Extraction of Surface Features to Predict Defects

MASTER'S THESIS

Conducted in partial fulfillment of the requirements for the degree of a

Diplom-Ingenieur (Dipl.-Ing.)

supervised by

Ao. Univ.-Prof. Dipl.-Ing. Dr. techn. M. Vincze

submitted at the

TU Wien

Faculty of Electrical Engineering and Information Technology

Automation and Control Institute

by

Johannes Huemer, BSc

Wien, March 7th, 2019

Vision for Robotics Group

A-1040 Wien, Gusshausstr. 27, Internet: http://www.acin.tuwien.ac.at

Acknowledgements

First of all, I would like to thank the company Profactor who made this master's thesis possible and, especially, Dr. Christian Eitzinger who supported me with his patient guidance throughout the whole project. His professional suggestions were of great help in finishing this work.
I would also like to sincerely thank Professor Dr. Markus Vincze for supervising this thesis and providing me with advice wherever it was necessary.
Finally, I am very thankful to my family for their unconditional support and especially Lejla Bico for her steady encouragement throughout the whole studies.

Johannes Huemer
Wien, March 7^{th}, 2019

Abstract

A common approach for quality assurance in manufacturing is to reject faulty products before they can reach the customer. This is often achieved using manual or automated visual inspection systems. The automated approaches can be of advantage when an accurate prediction of the quality level is needed, as they perform more consistent than humans. What is more, in many applications, machine vision methods can be used for on-the-line real time inspection which is manually not possible.
In Zero Defect Manufacturing (ZDM) the idea is to prevent defective products in the first place, by for example monitoring the quality of the manufacturing process itself and e. g. replacing worn tools if necessary. Within the scope of ZDM, this thesis introduces a machine learning approach for quality assurance of boreholes in carbon fiber reinforced polymers (CFRP). With the aid of visual analysis and intelligent learning methods, the quality of the manufacturing process is predicted by correlating drilling parameters with visual features extracted from images of the boreholes. The features proposed in this work include low-level characteristics like entropy or homogeneity of the gray level images, as well as high-level features like detection of interest regions. We also use a basic photometric stereo approach to extract 3-dimensional information from the borehole surface as input for the machine learning model. To provide the model with the most relevant information, several feature selection algorithms are evaluated and for the final prediction of the process quality category, three classification methods are compared.
The results show that the predictability of the drilling quality class is around 80% for both datasets in evaluation and that the separability between optimal and very poor drilling conditions like fractured tools is very distinct. In addition, the results approve the use of feature selection as a method to reduce the complexity of the machine learning problem while keeping the classification performance almost similar or even improving it.

Kurzfassung

Ein üblicher Ansatz zur Qualitätssicherung in der industriellen Fertigung besteht darin, fehlerhafte Produkte auszusortieren, bevor sie den Kunden erreichen können. Dies wird häufig durch manuelle oder automatisierte visuelle Inspektionssysteme erreicht. Die automatisierten Ansätze können von Vorteil sein, wenn eine genaue Vorhersage des Qualitätsniveaus erforderlich ist, da sie konsistenter sind als manuelle Prüfungen. Außerdem können Bildverarbeitungsmethoden für Inline-Inspektionen verwendet werden, die manuell so nicht möglich sind.
Beim Zero Defect Manufacturing (ZDM) geht es darum, fehlerhafte Produkte generell zu vermeiden, indem beispielsweise die Qualität des Fertigungsprozesses selbst überwacht und gegebenfalls abgenützte Werkzeuge ausgetauscht werden. Im Rahmen von ZDM wird in dieser Arbeit ein Machine Learning Ansatz zur Qualitätssicherung von Bohrungen in kohlefaserverstärkten Kunststoffen vorgestellt. Mit Hilfe einer visuellen Analyse und intelligenten Lernmethoden wird die Qualität des Fertigungsprozesses vorhergesagt, indem Bohrparameter mit visuellen Merkmalen korreliert werden, die aus Bildern der Bohrungen extrahiert werden. Zu den in dieser Arbeit vorgeschlagenen Merkmalen gehören einfachere Charakteristiken wie Entropie oder Homogenität der Graustufenbilder sowie auch komplexere Merkmale wie das Erkennen von Bereichen von Interesse. Ebenso wird ein grundlegender photometrisch-Stereo Ansatz verwendet, um 3-dimensionale Informationen der Bohrlochoberfläche als Input für das Machine Learning Modell zu extrahieren. Um dem Modell möglichst relevante Informationen bereitzustellen, werden mehrere Algorithmen zur Merkmalsselektion ausgewertet und für die endgültige Vorhersage der Kategorie der Prozessqualität werden drei Klassifizierungsmethoden verglichen. Die Ergebnisse zeigen, dass die Vorhersagbarkeit der Klasse der Bohrlochqualität für beide verwendeten Datensätze etwa 80% beträgt und dass die Trennbarkeit zwischen optimalen und sehr schlechten Bohrbedingungen, wie zum Beispiel beschädigten Werkzeugen, sehr gut möglich ist. Darüber hinaus befürworten die Ergebnisse die Merkmalsselektion als Methode, um die Komplexität des maschinellen Lernproblems zu reduzieren, während das Ergebnis der Klassifizierung nahezu gleich bleibt oder sogar verbessert wird.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Network
CART	Classification and Regression Trees
CCD	Charge-Coupled Device
CFRP	Carbon Fiber Reinforced Polymers
CFS	Correlation-based Feature Selection
CLAHE	Contrast Limited Adaptive Histograph Equalization
DT	Decision Tree
FAST	Features from Accelerated Segment Test
GLCM	Gray Level Co-occurrence Matrix
HDR	High Dynamic Range
HOG	Histogram of Oriented Gradients
MATLAB	Matrix Laboratory
MSER	Maximally Stable Extremal Region
PC	Principal Component
PCA	Principal Component Analysis
PCB	Printed Circuit Board
RF	Random Forest
ROI	Region of Interest
SFS	Shape from Shading
SIFT	Scale Invariant Feature Transform
SMLT	Statistics and Machine Learning Toolbox
SMOTE	Synthetic Minority Over-sampling Technique
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
UD	Uni- Directional
ZDM	Zero Defect Manufacturing

1 Introduction

This work is part of a range of projects within the term Zero Defect Manufacturing (ZDM) by the Austrian company Profactor [1]. The primary goal of these projects is to develop methods and tools to ensure product quality by controlling the parameters of industrial manufacturing processes. Another aim of the projects is to inspect complex components which are hard to examine manually due to e. g. their geometry.
The concept of Zero Defects was first introduced as a revolutionary management tool in the 1960s [2], with the aim of reducing defects in manufacturing through prevention. Until then, the quality assurance in production was carried out by end-of-line inspection which rejected faulty products before they could reach the end customer. Zero Defects shifted the focus from the quality inspection to controlling the manufacturing process. This is for example achieved by using information from methods for quality assurance like visual inspection together with data from the manufacturing process to train prediction models. The machine learning model allows to keep the processes in a certain quality tolerance range or helps to adjust parameters of the process, especially if new variants of products are introduced. In contrast to other quality assurance approaches like sorting out defective products after manufacturing, this concept aims to prevent defects in the first place, by closing the feedback loop to the process. This avoids for example costly quality inspection to discover defective products and also the need to rework them, as this is almost always more expensive and laborious than examining the reasons for the defects during the manufacturing process and trying to find methods to prevent them in the future.

1.1 Problem Statement

The concrete problem this work is dealing with, is the quality assurance of boreholes in carbon fiber reinforced polymer (CFRP) parts. During the process of drilling holes into the components, inhomogeneities in the layered material lead to defects in the form of delaminations and fraying on the inside surface of the boreholes, as shown in Figure 1.1 on the right side. As the condition of the holes is very important for the stability of the rivet connections, there has to be a certain type of quality inspection after the drilling. Thus, Profactor developed a prototype for visual inspection in the project HScan [3], which allows the examination of the inside surface of the holes with an endoscope camera. This very small-geometry imaging system provides multiple pictures from the same recording position with varying illumination angles, to be able to inspect the whole surface. The left picture

in Figure 1.1 shows the image acquisition process and Figure 1.2 illustrates the principal setup of HScan.
To prevent defect boreholes in the first place, an approach is to ensure the quality of the drilling process itself, like selecting appropriate manufacturing parameter values and controlling the condition of the tools. So far, the method was to ensure the quality by frequently changing the drilling tools.

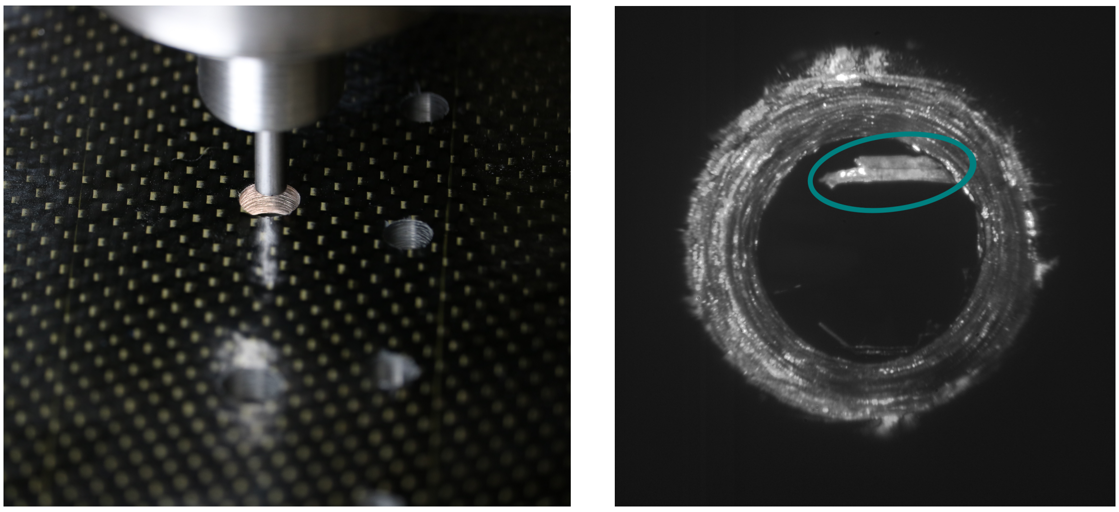


Figure 1.1. Example of image acquisition process with HScan (left) and image of defective borehole with encircled fiber pullout (right). Source: [1]

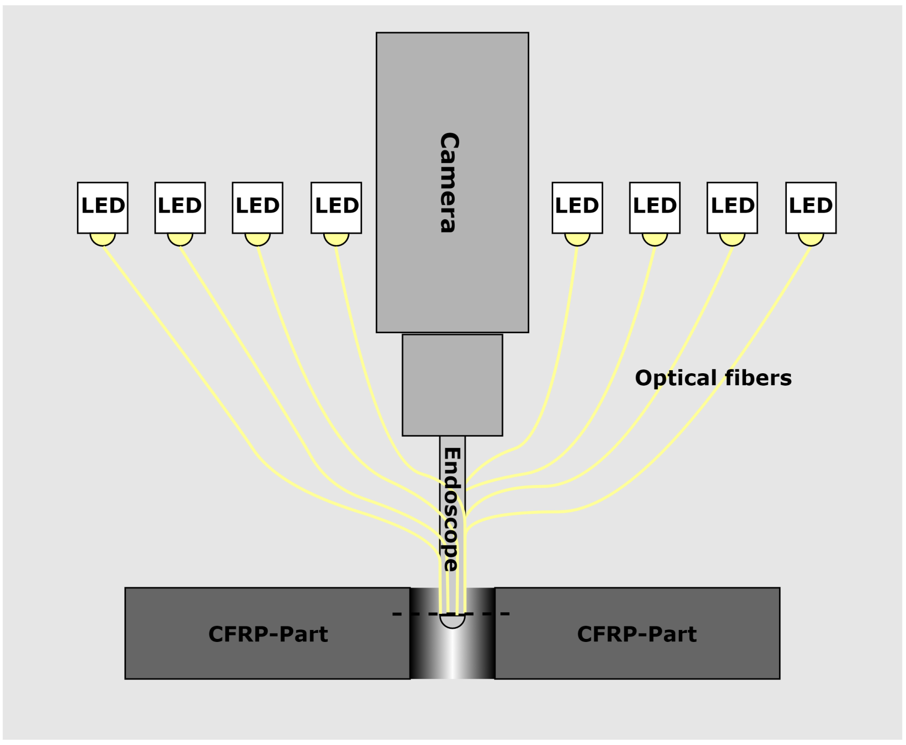


Figure 1.2. Principal setup of the endoscope imaging system.

1.2 Proposed Solution

In this thesis, we propose a machine learning approach to ensure the quality of the drilling process by correlating the information from the visual inspection results of HScan with parameters and tool conditions in the boring process. Figure 1.3 shows a graphical overview on the proposed implementation.

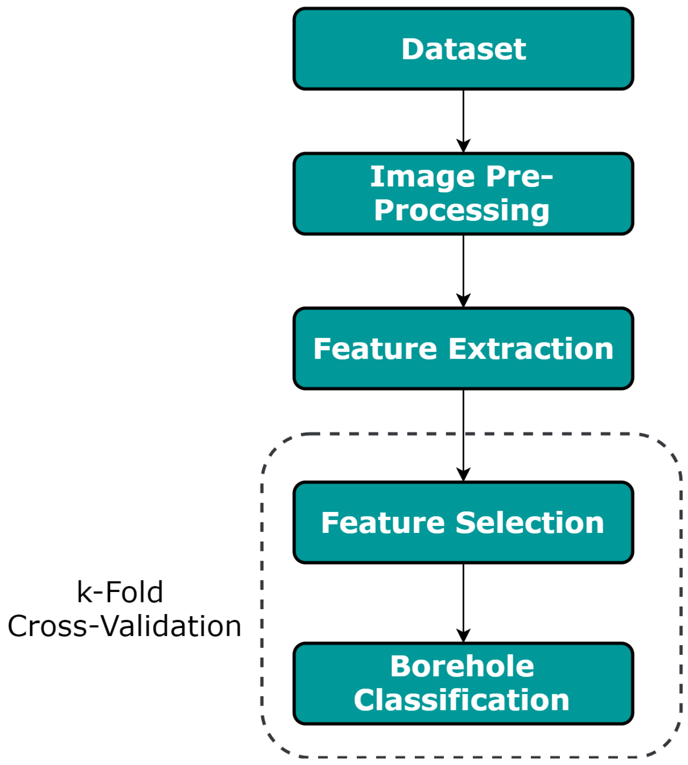


Figure 1.3. Graphical overview of the proposed machine learning implementation.

The first step in this work is the data preprocessing, starting with stitching together the images acquired from different illumination angles. We then segment the resulting images into the inside surface of the borehole and the interior. Following, several surface features are extracted from the segmented images of which the most relevant are then used to create a machine learning model which predicts the quality of the drilling process.
The features which we are going to present in this thesis include texture based features like the entropy or homogeneity of the image and also higher level features like the results of interest point detections. In addition, a simple photometric stereo implementation is used to estimate the surface normals of the borehole inside from which more features are extracted.
The relevance of the surface features is determined using several feature selection approaches and then, for the final prediction of the quality category, we evaluate three classification algorithms.

1.3 Outline

This thesis is structured as follows. The next chapter, Chapter 2, reviews related work in terms of machine learning systems for visual inspection. Chapter 3 explains the mathematical background on the methods used in this work. In Chapter 4, the first three steps of Figure 1.3 - the dataset, the preprocessing of the images and the following surface feature extraction - are discussed in detail. The last two steps - feature selection and classification of the boreholes - are presented in Chapter 5. In addition, this chapter discusses the evaluations of the methods proposed in this thesis. The last chapter summarizes the thesis and gives an overview on possibilities for future work.

2 Related Work

As already stated in the last chapter, this work is based on the results of a visual inspection system called HScan [3], developed by Profactor, which is used to ensure the quality of the inside of boreholes in CFRP parts. This thesis presents a machine learning approach for the quality assurance with the aim of preventing defective bores in the first place. Therefore, in this chapter we discuss related work in the domain of quality inspection, especially with visual methods. In addition, previous work on image segmentation and also on photometric stereo approaches is reviewed. Finally, various projects which are similar to this work in terms of machine learning methods are addressed.

Visual Inspection and Image Segmentation

Machine vision is an automated and cost-effective way to provide visual inspection for a variety of applications. In a survey of visual inspection works [4], Newman and Jain state that automatic is preferable over manual inspection in many applications. For example, automatic approaches perform rather consistent, therefore the quality level can be predicted better. Another advantage is the on-the-line real time inspection with machine vision approaches which is possible in many environments if the requirements of the manufacturing process are met. Finally, many inspection tasks might be too boring or time-ineffective for a human to carry out.
One example for visual inspection is the work of Tien et. al [5]. They propose an automated visual approach for detecting major defects of microdrills, based on color images acquired by a charge-coupled device (CCD) camera. The regions of interest are segmented via thresholding and boundary detection which is achieved using corner detection and a subsequent least-squares line fitting. This segmentation procedure is in principal kind of similar to the proposed approach in this thesis. For the following defect detection, various geometric properties of the segmented regions are extracted and evaluated.
Another more recent machine vision application for inspection is presented in [6]. Wang et. al developed an optical system for measuring the quality of drillings in printed circuit boards (PCB). This system is designed to examine boreholes with a specified size of 2 mm, therefore it needs an appropriate imaging resolution. This is achieved with CCD cameras which are moved across the PCBs while acquiring images of sub-regions of the boards. All sub-images for a board are registered in the end, to one large image. The boreholes are then separated from the surrounding regions by gray-level thresholding. For the evaluation of the boreholes, several geometric features are calculated, like the center coordinates, the area or the roundness of the hole. In this paper, the authors also point out that for a successful

distinction between holes and noise, a suitable illumination is an important factor. Other related works on vision inspection systems include, amongst others, detecting surface defects of strongly reflective metals [7], or quality inspection of multi-axial non-crimp fabrics [8].

