, which can give pictures in $4000 \times 3000 pixels$. To get as good pictures as possible, the camera will be mounted on a custom made tripod, which will give pictures from a desirable height.

When the dataset has been specified, a sequence of images over the same area for different times will be overlapped so a time-lapse of the plants will be available. This is a non trivial task if done autonomously as is desirable since the project should be able to scale. The method that will be used for this process is image warping and matching. Possible problems that might occur is that the images are taken to far from each other in time that the images are too different, and some preprocessing might be used if this is the case.

The next part is image segmentation in order to extract the parts of the pictures which represents the different plants. The first part of this process is to take into account of the color channels in the image, since the plants usually have a distinct green color against the background soil which is usually in a brownish color which consist of a heavy redness. Then different edge detection methods will be used, such as applying a laplacian filter, in order to differentiate plants that resides close to each other.

When the position of each plant is determined, it is time to perform two classification problems on the dataset. The first is a binary classification, is the plant a crop or a weed, and the second is only performed on the weeds given from the first classification. During this part the kind of weed should be determined. Given the distinct features in each of these classes different methods will be considered. If the features is easily distinguished between the classes, e.g. shape and color, then classification methods such as linear or quadratic discrimination will be considered as well as support vector machines. If the classification part seems to be more complex, a machine learning approach be more appropriate, such as a convolutional neural network. This approach is actually preferable as it is easier to scale to include more classes in the future, but I might be limited by the amount of data accessible, since this approach requires a large amount of data.

5.2 Instruments

Basically, the only instruments used for this project is the cameras which the iamges was aquired with. The qualitative test set was provided from another master thises[**mansersson**], and the iamges was taken with a CANNON SOMETHING. The quantitatve data set was aquired using CANON ANNAN camera with a near infrared channel instead of the red.

5.3 Preprocessing

From the two different cameras we have two different types of images. The first one from the previous master's thesis is a very clean data set. It does not contain much noise from other object than those being

5.4 Algorithm evaluation

5.5 Feature selection

In the previous chapter. **??** we introduced a plethora of features for object recognition, but not all of these are relevant for the classification or might even be worsening the results. There are several ways to obtain a good combination of features using different techniques.

5.5.1 Forward selection

Forward selection is a very straight forward technique of choosing features, it does not retrieve the best combination but it sacrifices the absolut best for speed. One starts with the single feature which gives best result to the algorithm and then combining this feature with every other feature. The combination which yields the best result is selected for the next pass of the selection. The selection continues by testing the previous combination with every non-selected feature, progressively combining more features and decreasing error-rates. The number of features is then selected when the error-rates start to increase again or after a set number of features.

5.5.2 Backward selection

Backward is as the name suggest the same as forward selection but in reverse. Instead of starting with one feature and add one each pass, we start with all features and remove the one that changes the error-rate the most. This method works when using all features provides superflous information and only provides negative impact on the classification.

5.5.3 Best combination

Instead of progressively add/subtract one feature, the best combination tests all possible combination of features at each number of features. This method ensures to find the absolutely best combination as it testa all cases, but the down side is that is very computationally expensive.

5.5.4 Principal component analysis

det som ska användas till neurala nätverk

5.6 Algorithm Evaluation

To make sure an algorithm is mature enough for the real world we need to know how well it performs on data it has not seen before, that is how well it has learned the underlying structure of the data. This can not be done without extra data as presenting it with the same data as it has been trained for is superflous, it is like studiying for an exam by reviewing only previous tests without considering the background theories. In order to simulate this real world senario we will only show the model a certain set of the data and only after the training is done it will see the rest. This makes sure the model can be evaluated on data is has not been trained on and also gives us a way to measure how well it has generalised the problem.

5.6.1 Training and validation sets

For the quadratic discriminant classifier, the parameters for the model is directly related to the data used for training it. It would not be possible how well the model has generalised the data using the availiable data. To get an estimate how well the problem has been modeled it is common to separate the training data in a training and a validation set, where the training set is only used for the training process and the validation set is used to validate the current training. This ensures the model is tested on new data. One problem, though, is when the model is dependent on hyperparameters such in the case of the nerual networks with its activation function, network types and number of layers. If one would separate the availiable data in training and validation sets, and then select the hyperparameters which gives the best result for the validation set, one has basically overfitted the problem to match the validation set.

5.6.1.1 Training, validation, and test sets

The most common way to evaluate the performance of an nerual network is to divide the data set in not only two but three different sets; namely training, validation and test sets. The network is trained on the training set for a number of epochs, and then evaluated on the validation set. This continues as long as the validation error continues to decrese, but can start increase again while the training error is sill declining. The reason for this is that the model has stopped learning the general features of the data sets and start to overfit the training data. This gives a well defined stop criterion for the training of the network, and the epoch of the network which minimizes the validation set for that specific set of hyperparameters is then tested on the test set. Choosing the epoch which minimizes the validation set and choosing the hyperparameters which minimimizes the test set gives ensures that the model has neither overfitted the training data nor overfitted the hyperparameters for the validation.

Although this is the most common way to validate nerual networks, it is not the way we will validate the networks in this project. The reason is that we have to few data points to consider in the first place and to compare the nerual networks to the quadratic discriminant using different evaluation methods does not seem inviting.

There is no reason to train the quadratic discriminant in this way as there are no hyperparameters in that model.

5.6.1.2 K-fold cross validation

Instead of dividing the data in three sets, we will keep the training and validation sets and use a method called, K-fold cross validation. This method is good for evaluating a model when there are a small number of training data or many features compared with the data. To use this method one would randomize the order of all the availiable data and assign them in K different groups of the same size. The chosen model is then trained on $K-1$ of these folds and then evaluated on the last. This is repeated K times, so that each fold has been the validation set once. The total error of the training is then the average over the K different combinations. This process can be repeated over different hyperparameters to ensure those parameters model the entire data set and not just one combination of validation set.