Photometric Stereo

The idea of using multiple images, taken from the same position with illumination from different directions, to compute surface orientations, was introduced in 1992 by Woodham [9]. Since then, the concept was applied in many practical applications, including surface inspection.
In [10], photometric stereo is used as an improvement over 2D defect detection for steel strips. The proposed 3D approach can distinguish between real defects and pseudo-defects which are falsely detected by 2D methods. In this work, an ideal diffuse reflection model of the steel surface is used for the photometric stereo approach. For the image acquisition, they use a special splitting prism camera, which avoids interferences among R, G and B color channels and thermal radiation, as there are also samples of hot steels in inspection. The illumination is provided by a set of linear-ranged lasers in blue and green, again in order to avoid interference with thermal radiation. Due to the proposed type of photometric stereo approach, only two image channels (blue and green) are required.
Palfinger et. al present a photometric stereo based inspection system that evaluates quality of carbon fiber surfaces in [11]. The imaging geometry used in this work is a combination of CMOS cameras together with high power LEDs placed in a circle around the optical axis for illumination. As carbon fibers have special reflective properties, a method to compute the reflectivity of the surface is proposed. The results of the photometric stereo approach are the fiber orientation and a good estimate for diffuse and specular reflectivity. The individual fiber segments are detected using the proposed fiber line segmentation which allows fast and robust partitioning. From the segment contours, various characteristics like orientation information or width and height are used as features for defect detection.

Feature Extraction

An important requirement for a well performing machine learning model is a suitable data preparation. In the case of visual inspection, this data consists of information extracted from images acquired by the optical system. The extracted information is also referred to as features of the image. In this section, several works which use feature extraction as a basis for visual inspection are discussed.
The work from Palfinger et. al [11] was already mentioned in the last section. The information extraction is divided into first and second order features where first order characteristics contain information of single fiber segments like mean and standard deviation of width, height and orientation. The second order features are derived from the relation of adjacent segments, like orientation differences.

In [12], an auto-inspection system for molding surfaces in integrated circuit manufacturing is presented. The quality of the molding process is responsible for the chip's resistance against damages caused by e. g. external forces. Thus, defects like cracks or voids have to be detected accurately. Several features which are used to distinguish between defects and regular patterns are proposed. These include the area, the average gray value, the eccentricity and the ellipticity of a segmented region. Depending on the feature values, binary decisions are made to determine whether a region corresponds to a regular pattern or a defective segment.
A rather different approach to the aforementioned works is presented in [13]. Weimer et. al propose an automated feature extraction system for industrial inspection using deep convolutional neural networks. The architecture is given an image as input and in each stage of the process a convolution, non-linear neuron activation and feature pooling are used to automatically extract features. The advantage of this approach is that for different types of inspection problems, no manual redefinition of features is needed. In addition, only a minimum of expert knowledge of the specific process is required.

Classification

The feature extraction results are used to identify the category or state a certain sample belongs to. In defect detection, this is often a binary choice between defective and faultless.
Weimer et. al [14] propose a machine vision system for defect detection on textured surfaces. They use a neural network learning model for binary classification of image patches. This model is trained from images with premarked defective regions. The proposed system is able to detect 100 % of the defects for the evaluated real scenario.
The approach from Zhang et. al [7] uses a multi-class Support Vector Machine (SVM) to detect seven classes of defects on strongly reflective metal surfaces successfully. They also evaluate different kernel functions and parameter settings for the SVM.
In [15], Ravikumar et. al propose a machine learning system for visual inspection of machine components. The state of the components is divided into three categories, depending on the quality. As classification algorithms, the authors evaluate a decision tree approach and a Naïve Bayes classifier. The results show that the decision tree performs superior to the Naïve Bayes.
Two other works on flaw detection ([16],[17]) use convolutional neural networks for classifying defects for various types of surfaces.

3 Background

3.1 Least Squares Problem

The starting point of this topic is a problem in data analysis, where a certain value b is determined through taking measurements. Assuming that b depends on various parameters, collected in a parameter vector $\mathbf{p} \in \mathbb{R}^n$, ideally n measurements have to be taken to identify the relation between b and $\mathbf{p}$. This process can be described by a system of linear equations in the form of $\mathbf{Ap} = \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is the coefficient matrix, $\mathbf{b} \in \mathbb{R}^m$ the measurement vector and $\mathbf{p}$ the vector of unknown parameters, with the formalization in matrix form being

$$\underbrace{\begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \dots & A_{mn} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}}_{\mathbf{p}} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}}_{\mathbf{b}}. \tag{3.1}$$

In the aforementioned ideal case, $m = n$ is sufficient to determine $\mathbf{p}$. In reality, measurement errors and model uncertainties make additional measurements necessary. Therefore, $m > n$ and (3.1) is now called overdetermined. The problem in this situation is that in most cases (where $\mathbf{A}$ has full rank), overdetermined systems have no exact solution. The approach is now, to find an optimal solution $\hat{\mathbf{p}}$ such that $\mathbf{A}\hat{\mathbf{p}}$ is as close as possible to $\mathbf{b}$ [18]. This is achieved using a least squares approach, by minimizing the ℓ_2-norm of the vector of residuals $\mathbf{r}(\mathbf{p}) = \mathbf{Ap} - \mathbf{b}$ as follows [19]:

$$\hat{\mathbf{p}} = \underset{\mathbf{p}}{\operatorname{argmin}}\, J(\mathbf{p}), \tag{3.2}$$

where

$$J(\mathbf{p}) = \frac{1}{2}\mathbf{r}^T\mathbf{r} = \frac{1}{2}\left\|\mathbf{Ap} - \mathbf{b}\right\|_2^2. \tag{3.3}$$

The solution for the minimization problem is found by computing the gradient of J and setting the result to zero:

$$\begin{aligned} \nabla_{\mathbf{p}} J &= \nabla_{\mathbf{p}}(\frac{1}{2}(\mathbf{Ap} - \mathbf{b})^T(\mathbf{Ap} - \mathbf{b})) = \nabla_{\mathbf{p}}(\frac{1}{2}(\mathbf{p}^T\mathbf{A}^T\mathbf{Ap} - 2\mathbf{b}^T\mathbf{Ap} + \mathbf{b}^T\mathbf{b}) \\ &= \frac{1}{2}(2\mathbf{A}^T\mathbf{Ap} - 2\mathbf{A}^T\mathbf{b}) \overset{!}{=} 0 \end{aligned} \tag{3.4}$$

The least squares solution [19] $\hat{\mathbf{p}}$ is now given by

$$\hat{\mathbf{p}} = \mathbf{A}^{\dagger}\mathbf{b}, \tag{3.5}$$

where $\mathbf{A}^{\dagger}$ is called the generalized inverse or pseudoinverse with

$$\mathbf{A}^{\dagger} = (\mathbf{A}^{T}\mathbf{A})^{-1}\mathbf{A}^{T}. \tag{3.6}$$

The following conditions for $\mathbf{A}^{\dagger}$ to be the generalized inverse of the matrix $\mathbf{A}$ have to be met [19]:

$$\mathbf{A}\mathbf{A}^{\dagger}\mathbf{A} = \mathbf{A}, \tag{3.7}$$
$$\mathbf{A}^{\dagger}\mathbf{A}\mathbf{A}^{\dagger} = \mathbf{A}^{\dagger}, \tag{3.8}$$
$$(\mathbf{A}^{\dagger}\mathbf{A})^{T} = \mathbf{A}^{\dagger}\mathbf{A}, \tag{3.9}$$
$$(\mathbf{A}\mathbf{A}^{\dagger})^{T} = \mathbf{A}\mathbf{A}^{\dagger}. \tag{3.10}$$

The direct inversion of the matrix $\mathbf{A}^{T}\mathbf{A}$ is often omitted by using numerical methods to compute the pseudoinverse, like the QR decomposition or the singular value decomposition for computational cost efficiency [18].

3.2 Photometric Stereo

There are various approaches in computer vision to reconstruct 3D geometry from 2D images. These methods together are sometimes referred to as *Shape from X* [20], as the shape of an object is determined using different information derived from the images or the imaging geometry.
In this work, an extension of the *Shape from Shading* (SFS) principle, called *Photometric Stereo*, is used. SFS was first introduced in the work of Horn [21] and is based on recovering the shape of a surface from variations in the intensity of an image. Generally speaking, the fraction of light reflected by the surface of a material depends, amongst others, on its optical properties and surface orientation. For many surfaces, the assumption that the reflection of incident light can be written as a function of the incident angle i, the emittance angle e and the phase angle g, depicted in Figure 3.1, holds. This reflectance function $\phi(i,e,g)$ determines the ratio of surface radiance to irradiance in the direction of the viewer [9].

Assuming an imaging geometry where the size of the objects in the scene is small compared to the camera distance, the object points (x,y,z) and the image points (u,v) can be used interchangeably, with $u = x$ and $v = y$. The negative z-axis is conveniently aligned in direction of the camera. This is called an orthographic projection.
The z-coordinate of the object and, thus, the object surface can be expressed as $z = f(x,y)$ and the corresponding surface normal is given by the gradient of $f(x,y)$ [9]

$$\mathbf{n} = \nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \\ -1 \end{bmatrix} = \begin{bmatrix} p \\ q \\ -1 \end{bmatrix}, \tag{3.11}$$

with the parameters p and q. In the case of orthographic projection and homogeneous illumination of the scene, the image intensity depends only on these gradient coordinates p and q, hence, the intensity of an image can be modeled as a function of both parameters by

$$I(x,y) = R(p,q), \tag{3.12}$$

with R being the reflectance map, which captures the reflections of an object for a particular imaging geometry.
The relation between $R(p,q)$ and $\phi(i,e,g)$ can be derived from the scene in Figure 3.1, with normalized dot products of the vectors $\mathbf{n} = [p\ q\ -1]^T$, $\mathbf{s} = [p_s\ q_s\ -1]^T$ and $\mathbf{v} = [0\ 0\ -1]^T$. These result in equations for $\cos(i)$, $\cos(e)$ and $\cos(g)$ [9]:

$$cos(i) = \frac{1 + pp_s + qq_s}{\sqrt{1 + p^2 + q^2}\sqrt{1 + p_s^2 + q_s^2}} \tag{3.13}$$

$$cos(e) = \frac{1}{\sqrt{1 + p^2 + q^2}} \tag{3.14}$$

$$cos(g) = \frac{1}{\sqrt{1 + p_s^2 + q_s^2}} \tag{3.15}$$

For an idealized example of the reflectance function $\phi_L(i,e,g) = \varrho \cos(i)$ for a Lambertian surface, which reflects radiance equally into all directions, the corresponding reflectance map can be expressed by

$$R_L(p,q) = \frac{\varrho(1 + pp_s + qq_s)}{\sqrt{1 + p^2 + q^2}\sqrt{1 + p_s^2 + q_s^2}}. \tag{3.16}$$

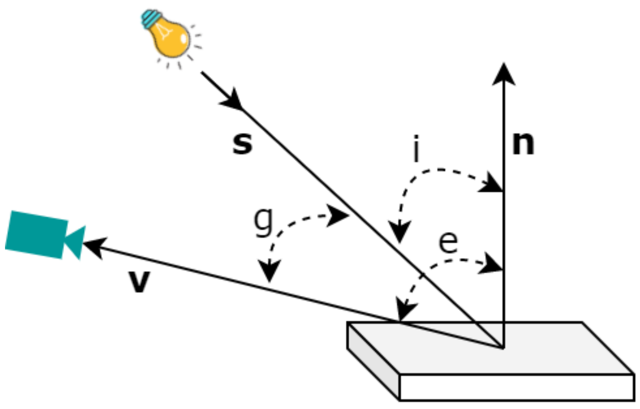


Figure 3.1. Depiction of the imaging geometry for a single light source.

Looking at (3.12), the image intensity can be used to determine the surface normals of an object. The problem is that for the parameters p and q there is only one partial differential equation, therefore, it is underdetermined. To solve this issue, on the one hand additional assumptions regarding special reflectivity or surface curvature can be made. On the other hand, the concept of photometric stereo, introduced in [9], can be applied.
The idea is to use multiple images (at least three), taken from the same position, with illumination from varying directions. This variation in the illumination results in a different reflectance map and intensity distribution for each image, whereas the surface orientation doesn't change. Thus, generally a system of equations in the form of

$$\begin{aligned} I_1(x,y) &= R_1(p,q) \\ I_2(x,y) &= R_2(p,q) \\ I_3(x,y) &= R_3(p,q) \end{aligned} \tag{3.17}$$

is satisfactory to uniquely determine the surface normals and the reflectance factor ϱ [9] in the case of (3.16). In certain other cases, only two images would be sufficient, however, the common approach is to use at least three images.
Assuming three images with different illumination, the unit surface normal $\mathbf{n} = [n_x\ n_y\ n_z]^T$ at (x,y) can be computed with the corresponding intensity values for the three images $\mathbf{I} = [I_1\ I_2\ I_3]^T$ and the unit vectors in direction of the illumination sources

$$\mathbf{L} = \begin{bmatrix} \mathbf{l_1} \\ \mathbf{l_2} \\ \mathbf{l_3} \end{bmatrix} = \begin{bmatrix} l_{x1} & l_{y1} & l_{z1} \\ l_{x2} & l_{y2} & l_{z2} \\ l_{x3} & l_{y3} & l_{z3} \end{bmatrix} \tag{3.18}$$

from the equation

$$\varrho\, \mathbf{L}\, \mathbf{n} = \mathbf{I}. \tag{3.19}$$

If the inverse $\mathbf{L}^{-1}$ exists (if there is no plane that can be fit to the vectors $\mathbf{l_1}$, $\mathbf{l_2}$ and $\mathbf{l_3}$), the solution for the reflectance factor and surface normals at the point (x,y) is [9]

$$\varrho\, \mathbf{n} = \mathbf{L}^{-1}\, \mathbf{I} \tag{3.20}$$

where

$$\varrho = |\mathbf{L}^{-1}\, \mathbf{I}| \tag{3.21}$$

and

$$\mathbf{n} = \frac{1}{\varrho} \mathbf{L}^{-1}\, \mathbf{I}. \tag{3.22}$$

3.3 Spherical Coordinate System

In the spherical coordinate system, the position of a point in 3-dimensional space can be expressed using three parameters, as shown in Figure 3.2. The first one is the radial distance r which is the distance between the point and the origin of the coordinate system. The second parameter is the angle θ from the zenith direction, which coincides with the z-coordinate in the Cartesian coordinate system, to the vector connecting the origin and the point $\mathbf{P}$. The last coordinate is the angle φ, measured from the x-coordinate to the projection of the vector between origin and expressed point onto the xy-plane.

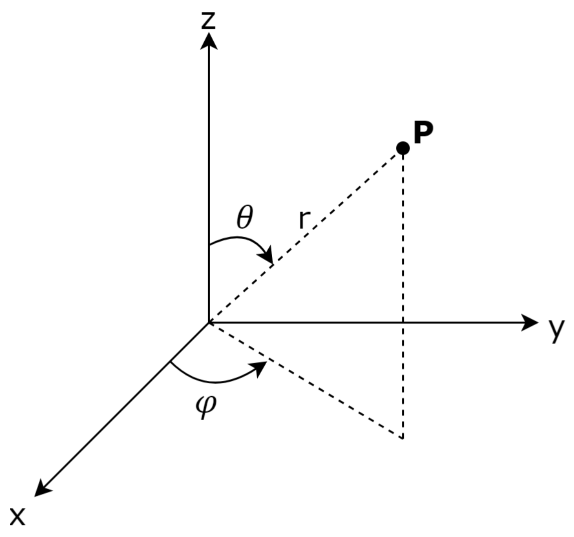


Figure 3.2. Spherical coordinate system.