5.6.1.2.1 Visual example of 3-fold cross validation To give the reader a better understanding of the method, a visual example will be given here using $K = 3$. The data is divided into three groups as seen in figure. 5.1. Each data point is then assigned a new random group and then the data is grouped together as in figure. 5.2.

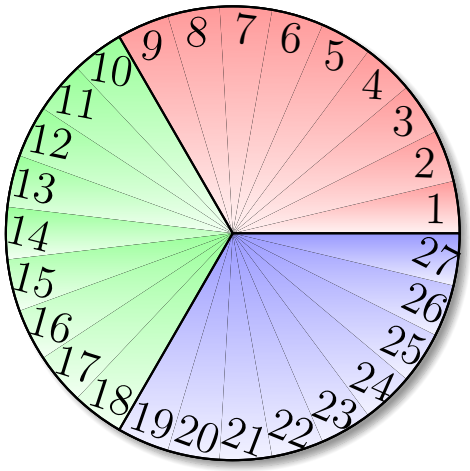


Figure 5.1: The simplest way to divide the data into several groups is just to select the groups in order.

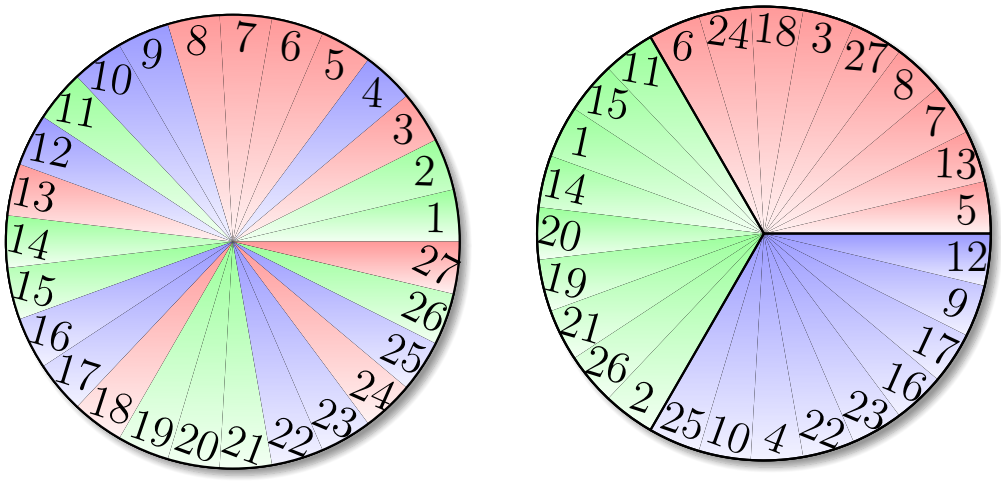


Figure 5.2: To make the grouping stochastic, each number if assigned a new random group, while still ensuring the group sizes remain the same. The new groups can then be placed together for easy access.

6

Results

"Cut to the chase"

6.1 Image segmentation

6.1.1 Data set 1

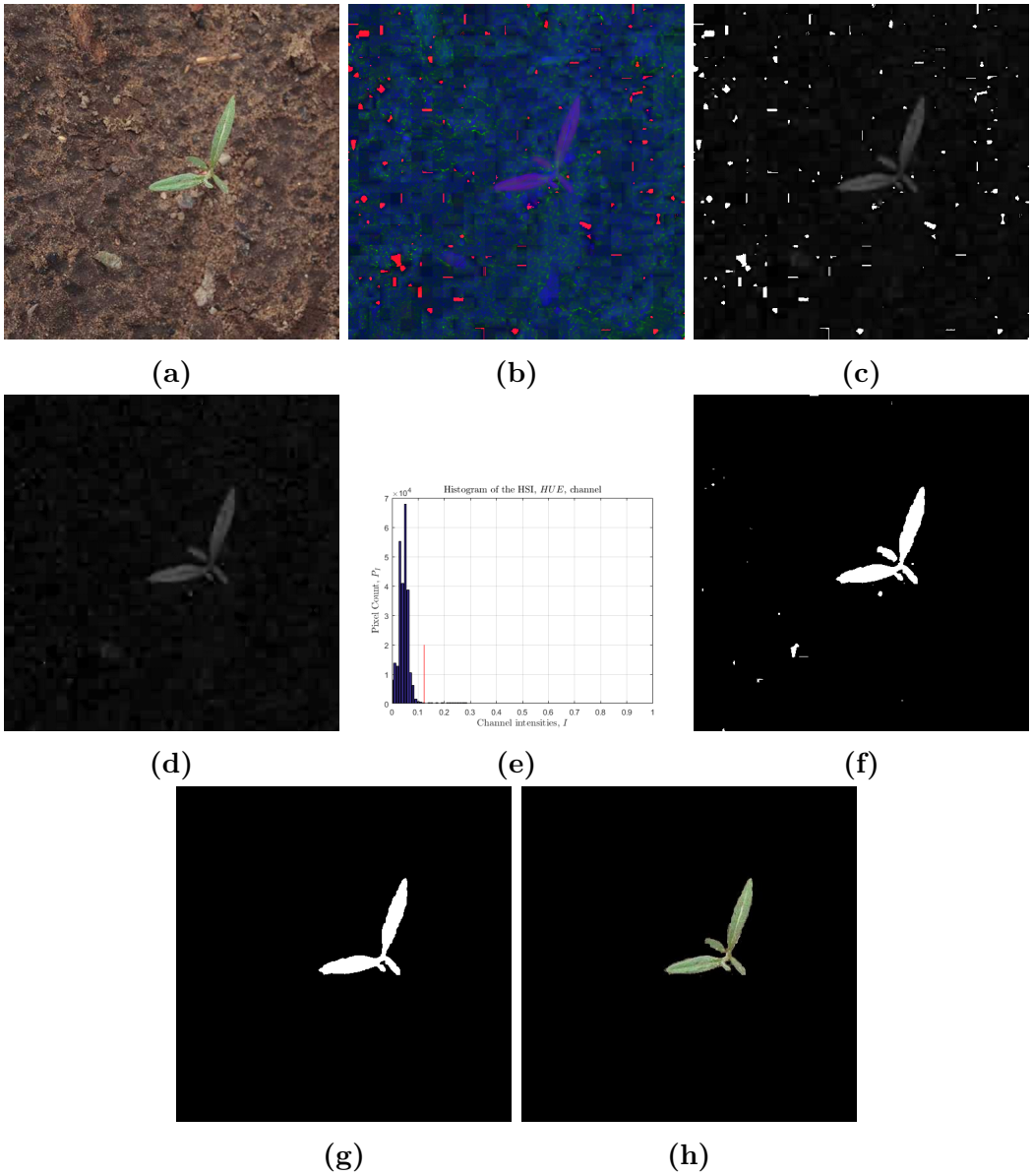


Figure 6.1: The segmentation process as follows: *(a):* The original image is loaded and prepared for processing. *(b):* The image converted to HSI but still represented in RGB colours. *(c):* For the segmentation to work properly the image needs to be represented in one channel that represents the two segments well, and this is what the Hue channel in HSI representation does. *(d):* The Hue channel clean up. *(e):* The optimal threshold between the two segments is represented by the red line *(f):* Binarization of the HSI image using clipping. *(g):* Clean up the binary image by removing small objects. *(h):* Final segmented image.

6.1.2 Data set 2

(a) (b)

Figure 6.2: Directly applying the same segmentation algorithm on non-perfect noise data does not really give satisfying results.

Using the same algorithm on the data received on a field without alterations, and also quite late in the development does not give good result as can be seen in figure 6.2. This is due several reason, first we can see that there are more than one interesting plant in the picture, and from the results we see that only a part of a plant is given. This is where the convolutional neural networks come in for the first time, using a convolutional neural network one can train the networks to segment the image into several important parts. Using this network to classify segments of an image could be done as prework and then the normal classification using a normal feed forward network or just the quadratic classifier, but why stop here when we have already integrated a CNN, why not use it in series with another CNN to only train one big network instead of using several different parts of the classification.

6.2 Method

6.3

Återger resultat från observationer och datorstödda analyser/mätningar. Vanligtvis återges resultat i form av figurer, tabeller, grafer, eller fotografier men det är avgörande att även förse dessa med tillräckliga kommentarer för att belysa resultatets värde och eventuella avvikelser eller oväntade resultat. Givetvis ska resultatavsnittet också bestå av text som binder samman illustrationerna. Beroende på experiment eller metod bör skribenter ange och berättiga beräkningar och formler som ligger till grund för tolkningen av resultat och datorstödd analys. I vissa fall måste även primärdata anges för att resultat ska bli meningsfulla och kunna diskuteras. Med primärdata avses obehandlade mätdata. Notera att detta inte är tillämpligt i alla enskilda fall. Fråga din ämneshandledare.

6.4 Results from quadratic discrimination

Table 6.1: Feature names along with their corresponding feature number during feature selection phase.

Feature Number	1	2	3	4	5
Feature Name	Form factor	Elognatedness	Convexities	Solidities	Mean Red
Feature Number	6	7	8	9	10
Feature Name	Mean green	Mean blue	Standard Deviation Red	Standard Deviation Red	Std blue
Feature Number	11	12	13	14	
Feature Name	Area Moment 1	Area Moment 2	Area Moment 3	Perimeter Moment	

Figure 6.3

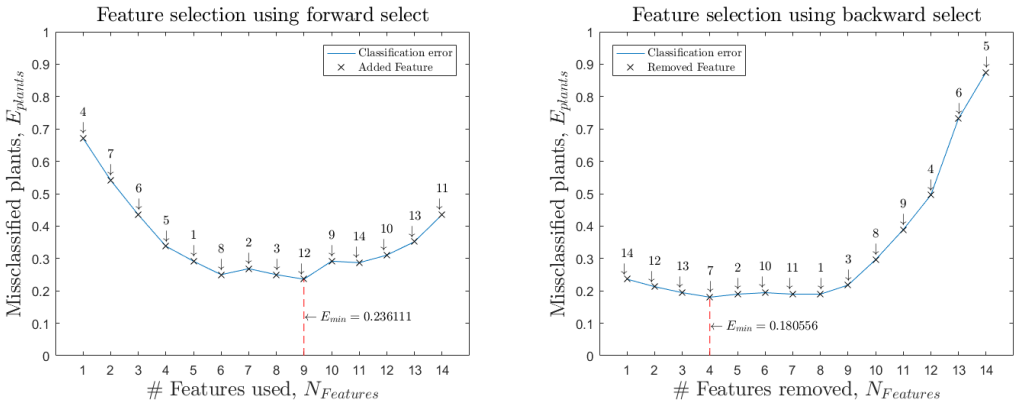


Figure 6.4: Using either forward or backward selection for the best feature combination yields different results. E.g. Feature number 7 (mean blue) is the second feature to be added in the forward selection algorithm and is the last to be removed in the backward selection. The corresponding features can be found in Table 6.1.

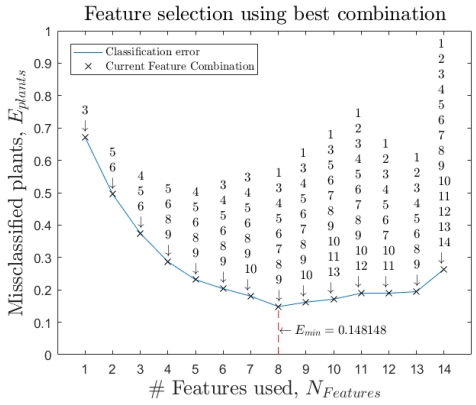


Figure 6.5: With immense computer power the best n combinations can be brute forced. This ensures to get the best available combination using n features as all combinations are compared to each other.