The spherical coordinates can be derived from the Cartesian coordinates using the equations

$$r = \sqrt{x^2 + y^2 + z^2}, \tag{3.23}$$
$$\theta = \cos^{-1}\frac{z}{r}, \tag{3.24}$$
$$\varphi = \operatorname{atan2}(y,x). \tag{3.25}$$

The *atan*2 is used to take the correct quadrant of (x,y) into account. A common definition for the possible range for the three coordinates is as follows:

$$r \geq 0 \tag{3.26}$$
$$0^\circ \leq \theta \leq 180^\circ \tag{3.27}$$
$$0^\circ \leq \varphi < 360^\circ \tag{3.28}$$

3.4 Feature Selection

The objective of variable or feature selection is to improve the performance in learning applications, with faster, more accurate and more cost effective classifications by reducing the dimensionality of the data of the underlying process. This is achieved by testing the feature space for redundancies and then, using one of many applicable methods, to reduce them. For instance, redundancy is present when several features contain similar information which already could be preserved through keeping only one of these features. The task of the feature selection approaches is now to determine which features to keep in the model for the best classification results, especially for high-dimensional data [22].
In general, there are three different types of methods for feature selection which differ in how the subset selection process is combined with the creation of the classification model [22]:

- Filter methods
- Wrapper methods
- Embedded methods

The feature selection with filter methods is done independently before the learning algorithm and consists of two basic steps: ranking the features and selecting the best subset.
Wrapper methods evaluate the feature subsets based on the performance of the learning algorithm which is an inherent step in these methods. Thus, they are more complex regarding computation.
Finally, the embedded methods are similar to wrapper approaches, with the difference that the variable selection is part of the specific learning algorithm and is directly related to the construction of the model. An example for an embedded approach is a Random Forest [23].
In the following sections, several filter-based feature selection and dimensionality reduction concepts are described in detail.

3.4.1 ReliefF

The *ReliefF* algorithm, proposed in [24], is an extension of the *Relief* approach [25] for the multi-class case. The idea of this method is to calculate the weight or importance of a feature depending on the differences between neighboring observations of the same, as well as other classes, regarding the considered feature. A parameter k defines the amount of neighbors used for both cases (same and different class).
The first step is to set all weights w to 0. Then, for a random instance $\mathbf{x}_r$, the k-nearest observations for every class are found, with the neighborhood criterion being the ℓ_1-norm over all features. For each of these observations $\mathbf{x}_q$, the weight of the *j-th* feature is updated as follows:

If $\mathbf{x}_r$ and $\mathbf{x}_q$ have the same class:

$$w_j^i = w_j^{i-1} - \frac{\Delta_j(\mathbf{x}_r,\mathbf{x}_q)}{m} \tag{3.29}$$

If $\mathbf{x}_r$ and $\mathbf{x}_q$ have different classes:

$$w_j^i = w_j^{i-1} + \frac{p_q^y}{1-p_r^y}\frac{\Delta_j(\mathbf{x}_r,\mathbf{x}_q)}{m} \tag{3.30}$$

w_j^i describes the weight of the *j-th* feature for the *i-th* iteration step and p^y indicates the a priori probabilities for the corresponding class, depending on the total number of observations for each class. The parameter m is the amount of iterations and corresponds to the number of random observations which are used for updating a weight. $\Delta_j(\mathbf{x}_r,\mathbf{x}_q)$ defines the normalized difference in the values for the *j-th* feature for $\mathbf{x}_r$ and $\mathbf{x}_q$:

$$\Delta_j(\mathbf{x}_r,\mathbf{x}_q) = \frac{\left|\mathbf{x}_r^j - \mathbf{x}_q^j\right|}{max(\mathbf{x}^j) - min(\mathbf{x}^j)} \tag{3.31}$$

3.4.2 Fisher Score

The next approach for feature selection in the multi-class case is *Fisher Score.* The idea behind this method, described in [26], is to select features in a way that the inter-class distance in the feature space is as high as possible and the intra-class distance as small as possible.

This concept is computationally expensive for a high amount of features, thus, for every feature j, independent from the other features, the weight w_j is calculated separately by

$$w_j = \frac{\sum\limits_{k=1}^{K} n_k(\mu_k^j - \mu^j)^2}{\sum\limits_{k=1}^{K} n_k {\sigma_k^j}^2}. \tag{3.32}$$

The total number of classes is K and n_k indicates the amount of observations with class k. μ_k^j and σ_k^j define the mean value and the standard deviation, respectively, for the *k-th* class and the *j-th* feature, while μ^j is the mean value for all classes for the *j-th* feature.

Fisher Score calculates the weights for each feature independent of the others, therefore it is not possible to detect redundant features or features which are individually unimportant but would be important in combination with others [27].

3.4.3 Correlation-based Feature Selection (CFS)

This approach, introduced in [28], uses a correlation-based filter method to evaluate the importance of a feature or set of features regarding classification. It accounts both individual influence and also the correlation between separate features to detect redundancy. The assumption is, that relevant feature sets contain elements which strongly correlate with the classes but not within the set [28].
The heuristic which is used as a measure for the importance of a feature subset S can be expressed by

$$M_S = \frac{p\overline{r_{kf}}}{\sqrt{p + p(p-1)\overline{r_{ff}}}}. \tag{3.33}$$

M_S is the merit of the feature subset containing p elements with $\overline{r_{kf}}$ and $\overline{r_{ff}}$ being the mean correlation between the separate features and the class k ($f \in S$), and the feature-feature correlation, respectively. In his work [29], Hall compares different methods to calculate the correlations $\overline{r_{kf}}$ and $\overline{r_{ff}}$, namely minimum description length, relief and symmetrical uncertainty.
The numerator in (3.33) is a measure of how predictive S is of the class k whereas the denominator gives information on the redundancy of the features. Hall implemented three different search strategies to find the best feature set S, according to (3.33), which are forward selection, backward elimination and best first. The stopping criterion for each search is basically when adding a feature to/removing a feature from a subset doesn't improve/decreases the merit.
The result of CFS is the estimate of the best feature subset according to (3.33).

3.4.4 Decision Tree (DT)

The general concept of Classification and Regression Trees (CART) was introduced by Breiman et al. in [30]. The classification tree is a model that is used to predict the class label y of an observation $\mathbf{x}$ by passing through the tree from its root node to a leaf. The classification result for $\mathbf{x}$ is the class which the majority of the observations in the corresponding terminal node belongs to.
The model is constructed by splitting every non-terminal node of the tree into two nodes, starting with the root node which contains all observations of the training data. The feature on which the split is based, is chosen according to the impurity of the child nodes, that the corresponding split results in.
Impurity can be described as the degree of homogeneity of observations in a node, regarding their classes. A pure node only consists of instances of the same class, therefore the aim of the splitting process is to minimize the sum of impurities of the two child nodes [31] and, accordingly, the best feature is selected for the current split. Figure 3.3 shows a simple example for a decision tree with 3 classes of observations, for a single feature (color). For each non-terminal node, the best split is used.

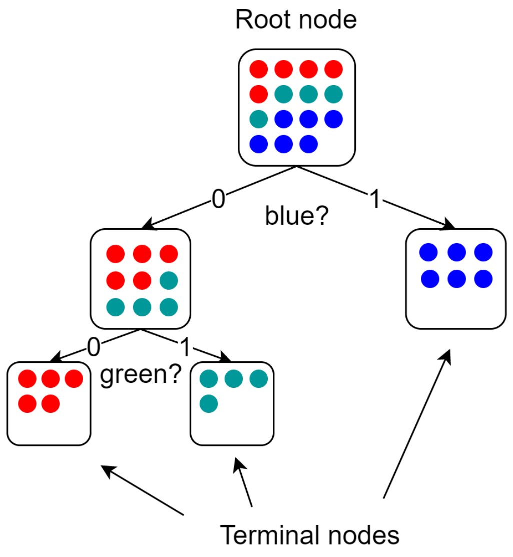


Figure 3.3. Example of a decision tree with observations (represented by circles) of 3 classes.

As a measure for impurity, the CART algorithm uses the Gini-Index G, proposed in [32]. It is calculated by

$$G_i = 1 - \sum_{j=1}^{K} p^2(j) \tag{3.34}$$

for the node i. $p(j)$ describes the ratio of elements from the class j to the total amount of observations n in the node. The tree is grown until there is no significant improvement regarding the impurity. If this is the case for a node, it is not split anymore and becomes a terminal node.
The importance of a single feature, in [30] referred to as variable importance, is determined by summing the impurity gain ΔG separately over all node splits for every feature:

$$\Delta G_i = G_i - \frac{n^l}{n} G_i^l - \frac{n^r}{n} G_i^r \tag{3.35}$$

G_i^l and G_i^r are the Gini-Indices for the left and right sibling node of i and n, n^l and n^r are the amount of observations in the split node and its left and right child, respectively.
As concluded in [31], it is also possible to use these decision trees for feature selection, even in high-dimensional space. The result of the feature selection is a ranking of the sums of the impurity gains over all splits where the highest value corresponds to the most important feature in the tree.

3.4.5 Principal Component Analysis (PCA)

The earliest works of the concept now known as Principal Component Analysis were carried out in [33] and [34] by Pearson and Hotelling, respectively.
PCA is a method to approximate a set of M features by a smaller set of $k \leq M$ linear combinations of these features, defined as principal components (PCs), as to reduce the dimensionality of the data. The assumption in this concept is, that the derived k variables can preserve most of the information regarding the variances of the m features [35]. In Figure 3.4, a simplified example is given, showing observations with two features f_1 and f_2. The fitted PC coordinate system indicates that most of the variance in the example data can be preserved by the first PC.

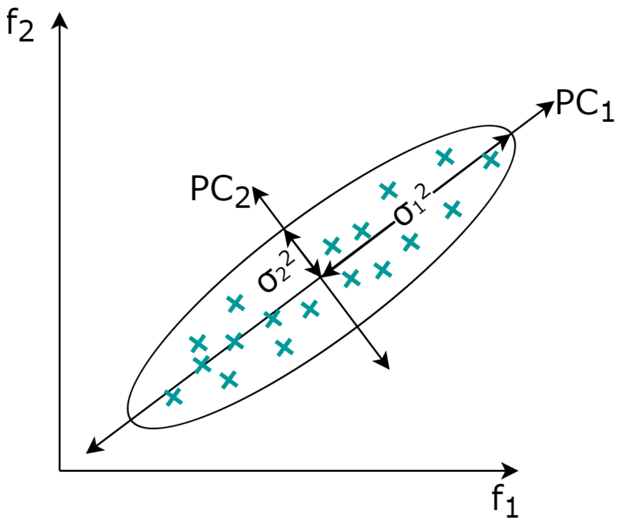


Figure 3.4. Plot with two-dimensional example data and fitted PC coordinate system.

Mathematically speaking, the original data $\mathbf{X} = [\mathbf{x}_1\, \mathbf{x}_2\, \dots\, \mathbf{x}_N]$, containing N observations, is projected onto the PC space, resulting in a new matrix $\mathbf{P} = [\mathbf{p}_1\, \mathbf{p}_2\, \dots\, \mathbf{p}_N]$.
The first step of this process [36] is the centering of the data in $\mathbf{X} \in \mathbb{R}^{M \times N}$ by subtracting the mean $\boldsymbol{\mu}$ for every feature over all observations with

$$\mathbf{X}_c = [\mathbf{x}_1 - \boldsymbol{\mu}\; \mathbf{x}_2 - \boldsymbol{\mu}\; \dots\; \mathbf{x}_N - \boldsymbol{\mu}], \tag{3.36}$$

where

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i. \tag{3.37}$$

Next, the covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{M \times M}$ of the centered data is computed by

$$\boldsymbol{\Sigma} = \frac{1}{N-1} \mathbf{X}_c \mathbf{X}_c^T, \tag{3.38}$$

where the element at position (i,j) in $\mathbf{\Sigma}$ is the covariance of the observations $\mathbf{x}_i$ and $\mathbf{x}_j$.
Following, the N eigenvectors $\mathbf{V} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \dots \ \mathbf{v}_M]$ and eigenvalues $\boldsymbol{\lambda} = [\lambda_1, \ \lambda_2, \ \dots, \ \lambda_M]^T$ of $\mathbf{\Sigma}$ are calculated from [37]

$$\mathbf{\Sigma}\mathbf{v}_i = \lambda_i \mathbf{v}_i. \tag{3.39}$$

According to Hotelling's definition of PCA in [34], the PCs are the orthonormal axes onto which the variance of the data is maximal. Thus, the eigenvectors, as they represent these orthonormal axes, are sorted according to their corresponding eigenvalues, as they correspond to the variances, starting with the highest value. The eigenvectors with the k largest eigenvalues are collected in the matrix $\mathbf{W} \in \mathbb{R}^{M \times k}$ which is used for the transformation from the original feature space to the lower dimensional PC space. With

$$\mathbf{P} = \mathbf{W}^T \mathbf{X}_c, \tag{3.40}$$

the original data is now represented in the PC space where $\mathbf{P} \in \mathbb{R}^{k \times N}$ contains the projected data.
Compared to the other approaches presented before, PCA doesn't select the most important elements from the feature space, but instead creates a new lower dimensional variable space. Therefore, the term feature selection is not really appropriate, but rather dimensionality reduction or feature extraction. Another difference is that PCA is an unsupervised method and thus has no knowledge of the classes of the samples.

3.5 Classification

In machine learning, the term classification refers to the task where the attributes or features of a sample are acquired (e. g. by measurements) and further processed, to identify the category or class to which the sample belongs. Typically, this means that the features corresponding to this sample i (which is often referred to as observation) are stored in a vector $\mathbf{x}_i$ and the aim of the process is basically to find the correct label y_j for the sample, where j is one out of a fixed set of discrete classes.
The classification algorithm itself can be thought of as a rule that takes $\mathbf{x}_i$ as an input and returns a class label. With the knowledge of the cost of mislabeling for each class, this rule is defined in a way that the expected mislabeling cost is minimized [38]. The realization of this rule is based on previously collected data that consists of a set of observations together with the acquired features and the correct labeling. This process of training the model from labeled data is also referred to as supervised learning. The classification model is fitted to this so called training data in a way that the above mentioned mislabeling cost is a minimum.
Nevertheless, if the model is evaluated with new observations - the test data - the classification error might be worse. This phenomenon is often called overfitting

which indicates that the model is biased to the training data [38]. To avoid this, a common approach is to validate the model during the learning process with a subset of the data which is intentionally left out during the training. Thus, the evaluation is done with data which is new to the model and the estimated performance is more likely to be similar to the test data.
An even better method to get a good estimate on the test performance, especially when the training set is sparse, is k-fold cross-validation. The training data is split into k parts which are about the same size. Then, the model is trained k-times with $k-1$ parts as training set and 1 part as validation set. The mean validation error over the k iterations is an effective estimate of the average test error [39].

The performance of a classifier can be measured with various methods. Probably the simplest, most obvious approach is the error rate which measures the percentage of wrong classifications or its counterpart, the accuracy, which indicates the correct predictions. In cases where the misclassification cost varies between the classes or where the dataset is imbalanced (if the amount of observations is not the same for every class), different performance measures achieve more appropriate results. An improvement over the usual accuracy is the recall which is computed equivalently to the average accuracy over all classes in the multi-class case:

$$recall_{mc} = \frac{1}{K} \sum_{j=1}^{K} \frac{1}{N_j} \sum_{n=1}^{N_j} s_j^n, \tag{3.41}$$

where

$$s_j^n = \begin{cases} 1, & \text{for a correct prediction} \\ 0, & \text{for a false prediction} \end{cases} . \tag{3.42}$$

N_j is the size of the test samples $\mathbf{s}_j$ for the class j and K is the amount of discrete classes.
Another performance measure, especially for more than two classes, is the confusion matrix. It is a table-like visualization where the rows correspond to the true labels of the observations and the columns correspond to the predicted classes. Thus, it gives hints on the general performance of the classifier and also which classes are often confused by the model [38]. A three-class example for a confusion matrix is shown in Figure 3.5, where the percentage of classifications is noted in every field. For instance, 16.2% of the samples from class 1 are falsely labeled with class 2.

3.5.1 Support Vector Machine (SVM)

A very useful tool in machine learning is the group of algorithms called Support Vector Machines. The basic idea of SVMs is to create a mathematical model that categorizes data from different classes by finding a hyperplane (e. g. a line in the case of two-dimensional space) in the feature space that best separates the data.

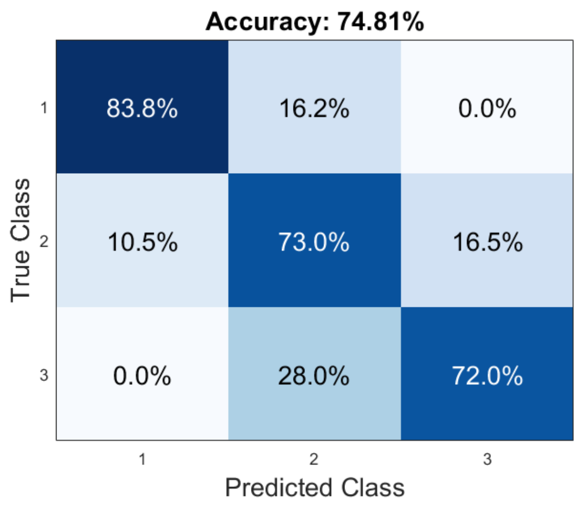


Figure 3.5. Example of a confusion matrix for a three-class problem.