As we can see in Figure 6.4, many of the defined features are present when selecting the best combinations of features using both the backward and forward selection algorithm. In both cases, the class of features which are neglected the most are the moment invariant features. During these algorithms $K-$crossvalidation was used with $K = 3$, i.e. the different plants was randomly divided into 9 different group with 3 plants in each. This grouping is different in each run and might thus yield different results for different runs. To make this more consistent, the grouping could be done using increasing the dart wheel as in Figure 5.1 indexing instead of the random distribution in Figure 5.2.

With much patients the absolute best combination was found by combining the best n features together. This can be seen in Figure 6.5. The result from using this algorithm will only be used in a best case scenario when comparing to other methods later as this algorithm is very slow compared to the other. The feature selection process usually takes place before the classification is done, thought, if one would find a new feature that might give more information this whole process would need to be made again and thus a fast algorithm is preferred.

6.5 Results from using Neural Networks

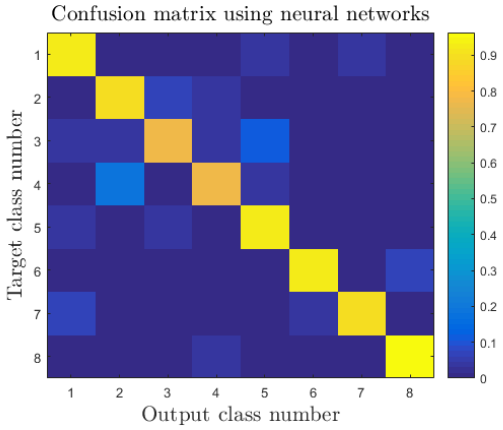


Figure 6.6

7

Discussion

"From a to z"

7.1

Här behandlas resultaten i förhållande till teori och metod och utvärderas med utgångspunkt från syftesformuleringen i inledningen. Diskussionen omfattar även det förväntade resultat som referenslitteraturen (teori, metod, databas) indikerar samt förklarar enskilda specifika resultat och mätdata. Även avvikande resultat och data måste identifieras och förklaras genom synpunkter angående material, teori och utförande. Diskussionen behöver dessutom också anknyta till eller hänvisa till inledningen och problemställningen.

7.1.1

I vissa situationer kan de två funktionerna resultat och diskussion kombineras till ett avsnitt i en rapport. Fråga din ämneshandledare.

8

Conclusion

"To make a long story short"

8.1

Här redovisas experimentets eller projektets slutsatser mot bakgrund av diskussion och resultatredovisning och ofta genom att också anknyta till inledningen av rapporten. För många undersökande projekt är det även vanligt att slutsatser leder fram till rekommendationer. Notera att vissa texter kombinerar avsnitten diskussion och slutsatser. Fråga din ämneshandledare

Bibliography

[1] Rafael C. Gonzalez . Richar... *Digital Image Processing: International Edition*. Pearson, 2009. ISBN: 0132345633 (cit. on p. V).

[2] John Weier and David Herring. *Measuring Vegetation*. 2010. URL: https://earthobservatory.nasa.gov/Features/MeasuringVegetation/measuring_vegetation_2.php (visited on 07/01/2017) (cit. on p. XI).

[3] *Soil-adjusted Vegetation Index*. 2013. URL: http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:soil-adjusted_vegetation_index (visited on 07/01/2017) (cit. on p. XI).

[4] *Modified Soil-adjusted Vegetation Index*. 2012. URL: http://wiki.landscapetoolbox.org/doku.php/remote_sensing_methods:modified_soil-adjusted_vegetation_index (visited on 07/01/2017) (cit. on p. XI).

A

Filters

In this appendix we will introduce image filters which can be explained by the mathematical operation convolution. We will start by explaining what convolutions are and then explain why they are used in image processing by explaining the correlation between an image and its Fourier transform components. Lastly, the discreet version used in images is introduced followed by some important examples and their origins.

A.1 Convolution

A.2 Fourier Transform

A.2.1 Continous Fourier Transform

A.2.2 Discreet Fourier Transform

A.2.3 Implications of Convolution Theorem

A.3 Important Filters

A.3.1 Average

A.3.2 Median

A.3.3 Edge Detection

B

Morphological Operations

B.1 Important section

C

Color representation

This part extracts infromation from the course book Digital Image Processing, Gonzales [1] unless other sources stated.

C.1 Primary Colors

The use of images is a great way of providing information visually. An image is represented by a rectangular grid where each cell, or hereafter pixel, emits light of a certain wavelength. When the color of each pixel is determined and is ready to be displayed, there is a problem of how to convey the color information from the source to the recipients eyes. On way would be to many different light sources for each pixel, where each source is responsible for a small region of wavelengths. This is problematic due to our wide range of visible colors and in order to reduce the number of light sources we need, these wavelength regions must be chosen with care. This problem is slightly simplified by the fact that our eyes consists of three different sensors that react to different wavelengths, and these are roughly equal to blue, green and red colors and their absorption can be seen in Fig. C.1.