Assuming a two-class problem for simplicity, the data of N observations is given by N features vectors $\mathbf{x}_i$, where each vector contains M elements. The corresponding class labels are given by y_i where $y_i \in \{-1{,}1\}$. The separating margin can now be defined by the equation [39]

$$f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta} + \beta_0 = 0 \tag{3.43}$$

with the vector $\boldsymbol{\beta}$ and the scalar β_0 as parameters. The function $f(\mathbf{x}_i)$ calculates the signed distance from the point $\mathbf{x}_i$ in feature space to the hyperplane defined by (3.43). Hence, the rule which gives the classification result can be written as

$$\tilde{y}(\mathbf{x}) = sign(\mathbf{x}^T\boldsymbol{\beta} + \beta_0). \tag{3.44}$$

For data with separable classes, the hyperplane parameters $\boldsymbol{\beta}$ and β_0 can be found in a way that $y_i f(\mathbf{x}_i) > 0 \ \forall i$. In [39], the optimization problem for finding the margin which separates the classes -1 and 1 best, can be expressed by

$$\begin{gathered}\min_{\boldsymbol{\beta},\beta_0} ||\boldsymbol{\beta}|| \\ \text{for } y_i(\mathbf{x}_i^T\boldsymbol{\beta} + \beta_0) \geq 1,\ i = 1,\ldots,n.\end{gathered} \tag{3.45}$$

The points for which the equality in (3.45) is true are called the support vectors and, in Figure 3.6, lie on the dashed lines with distance $M = \frac{1}{||\boldsymbol{\beta}||}$ to the hyperplane.

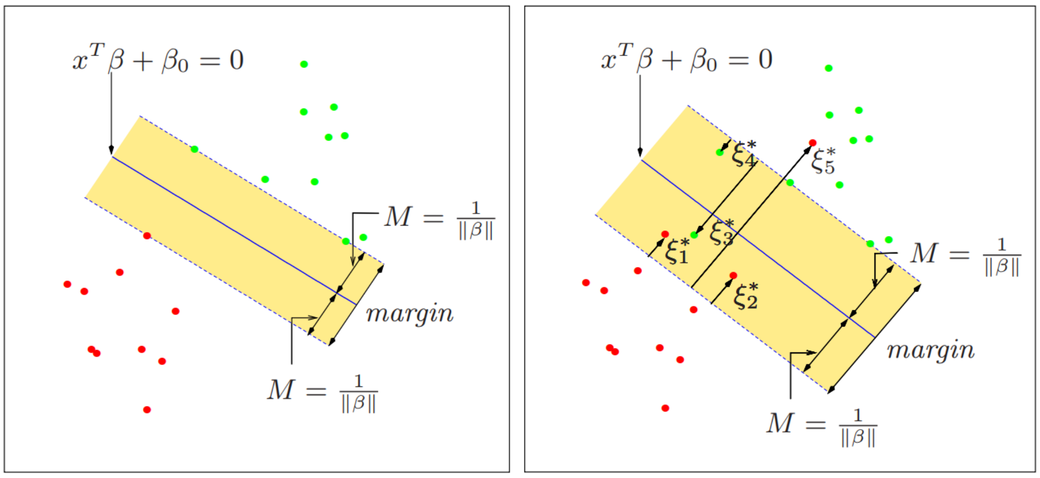


Figure 3.6. Example for an SVM in the two-dimensional case, on the left for separable and on the right side for non-separable data. Source: [39]

The previous concept works for separable data, however, in the case that the data overlaps for different classes, the model has to be adapted. This situation is depicted in the right image in Figure 3.6. The idea is now, to allow some observations on the wrong side of the hyperplane by defining so called slack variables ξ_i. This parameter defines the proportional amount by which the corresponding classification result $\tilde{y}_i(\mathbf{x}_i)$ is on the wrong side of the margin [39]. The sum of the slack variables $\sum\limits_{i=1}^{N} \xi_i$ is used as a boundary condition in the newly formulated optimization problem

$$\min_{\boldsymbol{\beta},\beta_0} \|\boldsymbol{\beta}\| \text{ for } \begin{cases} y_i(\mathbf{x}_i^T\boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i \,\forall i \\ \xi_i \geq 0, \sum\limits_{i=1}^{N} \xi_i \leq \text{constant} \end{cases} . \tag{3.46}$$

So far, linear margins for the features were described. In many cases, hyperplanes are not sufficient to separate the data, hence, more complex - non-linear - geometries are necessary. The original feature vectors are transformed into higher dimensional space, where a linear separation is possible. With the transformation expressed by $h(\mathbf{x})$, the now non-linear function is defined as $\hat{f}(\mathbf{x}) = h(\mathbf{x})^T\hat{\boldsymbol{\beta}} + \hat{\beta}_0$ with the corresponding classifier and the optimization problem similar to (3.44) and (3.46), respectively.
To deal with the computational complexity that this non-linear separation may require for very high-dimensions, the so called kernel trick is applied [40]. During the solving of the optimization problem, the term $\langle h(\mathbf{x}_i), h(\mathbf{x}_j)\rangle$ has to be computed for every pair of vectors. The trick is that now for particular functions (called kernels), the value for two feature vectors is the inner product of the transformed vectors and can be expressed with

$$K(\mathbf{x}_i,\mathbf{x}_j) = \langle h(\mathbf{x}_i), h(\mathbf{x}_j)\rangle. \tag{3.47}$$

The most common kernels for SVMs are [39]

- Polynomial kernel (*d-th* order): $K(\mathbf{x}_i,\mathbf{x}_j) = (1 + \langle \mathbf{x}_i,\mathbf{x}_j \rangle)^d$
- Gaussian kernel (Radial basis) : $e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}$
- Neural network (Sigmoid): $\tanh(\kappa_1 \langle \mathbf{x}_i,\mathbf{x}_j \rangle + \kappa_2)$

The basic concept of SVMs is binary classification. As in practice the number of classes K is often greater than two, multi-class models have to be used. The simplest method is to reduce the multi-class to a binary problem by creating K separate SVMs where each one separates the K-*th* class from the other $K-1$ classes. This approach is called the *one-vs-all* method [41]. The rule for the multi-class model is then expressed by

$$\tilde{y}(\mathbf{x}_i) = \max_K \tilde{y}_K(\mathbf{x}_i). \tag{3.48}$$

In the case, that several binary classifiers produce the same result, i. e. predict that $\mathbf{x}_i$ is of the corresponding class, often additional information, like the prediction confidence, is used to determine the actual class [41].

3.5.2 Random Forest (RF)

The concept of CARTs for feature selection was presented earlier in Section 3.4.4. As the term suggests, the method is intended for classification in learning applications. In this section, the usage of CARTs in an ensemble method is described, namely *Random Forests*, introduced by Breiman in [23]. Ensemble learning in general refers to a set of classifiers that are trained simultaneously. The overall prediction for an observation is then based on the majority vote or the averaging of the results for all learners.
A random forest takes up this concept, as it is a collection of decision trees, with the goal to improve the classification accuracy compared to the single classifier approach. This improvement is achieved with a modification of a technique called *bagging* which aims to reduce the variance of prediction results in ensemble learning, by using different subsets of the training samples for each classifier [39].
This concept is extended by the random forest approach which, in addition to varying samples for each tree, randomly selects a subset with $r \leq M$ of the input features for each split, with a typical value for r being $\sqrt{M}$.
As a result of this approach, the correlation between the individual trees is reduced, which improves the performance of the classifier [39].
An important advantage of random forests is the use of out-of-bag samples for classification. The label y_i for the observation $\mathbf{x}_i$ is the result of averaging only those trees for which $\mathbf{x}_i$ is not used for training. The estimate for the prediction error is similar to the result for k-fold cross-validation, hence, there is no need for additional validation of the model [39].

3.5.3 Artificial Neural Network (ANN)

Inspired by simplified models of the neural network structure of the human brain, Artificial Neural Networks are methods used in a wide range of learning applications. They can be graphically described as directed graphs, with nodes corresponding to neurons and edges linking them. The exact structure depends on the type of the ANN. Figure 3.7 shows an example for a common category of neural networks, called the *feed-forward* network.

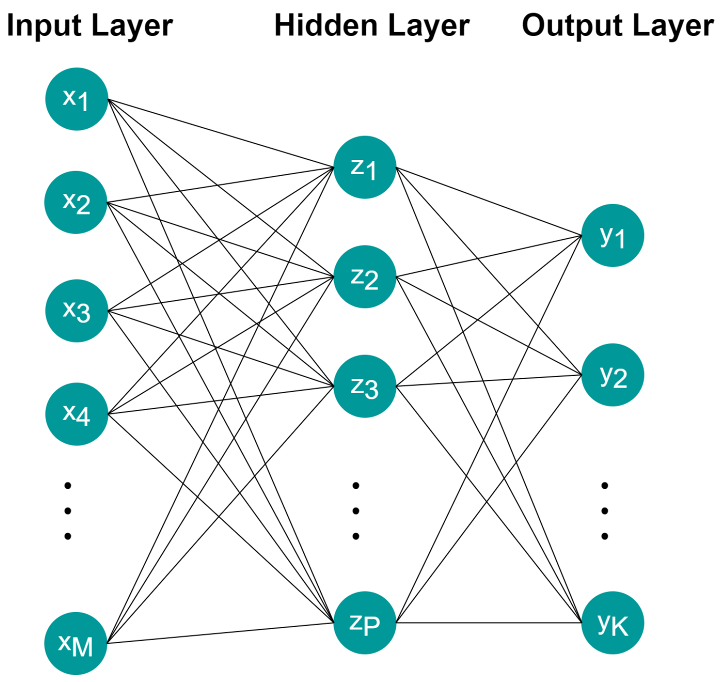


Figure 3.7. Structure of a feed-forward neural network with a single hidden layer.

In this structure, the input layer is connected to the output layer through at least one hidden layer, where the output of each node is linked to the input of every node in the following layer. Mathematically, the input of a neuron can be described by a linear combination of the connected outputs of the previous layer. This relation can be expressed by

$$z_m = \sigma(\alpha_{0p} + \boldsymbol{\alpha}_p^T \mathbf{x}), p = 1, \dots, P, \tag{3.49}$$

$$t_k = \beta_{0k} + \boldsymbol{\beta}_k^T \mathbf{z}, k = 1, \dots, K, \tag{3.50}$$

$$f_k(\mathbf{x}) = g_k(\mathbf{t}), k = 1, \dots, K, \tag{3.51}$$

with $\mathbf{z} = (z_1, z_2, \dots, z_M)$ as the output of the M neurons in the hidden layer and $\mathbf{t} = (t_1, t_2, \dots, t_K)$ as the input for the K neurons in the output layer [39]. The number of neurons in the hidden layer depends on the application and typically is somewhere between the size of the input layer (number of features M) and the output layer (number of classes K). $\sigma(v)$ is called the activation function

where common choices are the *sigmoid* function $\sigma(v) = 1/(1+e^{-v})$ or the *rectifier* function $\sigma(v) = max(0,v)$. Through choosing a nonlinear transformation function for σ, the neural network basically models a nonlinear function from the input variables $\mathbf{x}$ to the output variables $\mathbf{y}$ and thus can be used for complex scenarios. The vectors $\boldsymbol{\alpha}_p$ and $\boldsymbol{\beta}_k$ contain the so called weights for the corresponding edges. In addition to the neurons depicted in Figure 3.7, optional *bias* units can be inserted into the model, which can be thought of as neurons with the output value 1 and a weighting parameter, here α_{0p} and β_{0k}.
For classification purposes, the output of the hidden layer is usually transformed with the *softmax* function

$$g_c(\mathbf{t}) = \frac{e^{\mathbf{t}_k}}{\sum\limits_{\ell=1}^{K} e^{\mathbf{t}_\ell}}. \tag{3.52}$$

The results for each neuron in the output layer sums up to one, where the highest value is the estimate for the class label y_i of the model input $\mathbf{x}_i$.

During the learning or fitting process of ANNs, the set of weights and biases $\boldsymbol{\theta}$ is adapted, in order to fit the model to the training data. The typical approach to determine the parameters, desribed in [39], is called *back-propagation*, where, by using gradient descent optimization, a loss function in the form of the squared error

$$R(\boldsymbol{\theta}) \equiv \sum_{i=1}^{N} R_i = \sum_{k=1}^{K}\sum_{i=1}^{N}(y_{ik} - f_k(x_i))^2 \tag{3.53}$$

is minimized. With $\mathbf{z}_{pi} = \sigma(\alpha_{0p} + \boldsymbol{\alpha}_p^T x_i)$ from (3.49) and $\mathbf{z}_{pi} = (z_{1i}, z_{2i}, \ldots, z_{Pi})$, the partial derivatives of R_i are expressed by

$$\frac{\partial R_i}{\partial \beta_{kp}} = -2(y_{ik} - f_k(x_i))g_k'(\beta_k^T z_i)z_{pi} = \delta_{ki} z_{pi}, \tag{3.54}$$

$$\frac{\partial R_i}{\partial \alpha_{p\ell}} = -\sum_{k=1}^{K} 2(y_{ik} - f_k(x_i))g_k'(\beta_k^T z_i)\beta_{kp}\sigma'(\alpha_p^T x_i)x_{i\ell} = s_{pi} x_{i\ell}. \tag{3.55}$$

The values δ_{ki} and s_{pi} can be referred to as the errors of the output and hidden layer calculated from the current model. With the relation

$$s_{pi} = \sigma'(\alpha_p^T x_i) \sum_{k=1}^{K} \beta_{kp}\delta_{ki}, \tag{3.56}$$

the weights and biases are updated by repeating two steps:

- The *forward pass*, where the training data is classified with (3.49) - (3.51), using the current weights,
- and the *backward pass*, where the values for δ_{ki} are calculated and afterwards back-propagated to s_{pi} using (3.56)

The errors are then used for the computation of the gradients in the equations for the parameter updates

$$\beta_{kp}^{(r+1)} = \beta_{kp}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{kp}^{(r)}} \tag{3.57}$$

$$\alpha_{p\ell}^{(r+1)} = \alpha_{p\ell}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{p\ell}^{(r)}}, \tag{3.58}$$

with the learning rate γ_r which defines how much the weights are changed in each step according to the gradients.
To avoid overfitting of the model to the training data, the learning process is stopped before the global minimum is reached [39].

3.5.4 Synthetic Minority Over-sampling Technique (SMOTE)

In [42], Chawla et al. propose an effective method to deal with dataset imbalance in machine learning applications. This imbalance is often caused by the nature of the problem, e. g. in defect detection, where in most of the cases the “normal” elements without defects dominate over the faulty samples. Thus, the simple assumption that no observation is defective, leads to a seemingly good classifier performance, although all the defect samples are falsely classified.
To deal with this problem, the above mentioned paper describes a method called *Synthetic Minority Over-sampling Technique*, where the idea is to create “synthetic” observations for the underrepresented class(es). This is done by randomly generating new samples which lie between existing instances and their neighbors in feature space.
First, the k nearest minority class neighbors in the feature space are found for every instance $\mathbf{x}_i$ of the corresponding class. Depending on the required amount of oversampling, an appropriate set out of the k neighboring instances is chosen randomly for every sample and for every element $\mathbf{x}_j$ of this set, the distance vector $\mathbf{d} = (\mathbf{x}_i - \mathbf{x}_j)$ to $\mathbf{x}_i$ is calculated over all features. The “synthetic” sample $\tilde{\mathbf{x}}_j$ is the result of the sum of $\mathbf{x}_j$ and the distance $\mathbf{d}$ weighted with a random value r between 0 and 1. Mathematically this can be expressed by

$$\tilde{\mathbf{x}}_j = \mathbf{x}_j + r\mathbf{d} = \mathbf{x}_j + r(\mathbf{x}_i - \mathbf{x}_j). \tag{3.59}$$

4 Extraction of Surface Features

The following chapter describes the concepts for the steps of the feature extraction, including the pre-processing of the borehole images explained in detail, the image stitching, the following image segmentation and also the photometric stereo approach. In addition, the datasets, which are the basis of this work, are discussed. Finally, the ideas and mathematical formulations for the extraction of the different features is presented. Figure 4.1 shows an overview and a short description of the individual steps.

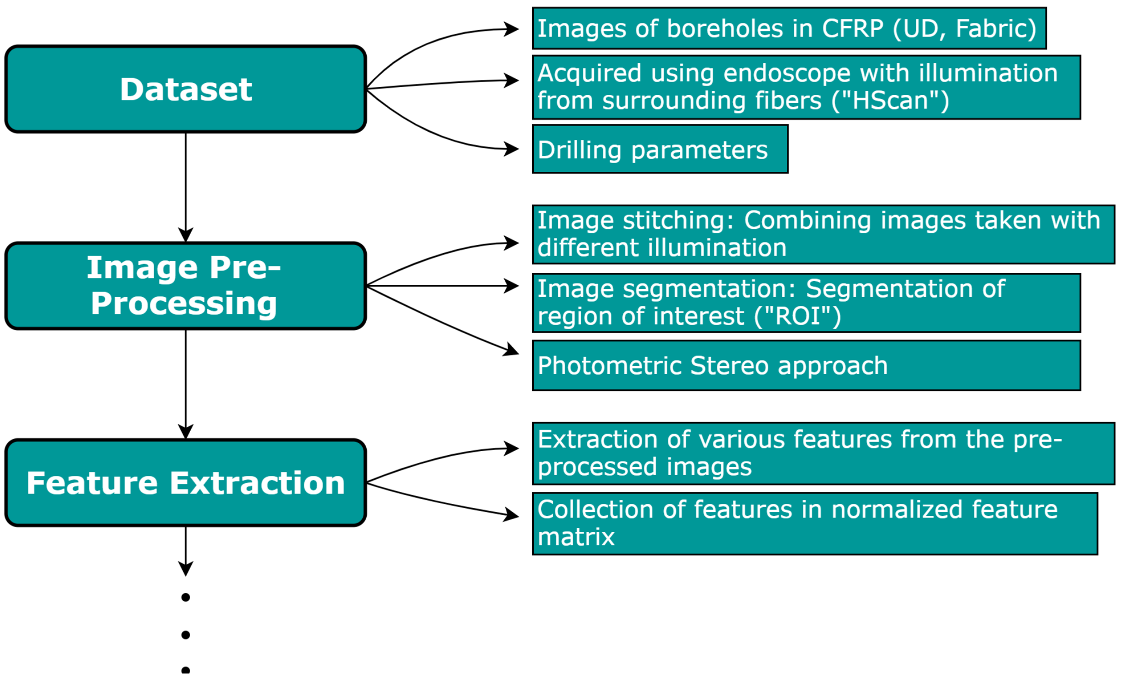

Dataset
Images of boreholes in CFRP (UD, Fabric)
Acquired using endoscope with illumination from surrounding fibers ("HScan")
Drilling parameters
Image Pre-Processing
Image stitching: Combining images taken with different illumination
Image segmentation: Segmentation of region of interest ("ROI")
Photometric Stereo approach
Feature Extraction
Extraction of various features from the pre-processed images
Collection of features in normalized feature matrix

Figure 4.1. Graphical overview of the feature extraction process.