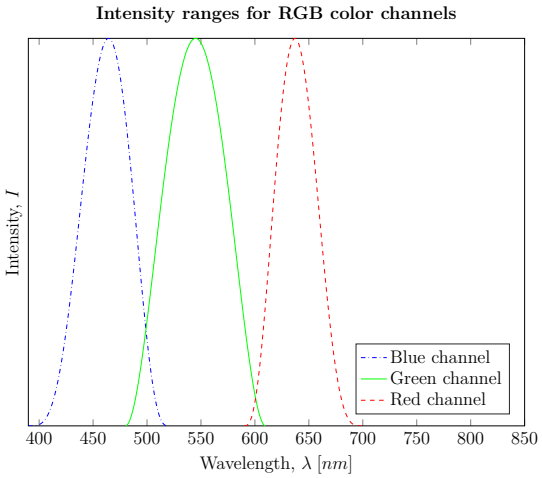
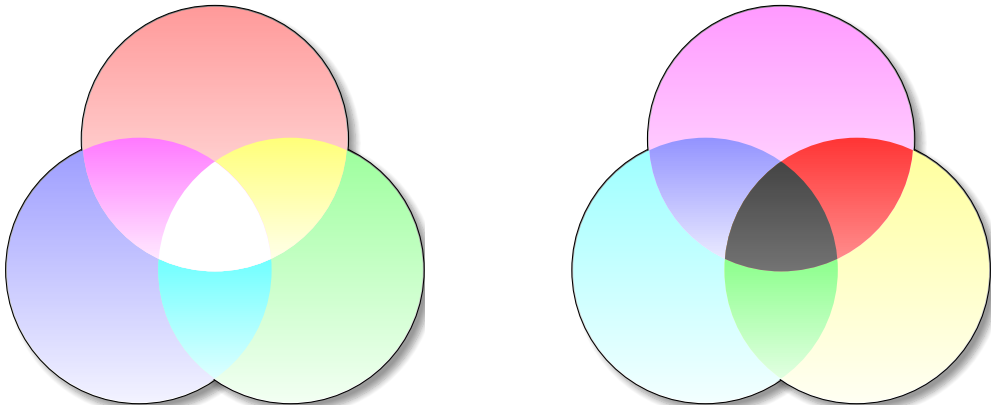


Figure C.1: When incoming photons with different wavelengths λ hits each receptor, they get excited depending on their ranges. The figure shows, not accurate ranges nor intensities, but illustrates a schematically the different ranges for the different colors.

These three colors, red, green, and blue are therefore called the *primary colors* and will be the basis for image representations. There are two different ways of using

these colors to provide the targeted color, in a additive or a subtractive manner. When adding the primaries we start with a completely black canvas and increase the intensity for each of the colors until we reach a white color. For the subtractive nature we start with a white canvas and determine how much of each color we want to subtract until a black color is reached. The difference between these can be thought of as absorption and emittance, and the distinction can be seen in Fig. C.2.



(a) **(b)**

Figure C.2: Combining colors can either be done additively *(a)*: The normal **RGB** color space combines red and blue to magenta,blue and green to cyan, green and red to yellow, and all three to white) or subtractively *(b)*: the **CMY** color space combines colors subtractively by removing the selected colors from a pure white color).

C.2 Color Spaces

C.2.1 RGB and CMY

In the introduction of this appendix we talked about the primary colors and how they are used to represent a wide range of colors. This representation of colors is the **RGB** (red, green, blue) color space for the additive colors, and **CMY** (cyan, magenta, yellow) for the subtractive. Many other color spaces exists and are used in different situations. Ultimately they represents the same color, only through other variables than red, green, and blue. Often the change of color spaces is done by changing the basis for the color representations. E.g. transforming from **RGB** to **CMY** color space is done by the linear transformation,

$$\begin{pmatrix} c \\ m \\ y \end{pmatrix} = 1 - \begin{pmatrix} r \\ g \\ b \end{pmatrix}, \tag{C.1}$$

which makes it clear why the other is called subtractive when the first is additive. Although we can change the color space in which we represent an image, the fact that our screens use red, green, and blue colors to mix colors is still present. So to view a new color space we simply use these light sources and pretend that our new

values still lies in a **RGB** color space. The result of this representation can be seen in Fig. C.3.

(a) **(b)**

Figure C.3: Presenting images using other color spaces than the ordinary **RGB** color space can yield interesting result. *(a)*: Image show in the ordinary **RGB** color space. *(b)*: The same image as in *(a)* but show in **RGB** representation using **CMY** color values.

Sometimes the information that is interesting in the image in not needed to be expressed using three color channels and a color conversion might only produce a single index for each pixel. These will be called color indices, and basically any color space is build by three different color indices, due to the three different kinds of receptors. The difference between three random color indices and a color space is that the color space spans the entire "color room", meaning that a color space can represent any color and conversion between spaces exists, which is not always true for color indices. Several color spaces/indices will be presented as well as some reason why using them.

C.2.2 HSI

In many computer vision applications there is a need to separate segments of images based on the color of these. This could be tedious if done using the **RGB** color space. E.g. suppose you would want to select a yellowish color, but you care not for an exact color and allow some deviation. A color value is chosen as base, and small deviations in the red color channel will yield about the same color, but also small deviations in the green and blue channels will produce almost the same result. So to select the appropriate range of colors that are representative is dependent on three different deviations, and the target lies in a three dimensional shape. So a color space in which the information of the color is embedded in one of the channels could prove useful. We can imagine that all available colors are represented on a disk where the colors changes on the viewing angle from the origin, and the prevalence of this color increases with distance. So to choose a specific color we would simply choose an angle from the origin which points at the targeted color. Then we would go in that direction until the amount of color is reached and then we increase/decrease the intensity of the selected color. This is a new color space which we call **HSI**,

Hue, **S**aturation, and **I**ntensity. The color coordinates for this color space is here represented by an angle, a radius and a height compared to three distances for the **RGB** color space, the differences is show in Fig. C.4

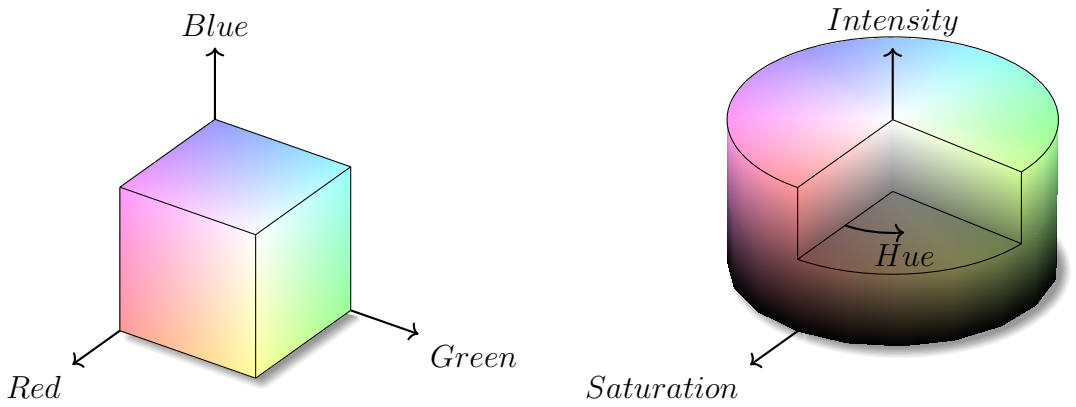


Figure C.4: Different color spaces uses different coordinates to represent the color room. In *(a)* we see the **RGB** color space which is spanned by orthonormal basis, whereas in *(b)* which shows the **HSI** color space uses an angle, a radius and a height instead.

C.2.2.1 RGB to HSI

Converting between **RGB** and **HSI** is a non-linear transfrom and can thusly not be written in matrix form. In the following calculations the values in **RGB** space is written as $R, G,$ and$B$ and **HSI** $H, S,$ and$I$. Also a **RGB** color is assumed to be in $C_{RGB} \in [0,1]^3$ and **HSI** $C_{HSI} \in [0°, 360°] \times [0,1]^2$. The Hue component is an angle and trigonometric calculations are therefore necessary,

$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360° - \theta & \text{if } B > G \end{cases}, \tag{C.2}$$

where,

$$\theta = \cos^{-1}\left\{ \frac{\frac{1}{2}\left[(R-G)+(R+B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}} \right\}, \tag{C.3}$$

whereas the Saturation and Intensity is easier to calculate,

$$S = 1 - \frac{3}{R+G+B}\left[min(R,G,B)\right], \tag{C.4}$$

and

$$I = \frac{1}{3}(R+G+B). \tag{C.5}$$

C.2.2.2 HSI to RGB

Converting back to RGB colors is also a tedious task, mostly due to that we need three different algorithms depending on the value of H.

$0^\circ \leq H < 120^\circ$

$$\begin{aligned} R &= I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right] \\ G &= 3I - (R + B) \\ B &= I(1 - S) \end{aligned} \tag{C.6}$$

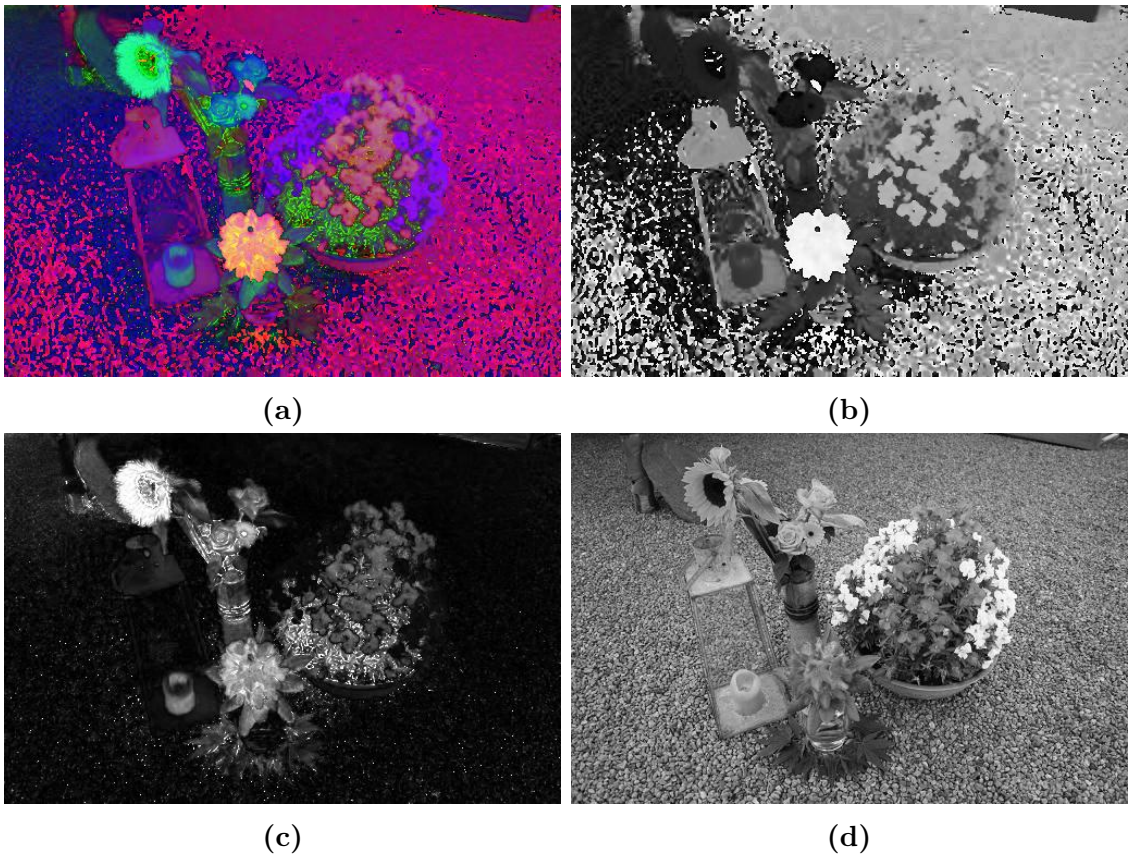
$120^\circ \leq H < 240^\circ$

$$\begin{aligned} H' &= H - 120^\circ \\ R &= I(1 - S) \\ G &= I\left[1 + \frac{S\cos H'}{\cos(60^\circ - H')}\right] \\ B &= 3I - (G + B) \end{aligned} \tag{C.7}$$

$240^\circ \leq H \leq 240^\circ$

$$\begin{aligned} H' &= H - 240^\circ \\ R &= 3I - (G + B) \\ G &= I(1 - S) \\ B &= I\left[1 + \frac{S\cos H'}{\cos(60^\circ - H')}\right] \end{aligned} \tag{C.8}$$

The applicability of the **HSI** color space is hopefully evidently shown in Fig. C.5.