4.1 Dataset

4.1.1 Borehole Images

The first part of this dataset, provided by the company Profactor [1], consists of grayscale images of the insides of boreholes (⌀ ∼6mm) in CFRP parts. Prior to this work, pictures of 72 boreholes in uni-directional material (UD) and 72 holes in multi-directional material (Fabric) were acquired by a special endoscope with illumination from optical fibers, where each one is connected to an LED. In total, there are 8 fibers, which are distributed equally around the endoscope, to enable illumination of the borehole from all angles.
To be able to capture the whole inside of the hole, the endoscope is moved step by step along the z-axis and for each of the 13 steps, 8 High Dynamic Range (HDR) images are taken, where for each image, only the corresponding fiber is active. As a result, each of the 8 images per acquisition step shows 360° (normal to the z-axis) of the inside of the borehole with the only difference being the illuminated area. Thus, the position of a pixel corresponding to a point on the surface is the same for all images per step, so the different illuminated parts can be easily stitched together without complex registration methods. This stitching is discussed in the next section. An example of the imaging process is shown in Figure 1.1.
The sensor unit, which was used to create the dataset, is called HScan [3] and was designed by Profactor for borehole inspection. Figure 1.2 pictures the principal setup of the endoscope and the arrangement of the illumination.

4.1.2 Drilling Parameters

The second part of the dataset consists of the parameters of the drilling process for each borehole, like the wear of the tool, or the applied feed. Based on these parameters, the holes were manually classified in three categories:

1: Optimal (Fabric: 14 samples, UD: 14 samples)

2: Bad drilling (Fabric: 38 samples, UD: 21 samples)

3: Pushed through (Fabric: 20 samples, UD: 37 samples)

This separation was done in advance to this thesis and is later used as groundtruth for the classification process.

4.2 Pre-Processing

To extract surface features and other data from the borehole images, it is useful to process them in advance, so e. g. to combine the different illuminated regions of the inside of the hole to get a more or less fully illuminated image for every acquisition step.

4.2.1 Image Stitching

The combination of the partial images with different lighting was implemented in two different ways, which are used depending on the desired information. In the following chapters, the first one is called *Maximum-based* and the second one *Region-based*. As a starting point for the pre-processing, the third acquisition step (*step2*) was used, as the entire inside of the borehole is visible and not occluded by the imaging geometry, like in other steps.

Maximum-based

This method compares all the images per step along the z-axis and the intensity value of the resulting image corresponds to the maximum value in the partial images. This is possible due to the matching pixel position through the 8 images, as already mentioned before. Figure 4.2 shows an example of maximum-based image stitching.

Region-based

The idea behind this method is to combine the most illuminated region from every partial image to a final, fully illuminated picture. For this approach, only 6 of the initially 8 images per step were used, as the illumination in the first and last picture for every step is worse than for the remaining 6 which provide sufficient information for the stitching. The result is shown in Figure 4.3, with an annotation of the corresponding image used for each region.

Comparison of methods

The results for the maximum-based and region-based stitching approaches are mostly similar. Nevertheless, for the second method, the edges between the image regions are visible, which can lead to problems when extracting certain features, like edge detection for example. Another difference in the approaches is the blending of images in the first method. This means that the reflections on the hole surface for every region of the image originate from different illumination angles, which might cause errors, for example with the photometric stereo approach. Therefore, both methods have different use cases, as shown in the following sections.

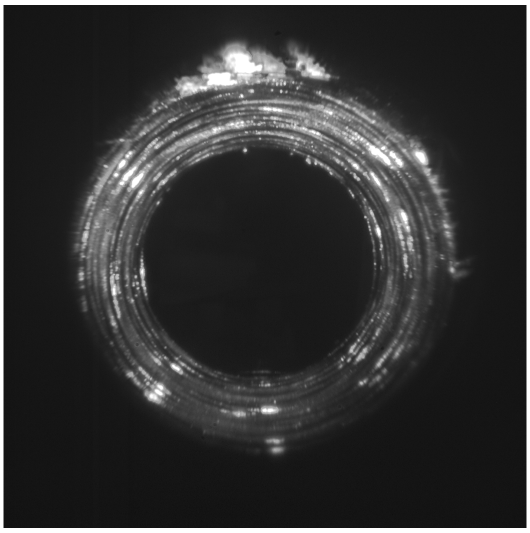


Figure 4.2. Maximum-based image stitching.

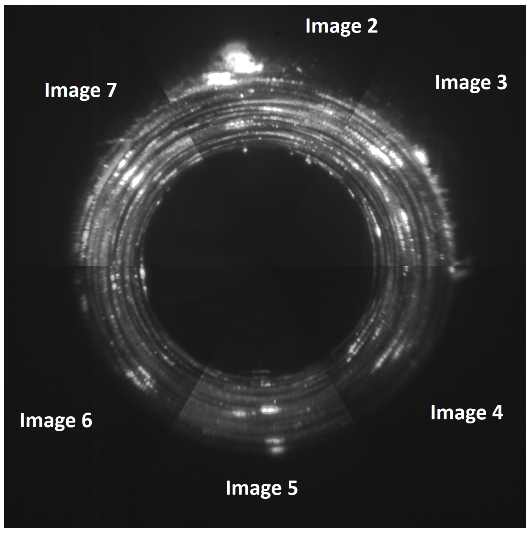


Figure 4.3. Region-based image stitching.

4.2.2 Image Segmentation

The type of pre-processing depends on the desired image information. If for example only the inside-surface of the borehole or the fiber pullout of the material is of interest, the picture has to be segmented accordingly. A graphical overview on the following segmentation process is given in Figure 4.4.

Approach

Assuming the endoscope centered in the borehole, it should be possible to approximate the hole by using two roughly concentric circles which originate from the inlet opening (outer circle) and the outlet opening (inner circle). Starting from the image center, the nearest pixels in radial direction determine the first circle, the inner circle, assuming that the image center is inside this circle. For the outer circle, radiant again from the image center, the farthest pixels are used. Therefore, with the identification of these two circles, the image can be segmented in three parts:

- The inside-surface of the borehole (between the circles)
- The interior of the borehole (inside of the inner circle)
- The surface of the CFRP part (outside of the outer circle)

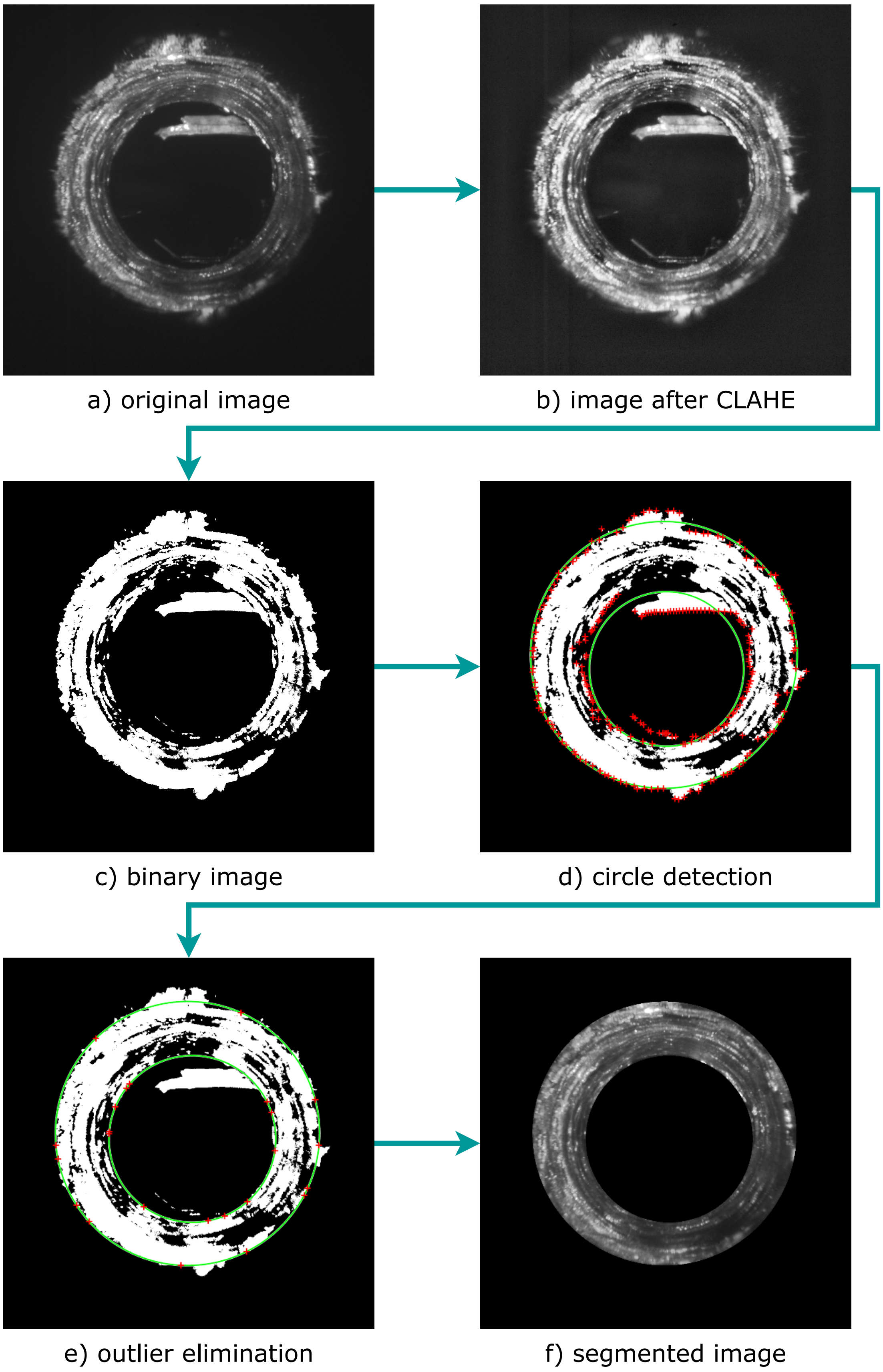


Figure 4.4. Steps of the segmentation process.

Implementation

To be able to detect the points of the borehole inside, beginning with the image in Figure 4.4 a), the picture is transformed to a binary mask (Fig. 4.4 c) with the MATLAB function *imbinarize*, in order to segment the inside surface of the borehole from the rest of the image. The intensity value of pixels under a certain threshold are set to 1 and the remaining pixels to 0. This threshold t is calculated with Otsu's Method [43], in a way that the intra-class variance $\sigma_w^2(t)$, defined as the weighted sum of both white and black pixels, is minimized. With the intensity variances $\sigma^2(t)$ and the class probabilities $\omega(t)$ (number of pixels per class divided by total amount of pixels), the following function is minimized:

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{4.1}$$

This is done iteratively, by computing the result for each value of t, which corresponds to 256 iterations in an 8-bit image.
If the intensity values of the borehole inside surface are too inhomogeneous, for example if some pixels are very bright compared to others, the transformation to the binary mask can result in black areas that ideally should be white, as their intensity value is under the threshold. To deal with this problem, an intermediate step is introduced (Fig. 4.4 b), where previous to the binary transformation, the image's intensity histogram is adapted. This is done using the function *adapthisteq*, which implements the *Contrast Limited Adaptive Histograph Equalization* (CLAHE) [44]. The idea of this method is to compute several intensity histograms for each section of the image, to improve the local contrast by spreading the most frequent intensity values.
Next, using the binary image, the closest and farthest points in each radial direction is determined. The number of directions n is pre-defined and equally distributed along the angle. A suitable value for n is around 60-120. In order to identify the desired points, the absolute image coordinates $\mathbf{p}_i^a$ for every pixel i are transformed (4.2) to coordinates relative to the image center ($\mathbf{p}_i^r$), using the homogeneous transformation matrix $\mathbf{T}$, with W and H as the image width and height, respectively.

$$\begin{gathered}\mathbf{p}_i^r = \mathbf{T}\,\mathbf{p}_i^a \\ \begin{bmatrix} x_i^r \\ y_i^r \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{W}{2} \\ 0 & -1 & \frac{H}{2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i^a \\ y_i^a \\ 1 \end{bmatrix}\end{gathered} \tag{4.2}$$

As a starting point, the standard form of the circle equation (4.3), with x_c and y_c as the relative coordinates of the circle center, is used:

$$(x_i - x_c)^2 + (y_i - y_c)^2 = r^2 \tag{4.3}$$
$$x_i^2 - 2x_i x_c + x_c^2 + y_i^2 - 2y_i y_c + y_c^2 = r^2 \tag{4.4}$$
$$2x_i x_c + 2y_i y_c + r^2 - x_c^2 - y_c^2 = x_i^2 + y_i^2 \tag{4.5}$$

The last equation can be formulated in matrix notation as

$$\begin{bmatrix} x_i & y_i & 1 \end{bmatrix} \begin{bmatrix} 2x_c \\ 2y_c \\ r^2 - x_c^2 - y_c^2 \end{bmatrix} = \begin{bmatrix} x_i^2 + y_i^2 \end{bmatrix} \tag{4.6}$$

Now, for n points per circle, the result is an overdetermined system of linear equations in the form of (4.6):

$$\underbrace{\begin{bmatrix} y_1 & x_1 & 1 \\ y_2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ y_n & x_n & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} 2x_c \\ 2y_c \\ r^2 - x_c^2 - y_c^2 \end{bmatrix}}_{\mathbf{p}} = \underbrace{\begin{bmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 \\ \vdots \\ x_n^2 + y_n^2 \end{bmatrix}}_{\mathbf{b}} \tag{4.7}$$

With the full rank matrix $\mathbf{A}$, there is no exact solution for

$$\mathbf{Ap} = \mathbf{b}. \tag{4.8}$$

The least squares approach, presented in Section 3.1, computes an optimal solution $\hat{\mathbf{p}}$ for this problem:

$$\hat{\mathbf{p}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}. \tag{4.9}$$

In MATLAB, this approach is implemented with the backslash operator:

$$\hat{\mathbf{p}} = \mathbf{A}\backslash\mathbf{b}. \tag{4.10}$$

The resulting parameters - the coordinates of the circle center $\hat{x}_c$, $\hat{y}_c$ and the radius $\hat{r}$ - can be extracted from $\hat{\mathbf{p}}$ as solutions of the system of equations

$$\hat{\mathbf{p}} = \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \hat{p}_3 \end{bmatrix} = \begin{bmatrix} 2\,\hat{y}_c \\ 2\,\hat{x}_c \\ \hat{r}^2 - \hat{x}_c^2 - \hat{y}_c^2 \end{bmatrix}. \tag{4.11}$$

After this step, the optimal solution, in a mathematical sense, for both inner and outer circle is available in standard form, depicted in Figure 4.4 d). As (4.7) is an overdetermined system, the detected points don't fit the calculated circles exactly. The openings are not perfectly smooth, especially for lower quality boreholes. Therefore, the circles are improved by eliminating outliers which originate for example from surface reflections on the outside of the borehole or delaminated material in the inner circle. In every step, the most distant point from the circle is eliminated and the remaining points are again least squares fitted to a new circle. This procedure is repeated until only a pre-specified number of points is left (minimum 3). The segmentation result with the final circles (Fig. 4.4 e)) is shown in Figure 4.4 f).

4.2.3 Photometric Stereo

In order to not only rely on features extracted directly from the grayscale intensity values, an idea is to get 3-dimensional knowledge about the inside surface of the borehole to be able to detect surface defects. This is implemented using a photometric stereo approach, as to calculate the surface normals of the inside. The surface normal vectors are then used as a starting point for calculating various features.

Approach

The 8 images per borehole and vertical step are taken from the same position, with the single difference, that for every image a different glassfiber (LED) is active and therefore the illumination angle changes. For the photometric stereo approach, a minimum of three illumination directions per scene is needed, as to calculate the normal vectors at a certain point from the incident angle-dependent reflection of the material. As a consequence, the three most suitable images for every region are chosen for the computations.
As the implementation of an optimal photometric stereo approach is difficult, especially due to the small geometries, the imaging system is assumed to be ideal and the material is assumed to reflect completely diffuse, which doesn't correspond to reality, however, is necessary for the ease of implementation.

Implementation

The calculation of the surface normals is implemented in two slightly different ways, due to manufacturing variances regarding the glassfibers. To begin with, their total brightness varies, as well as the diameter. In addition, the cutting angle at the fiber ends is not exactly the same, therefore the illumination angles are not consistent. For these reasons, the first implementation (Figure 4.6) takes only the 6 best illuminated images per step into account, whereas the second implementation (Figure 4.7) uses all 8 images.
Figure 4.5 shows the fiber placement around the endoscope with the consecutive numbering. For the first implementation, the images from the fibers 1 and 8 are not taken into account due to their lacking illumination.

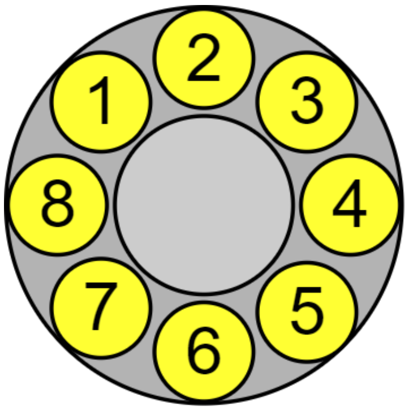


Figure 4.5. Placement of the fibers around the endoscope.

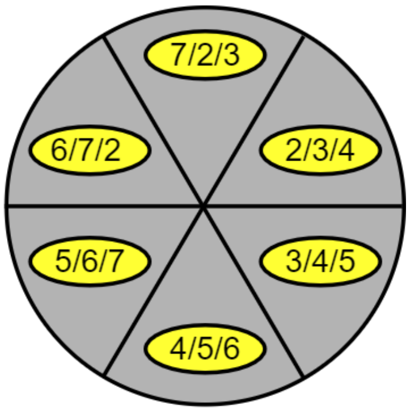


Figure 4.6. Fiber utilization for the 1st implementation.

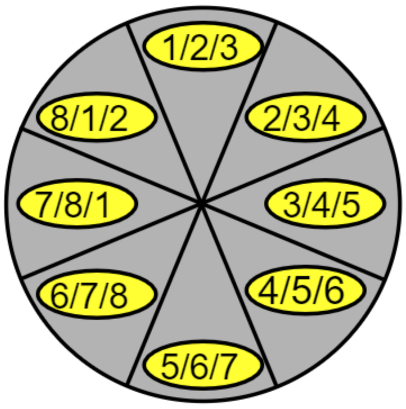


Figure 4.7. Fiber utilization for the 2nd implementation.

Intensity equalization

As mentioned before, the brightnesses of the fibers differ from each other. Figure 4.8 to Figure 4.15 illustrate the varying illuminations.
To cope with this, an intensity equalization factor (F_{eq}) is calculated for both photometric stereo implementations for each of the 6 or 8 regions, respectively. The mean intensity $\bar{I}$ over all n pixels of the borehole inside for the *step2* image j is averaged over all m *step2* images with the same illumination, separately for each single region, dataset and implementation.

$$\bar{I}_j = \frac{1}{n} \sum_{i=1}^{n} I_i \tag{4.12}$$

$$\bar{I}_{tot} = \frac{1}{m} \sum_{j=1}^{m} \bar{I}_j \tag{4.13}$$

Next, the highest value over all regions $\bar{I}_{tot}^{max}$ is multiplied with the reciprocal value for each $\bar{I}_{tot}$, which results in the F_{eq}, again for each single region, dataset and implementation. Finally, the corresponding factor is applied to each image matrix, to adapt the pixel intensities **I**:

$$F_{eq} = \frac{\bar{I}_{tot}^{max}}{\bar{I}_{tot}} \tag{4.14}$$

$$\mathbf{I}_{eq} = F_{eq}\,\mathbf{I} \tag{4.15}$$

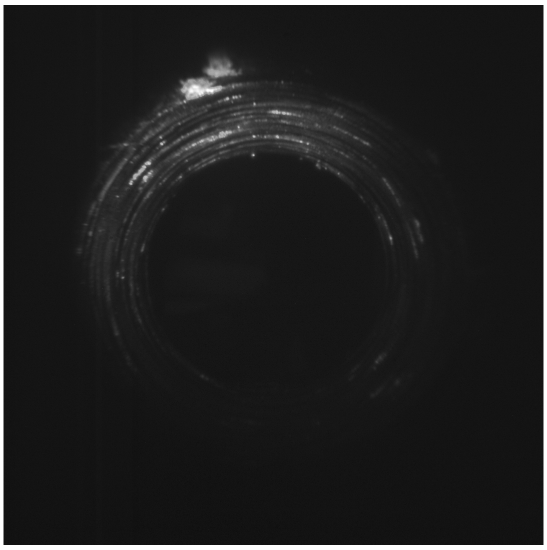


Figure 4.8. Illumination with fiber 1.

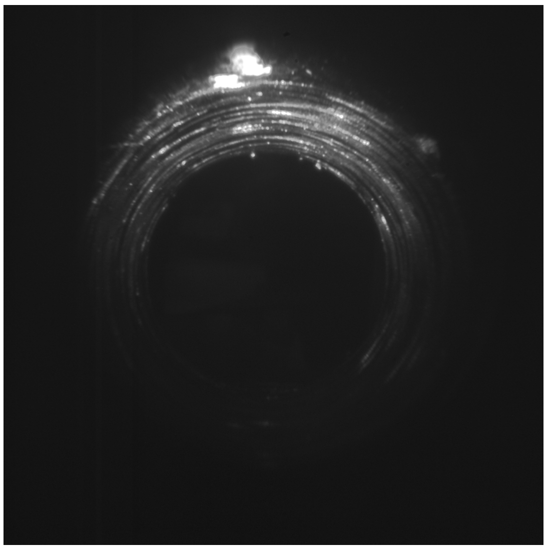


Figure 4.9. Illumination with fiber 2.

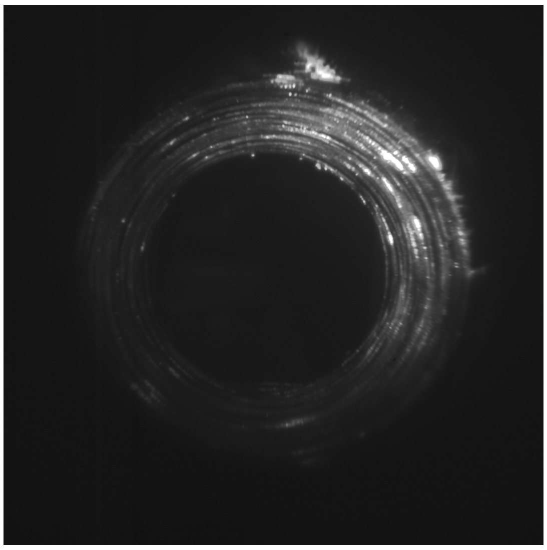


Figure 4.10. Illumination with fiber 3.

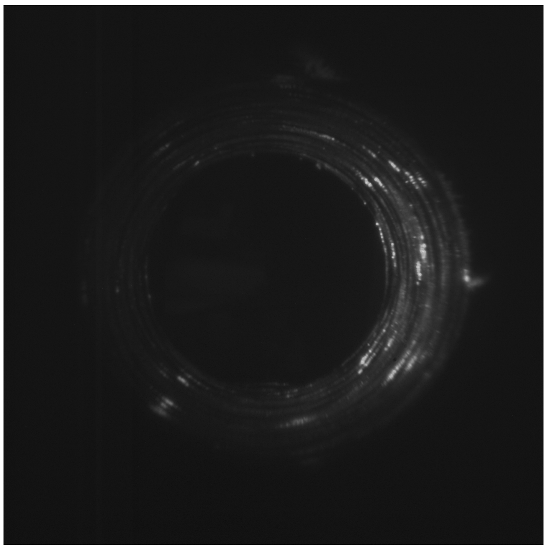


Figure 4.11. Illumination with fiber 4.

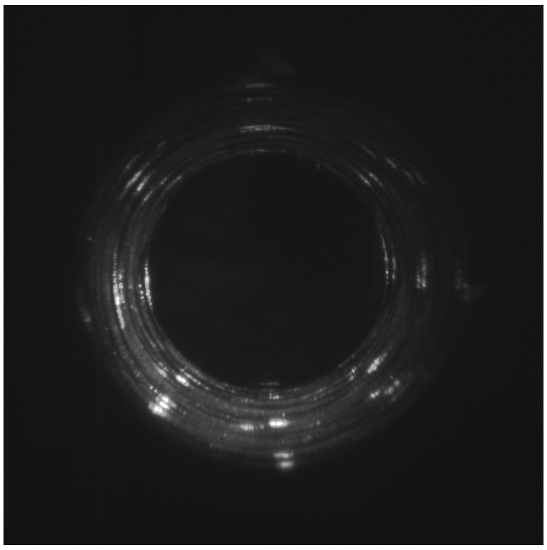


Figure 4.12. Illumination with fiber 5.

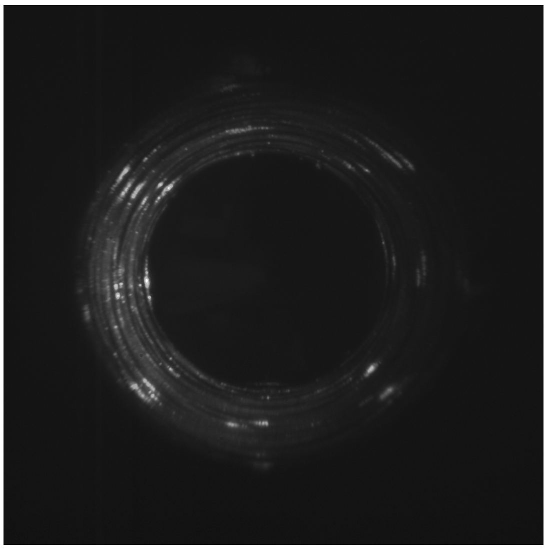


Figure 4.13. Illumination with fiber 6.

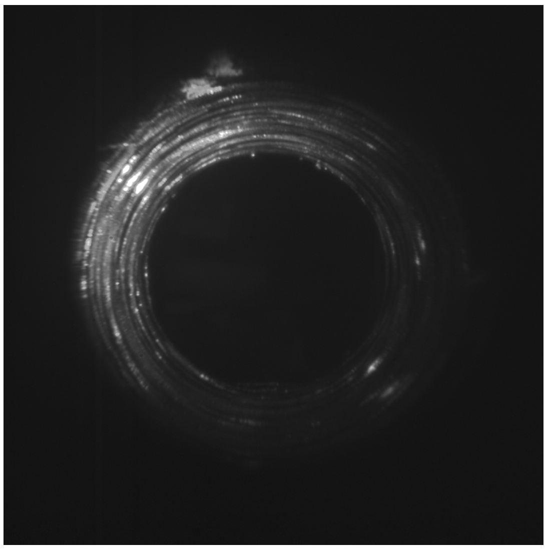


Figure 4.14. Illumination with fiber 7.

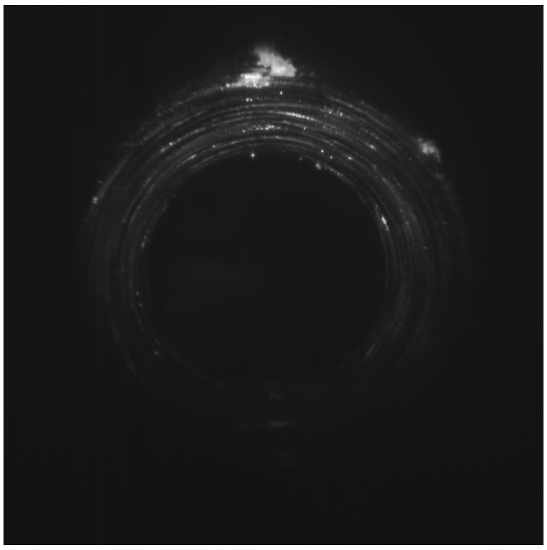


Figure 4.15. Illumination with fiber 8.

Calculation of surface normals

The ideal parameters for the imaging geometry, like camera position, distribution of the glassfibers around the endoscope and borehole dimensions are known. Therefore, as already mentioned, the surface normals can be computed from three images with different illumination direction.

Figure 4.16 shows the geometry of the imaging system and the borehole, respectively. Moreover, it depicts an example on how three images, necessary for the photometric stereo approach, are chosen for a certain point in the scene.

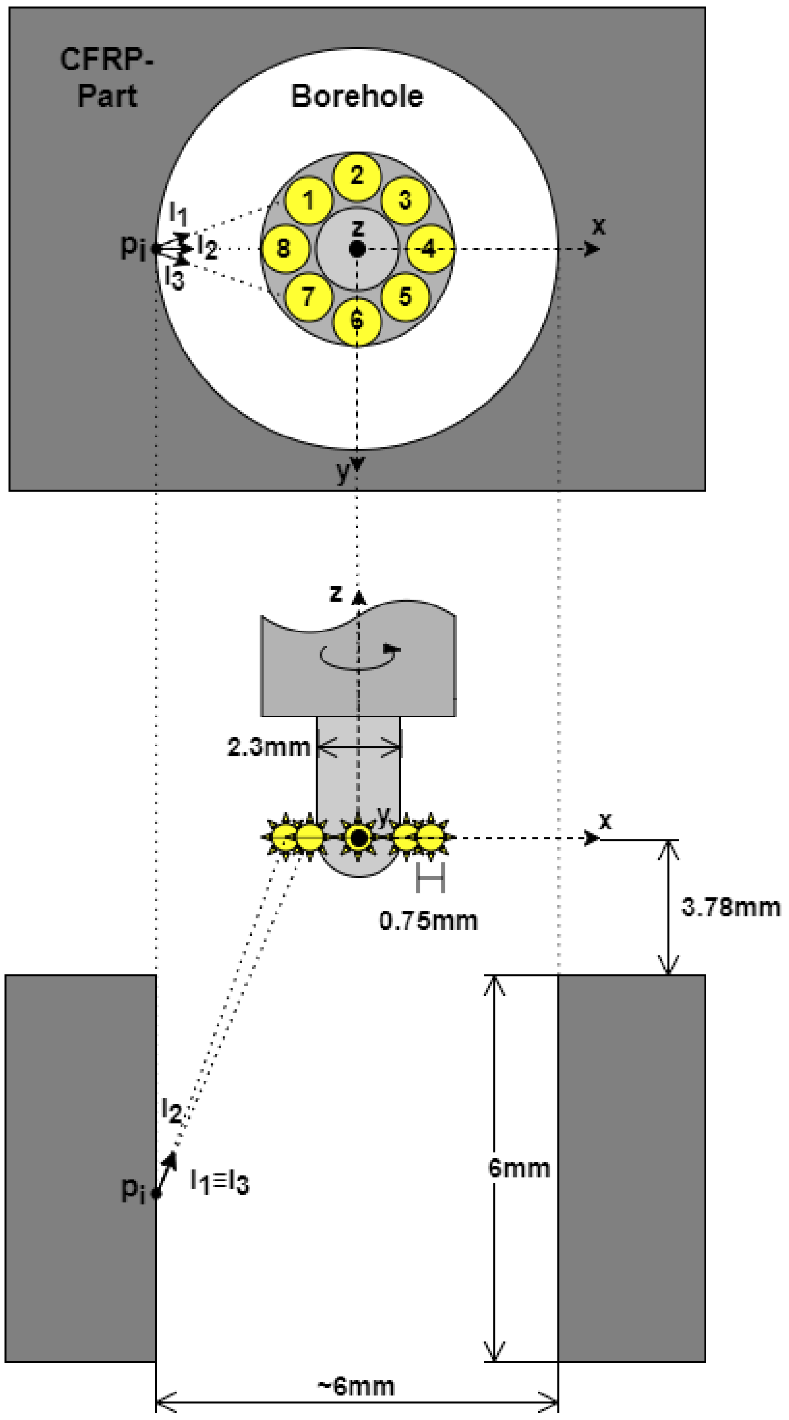


Figure 4.16. Illustration of imaging geometry and borehole dimensions.