(a) **(b)**

(c) **(d)**

Figure C.5: It is not evident from the **RGB** representation of the **HSI** color image shown in *(a)* that the colors can be grouped based on one color channel, but is clearer from the Hue channel in *(b)*, which shows large groupings for different parts of the image. The Saturation channels shows occasional grouping in *(c)* but is not as uniform as the Hue, and the Intensity in *(d)* shows large variations.

C.3 Color Indices using Near Infrared wavelengths

One great thing with using electromagnetic radiation (light) as a source of information is that we do not need to use wavelengths comprehensible to our brains. The implications of this gives us the ability to use light information from wavelengths outside the visible spectrum. This can be of immense help when gathering information from biological living things which usually function in certain ways due to physical or chemical consequences. To further explain this we realise that light affect biological cells by exciting its building components to a higher energy state, and if the energy in a photon does not lie within the range of the possible excitation energies, there is no reason for a cell to absorb that light. So the reason why we cannot see certain wavelengths is due to almost full reflectance of those energies in our eyes receptors, and the same is true for most biological molecules as they are of the same order of magnitude in size and should be affected by roughly the same wavelengths, albeit in different ways. If we could somehow capture the near infra red light ($\lambda \approx 750 - 1400\,\text{nm}$) we could "see" the healthiness of a plant as a sick and degenerate cells does not behave as a living, healthy one as the molecules starts to

crumble and behave differently.

C.3.1 NDVI

NDVI or **N**ormalized **D**ifference **V**egetation **I**ndex is used to differentiate healthy vegetation from either unhealthy or sparse vegetation sections[2]. What enables the usage of this index is that active chlorophyll in plants strongly absorbs visible light and a living cell strongly reflects near infrared light as it can not use this energy internally, and absorbing it would cause overheating. On the other hand, inactive or dead chlorophyll does not absorb the visible light to the same degree whereas the cell reflects roughly the same amount of near infrared light even if the chlorophyll is degenerate. The index is calculated by the formula,

$$NDVI = \frac{nR - VIS}{nR + VIS}, \tag{C.9}$$

where nR is the near infrared index and VIS is the representation of the visible light. This index ranges from -1 to +1 where a number close to +1 represents a healthy plant. Besides determining the health of plants, this index can also be used to determine what part of an image is a plant or not as a low index corresponds to sparse or no vegetation.

C.3.2 SAVI

The previously discussed index, **NDVI**, is great when determining the health of vegetation that is dense. If, on the other hand, the vegetations is sparse, then the reflectance of the soil could come to matter. Thus a **S**oil-**A**djusted **V**egetation **I**ndex [3], **SAVI**, could help with this problem which is a modification of **NDVI** using a *Soil brightness correction factor*, L is introduced to and extends Eq. (C.9) to,

$$SAVI = \frac{nR - VIS}{nR + VIS + L}(1 + L), \tag{C.10}$$

where $SAVI(L = 0) = NDVI$. The factor L is adjusted by the amount of green vegetation, where $L = 1$ means a low amount of greenness and $L = 0$ a large amount.

C.3.3 MSAVI

The biggest problem with the **SAVI** index is the parameter L which is determined for each problem by trial-and-error. The **M**odified **S**oil-**A**djusted **V**egetation **I**ndex [4], **MSAVI** is calculated the same way as **SAVI** Eq. (C.10),

$$MSAVI = \frac{nR - VIS}{nR + VIS + L}(1 + L), \tag{C.11}$$

only that in this case, L is not a constant but is calculated using,

$$L = 1 - \frac{2s \cdot (nR - VIS)(nR - s \cdot VIS)}{nR + VIS}, \tag{C.12}$$

where s is yet another parameter, but is a equipment dependent parameter rather than application dependent one.

C.3.4 MSAVI2

The final Color index we will consider using the near infrared channel is completely parameter free, but the expression looks nastier. This new index is derived from **MSAVI** using recursion (the details will not be given here) and the final expression is calculated by,

$$MSAVI2 = \frac{\left(2nR + 1 - \sqrt{(2nR+1)^2 - 8\,(nR - VIS)}\right)}{2}. \tag{C.13}$$

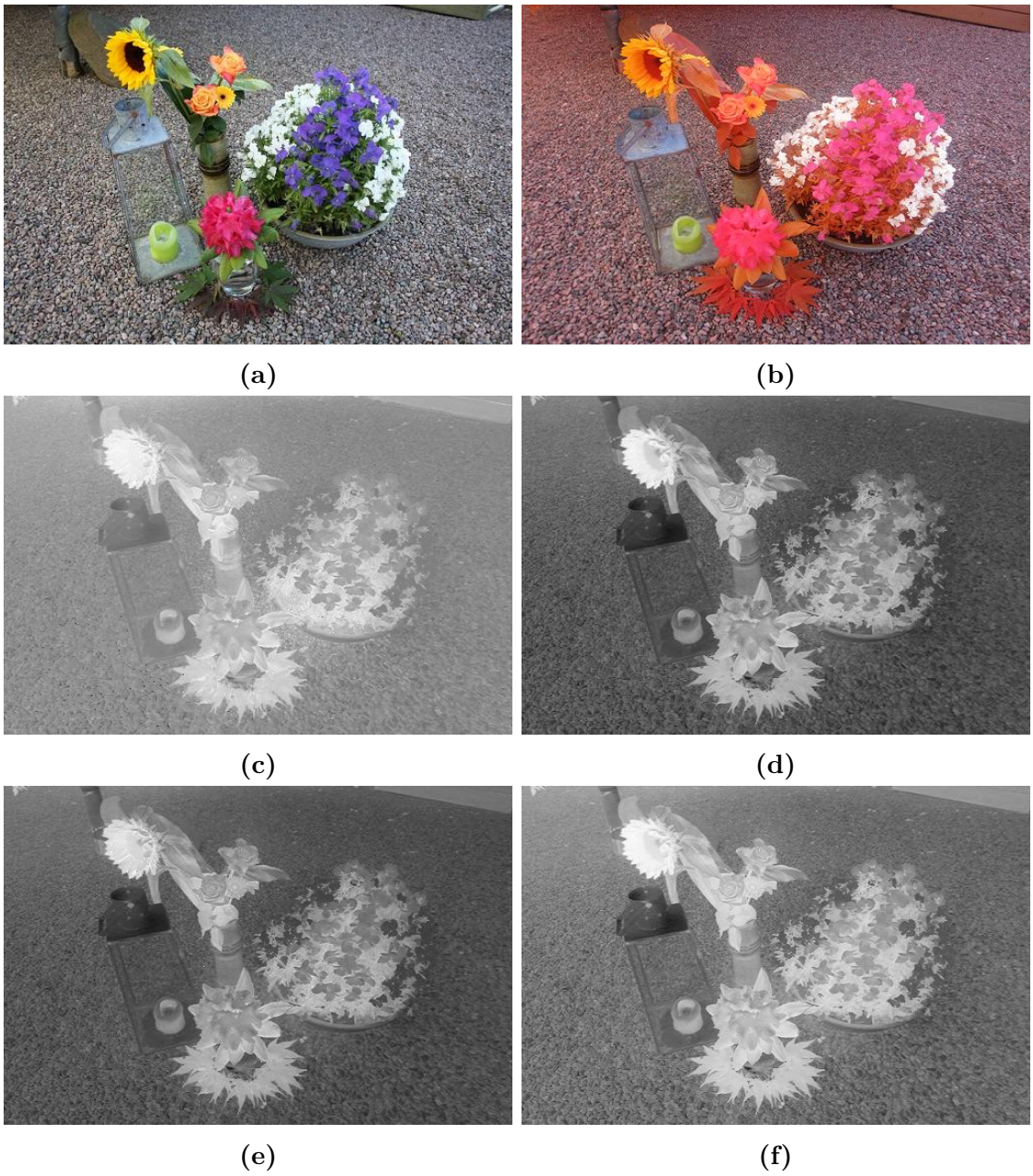
Use the indices for healthy and unhealthy plants, make new figure showing the comparisons

These four color indices can be seen in Figures C.6c to C.6f.

C.4 nRGB to RGB

Above it is mentioned that near infrared light can come to help in various image analysis problems. However, this does not come for free, unless having access to really special equipment, most image light sensors only provide three different channels for capturing. Thus, the inclusion of a new channel, near infrared, implies the exclusion of another. The simplest would be to change it with the red channel as it lies closest in wavelength. If viewed using the **RGB** format, the resulting image will most likely be very reddish in color as most materials reflect the near infrared light, the comparison can be seen in Figures C.6a and C.6b

Use this image to show rgb and nrgb channels



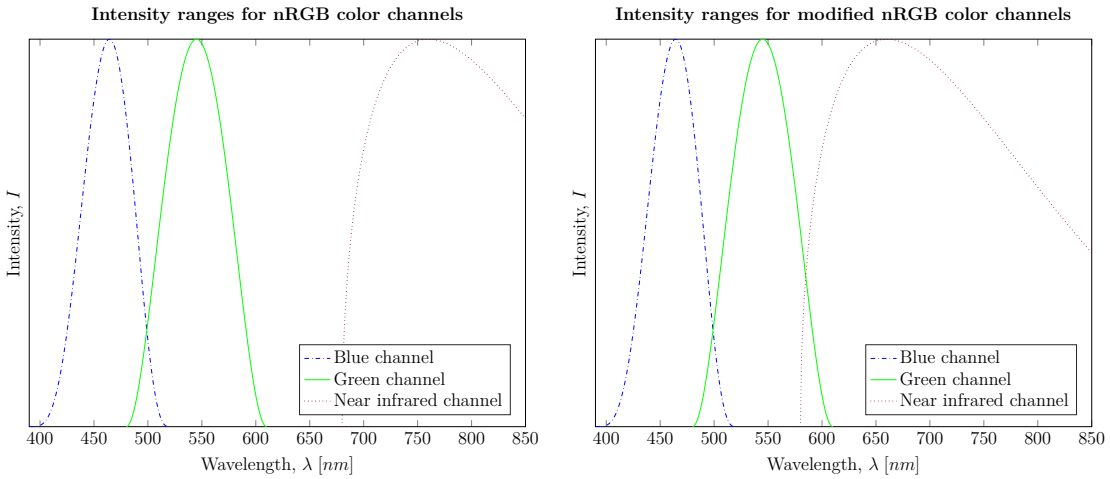
(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure C.6

In order to show the **nRGB** image we want to "simulate" it as it was an ordinary **RGB** image. So we want to move the wavelengths of the new channel to a targeted one. Mathematically this is done using a convolution, described in Chapter. , with convolving function, $\delta(x)$ where x is the targeted channels mean. This modification, replacing the red channel with the near infrared channel can be seen in Figure. **??**.



(a) **(b)**

Figure C.7: In *(a)* we can see the capturing of light in the **nRGB** color space. In order to display it using **RGB** color channels, we need to modify the wavelengths to the ordinary of **RGB** colors. This modification of the near infrared channel is showns in *(b)*.