For the computation of the surface normals, the vector $\mathbf{I} = [I_1\ I_2\ I_3]^T$ with the adapted intensity values of the three images at this point $\mathbf{p}_i$ is needed. In addition, the normalized vectors $\mathbf{l}_{1,2,3}$ in the direction from $\mathbf{p}_i$ to the three corresponding fibers, aggregated into the matrix $\mathbf{L}$, are necessary:

$$\begin{aligned}
\mathbf{l}_1 &= \begin{bmatrix} l_{x1} \; l_{y1} \; l_{z1} \end{bmatrix} \\
\mathbf{l}_2 &= \begin{bmatrix} l_{x2} \; l_{y2} \; l_{z2} \end{bmatrix} \\
\mathbf{l}_3 &= \begin{bmatrix} l_{x3} \; l_{y3} \; l_{z3} \end{bmatrix} \\
\mathbf{L} &= \begin{bmatrix} \mathbf{l}_1 \\ \mathbf{l}_2 \\ \mathbf{l}_3 \end{bmatrix}
\end{aligned} \tag{4.16}$$

For the calculation of the unit vectors, the coordinate system with the position of the endoscope as origin, is set as shown in Figure 4.16. Next, the x- and y-coordinates of the fibers are computed using the corresponding angle (e. g. for the second fiber this is 90°) in the xy-plane, together with their distance to the endoscope. This distance is the sum of the radii of the endoscope ($r_e = 1.15mm$) and the fiber ($r_f = 0.375mm$).
To compute the z-distance of the fibers to the points in the scene, the z-coordinate of a certain point is estimated with the knowledge of the borehole radius r_b and depth h_b, as follows:

First, the distance d_{et} from the endoscope to the top surface of the hole is calculated under consideration of the camera opening angle β, which is 80°. As for the third imaging step along the z-axis (*step2*) the inlet opening of the hole is just visible, this image is used for the calculation of d_{et}.

$$d_{et} = \frac{r_b}{\tan \frac{\beta}{2}} \tag{4.17}$$

Next, the effect of the perspective projection through the imaging process is recalculated. To prevent errors due to non-concentric circles (example in Figure 4.17) which originate from surface defects or inexact positioning of the imaging system, the distance between the farthest d_{max} and the nearest point d_{min}, in radial direction from the image center, is computed in intervals of 0.01 rad. The maximum point corresponds to a point on the top surface of the hole and the minimum point to a point on the outlet opening. With this projected distortion $d_p = d_{max} - d_{min}$ and the radial distance $d_r = d_i - d_{min}$ of the pixel $\mathbf{p}_i$ to the pixel with the minimum distance $\mathbf{p}_{min}$, the z-coordinate of $\mathbf{p}_i$ is calculated to

$$z_i = d_{et} + h_b - h_b \, \frac{d_r}{d_p} \tag{4.18}$$

For a point on the top surface of the hole ($d_r = d_p$), this results in $z_i = d_{et}$, as expected. The calculated z-coordinate of the pixel corresponds to the negative z-component of the vector from the pixel to the endoscope, as the endoscope and the origin lie on the same plane. Now, the unit vector for the surface normal $\mathbf{n} = [n_x \; n_y \; n_z]^T$ can be computed using the following equations:

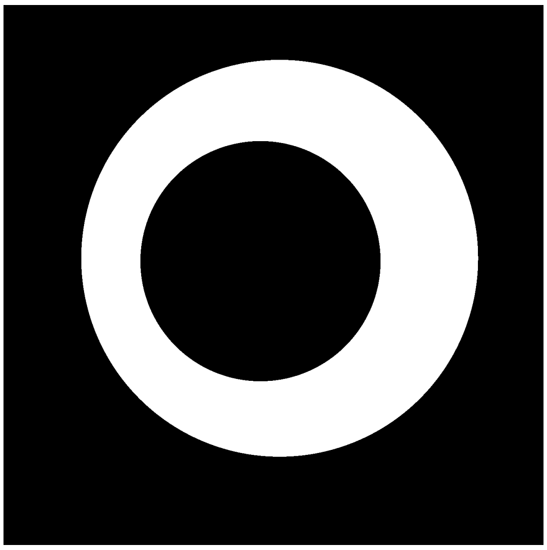


Figure 4.17. Image segmentation mask with unconcentric circles.

$$\varrho\,\mathbf{L}\,\mathbf{n} = \mathbf{I} \tag{4.19}$$

$$\varrho\,\mathbf{n} = \mathbf{L}^{-1}\,\mathbf{I} \tag{4.20}$$

The scalar ϱ corresponds to the surface albedo, which is a measure of the diffuse reflectivity of a material.
To provide additional information and features, the surface normal is also transformed to the spheric coordinate system. With the Euclidean vector norm of $\mathbf{n}$, $r = \sqrt{n_x^2 + n_y^2 + n_z^2}$, the results for the polar angle θ and the azimuth angle φ (starting from the y-axis) are:

$$\theta = \cos^{-1}\frac{n_z}{r} \tag{4.21}$$

$$\varphi = \operatorname{atan2}(n_x,n_y) \tag{4.22}$$

4.3 Proposed Features

In this section, the ideas and algorithms for the extraction of the implemented features are described. The features are separated in two classes, on the one hand the *Photometric Stereo features* and on the other hand the rest of the features (*Standard features*). Figure 4.18 to Figure 4.21, here with an example for a borehole with poor quality, show the different pre-processed images which are the basis for the extraction process. The image which is in the end used, is annotated for each feature separately.

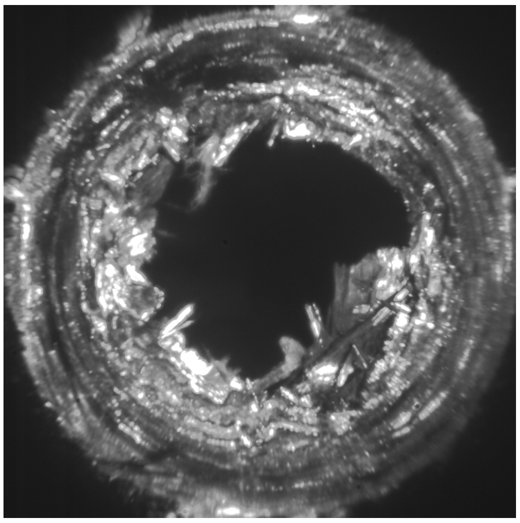


Figure 4.18. Preprocessed image *imgStep2*

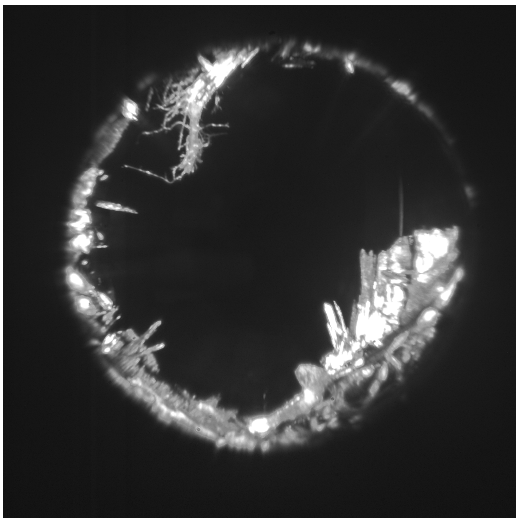


Figure 4.19. Preprocessed image *imgStep12*

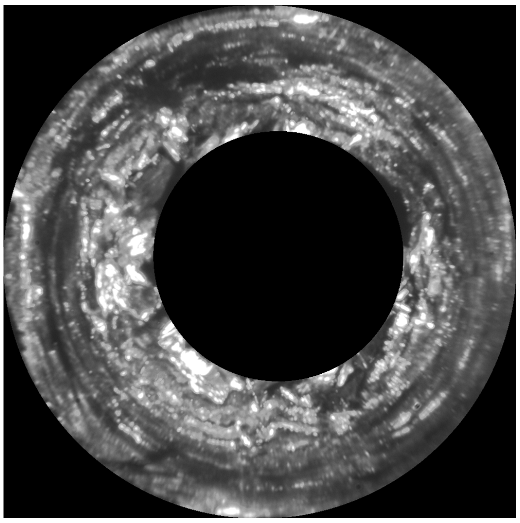


Figure 4.20. Preprocessed image *imgInnerROI*

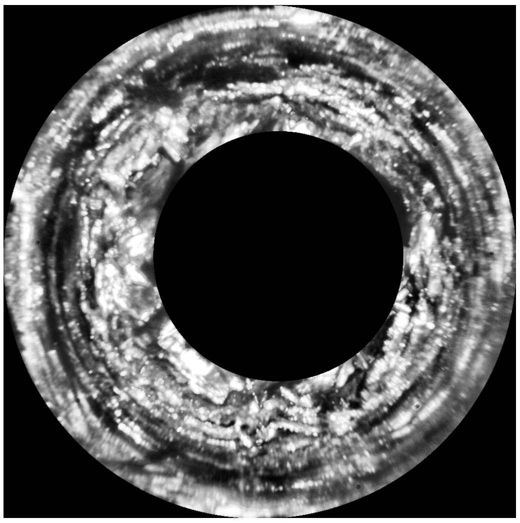


Figure 4.21. Preprocessed image *imgInnerROIEqHist*

4.3.1 Standard Features

Pixel intensities

The first features are the mean value and standard deviation of the pixel intensities. For the image *imgStep2*, the whole image is used, for *imgInnerROI* the pixels with $I_i \neq 0$ are used, so only the segmented region.

$$\bar{I} = \frac{1}{n}\sum_{i=1}^{n} I_i \tag{4.23}$$

$$I_\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(I_i - \bar{I})^2} \tag{4.24}$$

Processed images: imgStep2, imgInnerROI

Circle parameter

This characteristic corresponds to the image segmentation in Section 4.2.2. As features, the circle radii, the ratio of both radii and also the ratio of the radii to the image dimensions (width W), are used.

$$r_{io} = \frac{\hat{r}_{in}}{\hat{r}_{out}} \tag{4.25}$$

$$r_{iw} = \frac{2\,\hat{r}_{in}}{W} \tag{4.26}$$

$$r_{ow} = \frac{2\,\hat{r}_{out}}{W} \tag{4.27}$$

Processed images: imgStep2

Circle quality

This feature also refers to the image segmentation in Section 4.2.2. The idea is, to describe the uniform roundness of the hole, using the Euclidean distances d of the n points to the resulting circle fit, for the inner, as well as for the outer circle. Defects at the inlet and outlet opening, for example, lead to higher values for these distances.
To evaluate this attribute, the mean and standard deviation of the distances for both circles are computed, as follows:

$$d_i = \left|\sqrt{(x_{i_R} - \hat{x})^2 + (y_{i_R} - \hat{y})^2} - \hat{r}\right| \tag{4.28}$$

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{4.29}$$

$$d_\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - \bar{d})^2} \tag{4.30}$$

Processed images: imgStep2

Weighted average distance

The next feature is calculated from the mean weighted and normalized distances of all n pixels to the image center. The pixel intensity I_i is used as weight. For *imgInnerROI*, again only the pixels with $I_i \neq 0$ are used.

$$\bar{d}_w = \frac{1}{n} \sum_{i=1}^{n} I_i \frac{\sqrt{x_{i_R}^2 + y_{i_R}^2}}{W} \tag{4.31}$$

Processed images: imgStep2, imgInnerROI

Energy inner ROI

For this feature, the sum of the squared pixel intensities is computed. It can be an indicator for reflective fraying or other defects on the inside surface of the borehole.

$$e_{ROI} = \sum_{i=1}^{n} I_i^2 \tag{4.32}$$

Processed images: imgInnerROI

Hollow part defections

This feature is similar to the last one, with the single difference that now the *imgStep12* is processed, as for this image the fiber pullout inside the borehole is visible particularly good. Optimal holes result in a low value for this feature, whereas holes with inner defects achieve a high value.

$$e_{Holl} = \sum_{i=1}^{n} I_i^2 \tag{4.33}$$

Processed images: imgStep12

Entropy inner ROI

The entropy of an image is a measurement for the degree of randomness and is used to characterize the texture of the image. If, for example, the intensity values are distributed equally likely, the entropy is at the maximum. If, on the other hand, all intensities are equal, the entropy is zero.
In detail, the MATLAB function *entropyfilt* is used, which calculates the entropy value $E_{y,x}$ at the position (y,x) of the $n \times n$ neighborhood of a pixel for an image with the dimensions r (rows) and c (cols) as follows:

$$E_{y,x} = -\sum_{k=1}^{m} h_k \log_2(h_k) \tag{4.34}$$

h is the normalized histogram of the $n \times n$ neighborhood where m is the amount of bins.
Finally, the mean and standard deviation of the entropy values of all n pixels with $E_{y,x} \neq 0$ is calculated.

$$\bar{E} = \frac{1}{n} \sum_{y=1}^{r} \sum_{x=1}^{c} E_{y,x} \tag{4.35}$$

$$E_\sigma = \sqrt{\frac{1}{n} \sum_{y=1}^{r} \sum_{x=1}^{c} (E_{y,x} - \bar{E})^2} \tag{4.36}$$

Processed images: imgInnerROI

Radial frequency

Due to the layered structure of the CFRP parts and the endoscope imaging, the layers show similarities to concentric circles, especially if the holes are of good quality. The idea of the next feature is to measure this similarity.
Starting from the image center, in 8 radial directions (0°, 45°, 90°, 135°, 180°, -45°, -90° and -135°), the frequency of the bright peaks, due the more reflective layers, is measured. For a comparable measurement, at first the intensity vector in each direction is created and the zero entries are deleted, as they don't correspond to the layered region. Then a moving-average filter is applied to the vector, to eliminate high-frequent intensity differences and to smoothen the vector. The value of the i-th element of the vector $\mathbf{v}$ with the length of the filter $n = 15$ is calculated as follows:

$$v_{filt_i} = \frac{1}{n}(v_i + v_{i-1} + \cdots + v_{i-(n-1)}) \tag{4.37}$$

Afterwards, the function *findpeaks* is used to detect the peaks of the filtered vector $\mathbf{v}_{filt}$ and *diff* to compute the distances between those peaks. Finally, for every direction j the median m_j is computed and then the mean value and standard deviation over all medians is calculated.

$$\bar{m} = \frac{1}{8} \sum_{j=1}^{8} m_j \tag{4.38}$$

$$m_\sigma = \sqrt{\frac{1}{8} \sum_{j=1}^{8} (m_j - \bar{m})^2} \tag{4.39}$$

Processed images: imgInnerROIEqHist

Radial frequency correlation

The next feature is a measure for the symmetry of the hole, using the afore mentioned radial intensity vectors. With the MATLAB function *xcorr*, the cross-correlation of 2 of the 8 distance vectors at a time is computed, as to measure the similarity of the regions. In detail, the correlation of a vector x with another vector y and its shifted copies is needed, as two vectors with the same components which are shifted towards each other, are not identified as similar. If the vectors have different lengths, the one with less entries is repeated until it matches the length l of the other vector.
The cross-correlation $r_{x,y}(n)$ of the vector x and the vector y, shifted by n pixels is calculated with

$$r_{x,y}(n) = \sum_{m=1}^{l} x(m)y(n+m). \tag{4.40}$$

The result is a vector $\mathbf{r}_{x,y}$ with the length n, which contains all cross-correlation values of x with y and its shifted copies. Now, for each possible combination of the 8 radial difference vectors, the most suitable shifting is chosen, which is the maximum of $\mathbf{r}_{x,y}$. These 28 maxima are then averaged and their standard deviation is computed. Finally, both statistical measures are used as features.

$$\bar{r} = \frac{1}{28}\sum_{i=1}^{8}\sum_{j\neq i,j=1}^{8} max(\mathbf{r}_{i,j}) \tag{4.41}$$

$$r_{\sigma} = \sqrt{\frac{1}{28}\sum_{i=1}^{8}\sum_{j\neq i,j=1}^{8} (max(\mathbf{r}_{i,j}) - \bar{r})^2} \tag{4.42}$$

Processed images: imgInnerROIEqHist

Kurtosis 1D

To examine the directional vectors further, the fourth central moment, the kurtosis, is used. In statistics, it describes the shape of a probability distribution and measures the deviation from the normal distribution, so it provides information about outliers. The idea is to use the kurtosis as a measure for defects, as they will change the radial intensity distribution, compared to the ideal case.
The kurtosis k for a vector $\mathbf{v}$ of the length n with the mean $\bar{v}$ and the standard deviation σ is calculated to

$$k = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{v_i - \bar{v}}{\sigma}\right)^4 \tag{4.43}$$

As features, the mean and standard deviation over the 8 radial vectors are used.

$$\bar{k} = \frac{1}{8}\sum_{j=1}^{8} k_j \tag{4.44}$$

$$k_\sigma = \sqrt{\frac{1}{8}\sum_{j=1}^{8}(k_j - \bar{k})^2} \tag{4.45}$$

Processed images: imgInnerROIEqHist

Skewness 1D

Similar to the kurtosis, the third central moment, the skewness s, is computed. It is a measure for the asymmetry of a distribution.

$$s = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{v_i - \bar{v}}{\sigma}\right)^3 \tag{4.46}$$

Again, the mean and standard deviation over the 8 radial vectors are computed.

$$\bar{s} = \frac{1}{8}\sum_{j=1}^{8} s_j \tag{4.47}$$

$$s_\sigma = \sqrt{\frac{1}{8}\sum_{j=1}^{8}(s_j - \bar{s})^2} \tag{4.48}$$

Processed images: imgInnerROIEqHist

Kurtosis 2D

As for the 1-dimensional case, here the kurtosis is computed, but this time for the whole image, to be more concrete, for every column of the matrix. The result is then averaged over all columns m and also the standard deviation is computed.

$$\bar{k} = \frac{1}{m}\sum_{j=1}^{m} k_j \tag{4.49}$$

$$k_\sigma = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(k_j - \bar{k})^2} \tag{4.50}$$

Processed images: imgInnerROI

Skewness 2D

The same approach is used for the skewness of the image:

$$\bar{s} = \frac{1}{m} \sum_{j=1}^{m} s_j \tag{4.51}$$

$$s_\sigma = \sqrt{\frac{1}{m} \sum_{j=1}^{m} (s_j - \bar{s})^2} \tag{4.52}$$

Processed images: imgInnerROI

FAST Corner Detection

One of the more complex features is the result of the corner detection using the MATLAB function *detectFASTFeatures*, which implements the *Features from Accelerated Segment Test* (FAST) method proposed in [45] and [46]. The basic principle of the algorithm works as follows:

At first, a corner-candidate pixel p from the image with the intensity I_p is selected and an appropriate threshold value t is chosen. Considering a circle of a pre-defined number of points (shown in Figure 4.22) around the pixel as the basis for the decision, p is labeled a corner pixel if there are n contiguous pixels which are either darker than $I_p - t$ or brighter than $I_p + t$.
This basic method, together with some extensions for enhanced performance (e. g. machine learning and non-maximal suppression), is often used in real-time applications, when other approaches are not fast enough.

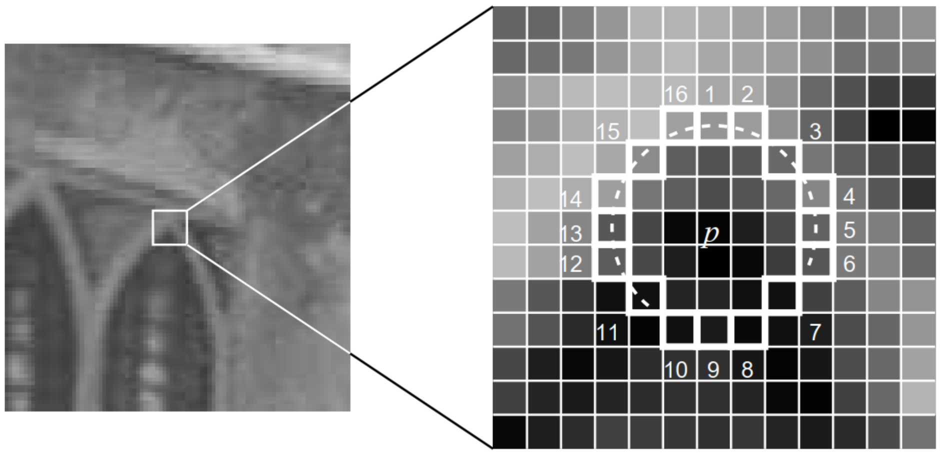


Figure 4.22. Circle around corner-candidate pixel in FAST corner detection. Source: [45]

The idea is now that boreholes with many defects result in a high number of detected corners and therefore, as a feature for the borehole classification, the number of corners is used.

Processed images: imgStep2

Harris Corner Detection

The next feature which also relies on detecting interest points is the Harris corner detection, proposed in [47]. This algorithm finds intensity differences between an image and a slightly shifted copy (by (u,v)) of itself. Mathematically, with the window function w, this can be formulated as

$$E(u,v) = \sum_{x,y} w(x,y)(I(x+u,y+v) - I(x,y))^2. \tag{4.53}$$

A corner in an image is characterized by big intensity differences, so the function $E(u,v)$ has to be maximized for the detection. The result of this maximization is the equation

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}, \tag{4.54}$$

with

$$\mathbf{M} = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \tag{4.55}$$

where I_x and I_y are the partial derivatives of the image in x and y directions. Next, the Harris response

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{4.56}$$

determines the "cornerness" of the window, i. e. if it contains a corner. λ_1 and λ_2 are the eigenvalues of M and k is an empirical constant with $k \in \left[0.04, 0.06\right]$. A small R (small λ_1 and λ_2) means no corner, $R < 0$ ($\lambda_1 \gg \lambda_2$ or vice versa) indicates an edge and a large R (both λ_1 and λ_2 are large) a corner. Finally, non-maxima suppression and thresholding is applied, to get the most important corner locations. In this work, the amount of corners is in the end used as a feature.

Processed images: imgStep2

SURF Interest Point Detection

Another method to detect points of interest in an image is the *Speeded Up Robust Features* (SURF) method [48]. It is an enhancement compared to similar approaches (e. g. *Scale Invariant Feature Transform* (SIFT) [49]), regarding the performance, mainly due to the use of box filters as approximations of Gaussian second derivative masks. The two images on the left side in Figure 4.23 show the original second derivative masks $\mathbf{G}_{yy}$ and $\mathbf{G}_{xy}$. The right side shows the box filters denoted as $\mathbf{D}_{yy}$ and $\mathbf{D}_{xy}$.

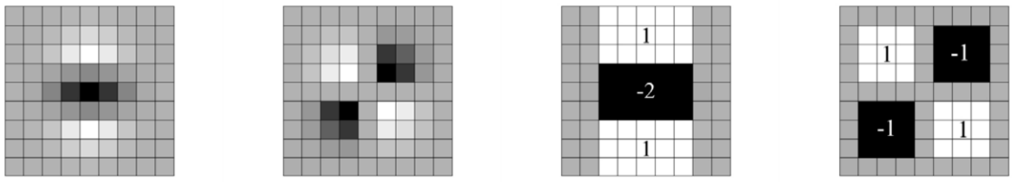


Figure 4.23. Gaussian second derivative masks (left) and approximations with block filters (right). Source: [48]

The big advantage of this approximation is the computation efficiency using integral images, which supports very little processing time for the application of rectangular masks. It basically allows the computation of any rectangular sum of pixel intensities in only four array references, as shown in Figure 4.24. The integral image equation for point 1 at the position (x,y) is

$$II(x,y) = \sum_{x' \leq x,\, y' \leq y} I(x',y'). \tag{4.57}$$

This equals the sum of all points in the rectangle A. Therefore, the integral image at point 2 is the sum of all points of A and B, and so on. Thus, the rectangular sum of D equals $4 + 1 - 3 - 2$, which are the four necessary array references.

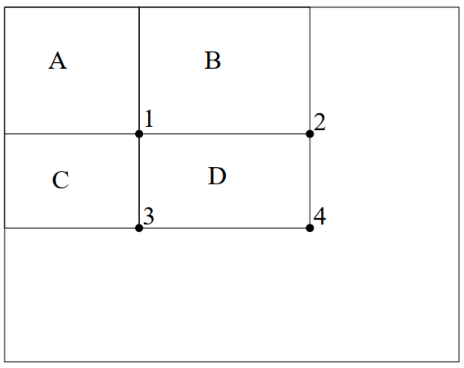


Figure 4.24. Graphical representation of integral images. Source: [50]

For the detection of the interest points, the Hessian matrix

$$\mathbf{H}(\mathbf{x},\mathbf{y}) = \begin{bmatrix} L_{xx}(x,y) & L_{xy}(x,y) \\ L_{xy}(x,y) & L_{yy}(x,y) \end{bmatrix} \tag{4.58}$$

is used, where $L_{xx}(x,y)$ is the convolution of $\mathbf{G}_{xx}$ with the image in (x,y). The maxima of the determinant of the Hessian matrix are then located by non-maximum suppression in a neighborhood around (x,y) and the amount of maxima detected is used as a feature for the borehole classification. The SURF algorithm is not further explained, as this work only uses the approach for detection of the interest points and not for their description.

Processed images: imgStep2

MSER Interest Region Detection

The last interest region detection method, used for feature extraction, is *Maximally Stable Extremal Region* (MSER) [51]. The basic idea of this approach is, to search the gray scale image for regions with the following characteristics:

- Firstly, a contrast in intensity compared to their surrounding areas
- Secondly, homogeneous pixel intensities within the region

A general explanation of this algorithm starts with the input image, shown in Figure 4.25 a).

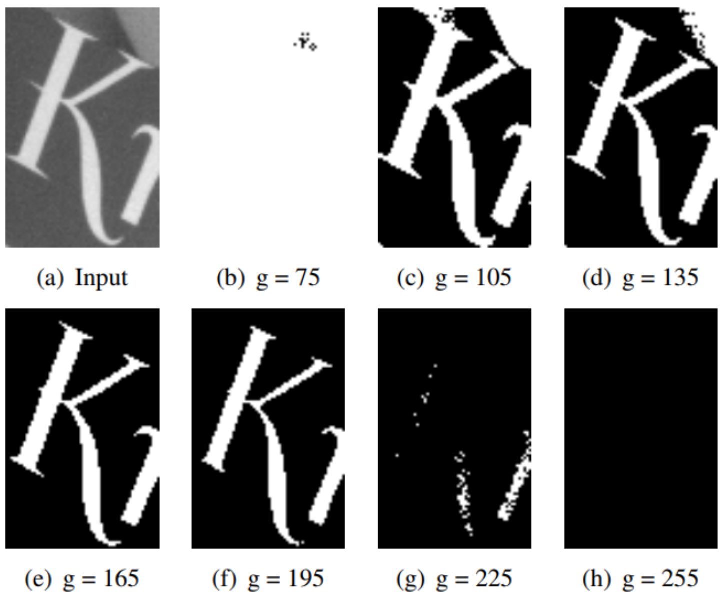


Figure 4.25. Thresholding process with the resulting images for each step. Source: [52]

Now, step by step, the intensity threshold g is increased and pixels below this threshold are set to zero. In every step, the remaining connected regions are stored in a data structure, e. g. a connected tree, as used in [52]. For this example, the data structure is shown in Figure 4.26.

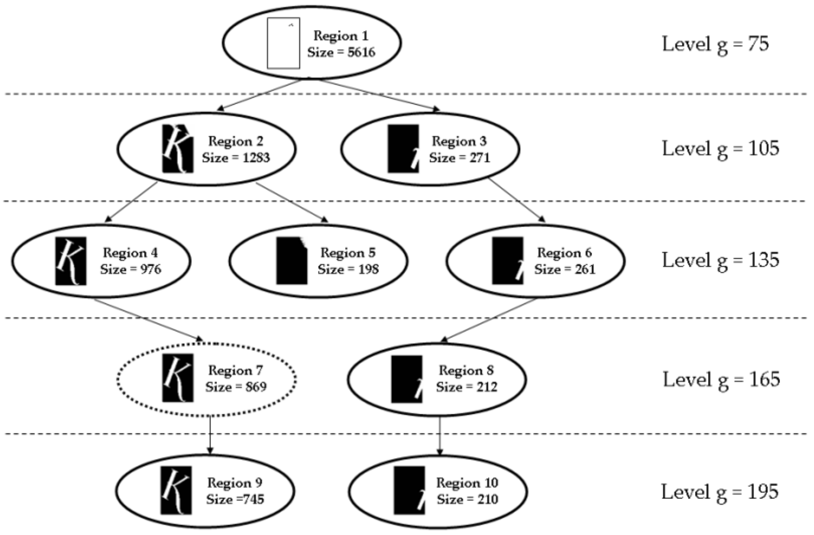


Figure 4.26. Structure of the connected tree used for the MSER algorithm. Source: [52]

The MSERs are identified by computing a stability value for each region in the connected tree. As the size of the regions decreases with increasing threshold, a stable region is characterized by an approximately constant region size. The stability value Ψ for the region R_i^g, which is a result of the thresholding at a value g, is defined as

$$\Psi(R_i^g) = (|R_j^{g-\Delta}| - |R_k^{g+\Delta}|)/|R_i^g| \tag{4.59}$$

where $|.|$ is the cardinality and Δ is the parameter for the desired intensity range of the stability, in both directions j and k. The MSERs are now the regions where Ψ is a local minimum along each path to the root of the tree [52], here for example region 7. A similar procedure can be applied to detect dark regions.
In this work, the amount of the MSERs found in each borehole image is used as a feature. The reason for the implementation of all the different interest point/region detection methods is the easy availability in MATLAB and also a possible comparison of them.

Processed images: imgStep2

Interest point matches

The next feature is a measure for the homogeneity of the borehole inside. This is done by comparing the original image with a rotated copy of itself (e. g. by 90°). First, both images are searched for interest points, using the Harris corner detector, as explained before. Then, at the location of these points, so called *Histogram of Oriented Gradients* (HOG) descriptors, first described in [53] and used in [54], are computed using MATLAB's function *extractHOGFeatures*.

The calculation starts with filtering the borehole image with a $[-1\ 0\ 1]$ kernel for the x-direction and $[-1\ 0\ 1]^T$ for the y-direction. Then, for every pixel $\mathbf{p}_i$, the magnitude g and direction θ of the gradient from the two resulting filtered images g_x and g_y is computed with

$$g = \sqrt{g_x^2 + g_y^2} \tag{4.60}$$

$$\theta = \arctan\left(\frac{g_y}{g_x}\right). \tag{4.61}$$

Next, the image is divided into quadratic cells with a parameter defined size (e. g. 8×8) and for each cell a histogram over the unsigned gradient direction (0°...180°) as bins (9 in total), with the values of g as basis for the votes (the values which go into the bins), is created. To remove the influence of different illuminations, the histograms are normalized over 4 neighboring cells (e. g. 16×16 in total). These 4 normalized cells are then combined to a feature vector with $9 \cdot 4 = 36$ elements.
As already mentioned, the HOG descriptors for both the original image and the rotated copy are computed. To be able to match similar features in both images, the pairwise distance between the feature vectors is computed. If this distance is under a certain threshold, the features count as a match. The idea in this work is now to count these matches and use the result as a feature. In addition, the ratio of the matches to the interest points detected by the Harris method is relevant. An example for the matching is depicted in Figure 4.27. Crucial for this approach is the rotation variance of the HOG descriptors, which allows to find matches between different regions of the image, as otherwise only physically identical points would be matched.

Processed images: imgStep2

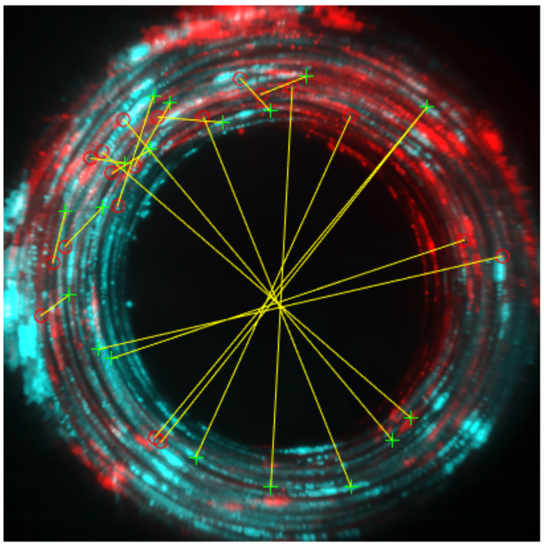


Figure 4.27. Comparison of a borehole image with a rotated copy of itself using matching interest points.

Image correlation with rotated copy

Another measure for the similarity of different regions within a single image is the 2D-cross-correlation of the image with a rotated copy of itself. As for the last feature, the rotation angle is 90°.
The 2D-correlation $C(k,l)$ of the $M \times N$ matrix $\mathbf{X}$ with the matrix $\mathbf{Y}$, shifted by k pixels in y-direction and l pixels in x-direction is calculated by

$$C(k,l) = \sum_{m=1}^{M} \sum_{n=1}^{N} X(m,n)Y(m+k,n+l). \tag{4.62}$$

For the computation of $C(k,l)$, only pixels of the borehole inside surface are used. To improve the performance of this approach, the maximal shifts in x- and y-direction are only small compared to the image dimensions.
Moreover, the correlation of the image with itself (the autocorrelation) is computed and as features the maximum of the cross-correlation matrix $\mathbf{C}$ and also the ratio of the autocorrelation to this maximum are used.

Processed images: imgInnerROI

Image difference to rotated copy

This feature is calculated similar to the previous, however, here the difference $D(k,l)$ between the two images is of interest.

$$D(k,l) = \sum_{m=1}^{M} \sum_{n=1}^{N} |X(m,n) - Y(m+k,n+l)| \tag{4.63}$$

Again, the rotated copy is shifted by k pixels in y-direction and l pixels in x-direction. Like before, only the borehole inside surface is processed. The minimum of the difference matrix $\mathbf{D}$ is used as a feature.

Processed images: imgInnerROI

Gray co-occurrence matrix

The last “standard” features are computed from the *Gray Level Co-occurrence Matrix* (GLCM), which provides various information about a gray level image. As an example, the GLCM of the Matrix $\mathbf{M}$ is computed. The elements of this matrix show the occurrences of certain gray value combinations in $\mathbf{M}$. The specific type of combination used in this work is the direct horizontal neighborhood of two pixels. For instance, the sequence [0 2] occurs 2 times in $\mathbf{M}$, thus, the corresponding entry in the GLCM is 2.

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 2 & 2 & 1 \\ 1 & 3 & 1 & 3 \\ 3 & 4 & 3 & 4 \end{bmatrix} \tag{4.64}$$

$$\mathbf{GLCM} = \begin{bmatrix} (0{,}0) & (0{,}1) & (0{,}2) & (0{,}3) & (0{,}4) \\ (1{,}0) & (1{,}1) & (1{,}2) & (1{,}3) & (1{,}4) \\ (2{,}0) & (2{,}1) & (2{,}2) & (2{,}3) & (2{,}4) \\ (3{,}0) & (3{,}1) & (3{,}2) & (3{,}3) & (3{,}4) \\ (4{,}0) & (4{,}1) & (4{,}2) & (4{,}3) & (4{,}4) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.65}$$

For the computation of the GLCM, the built-in MATLAB function *graycomatrix* is used. It scales the processed image down to 8 intensity values and provides a normalized 8×8 matrix as a result. With the function *graycoprops*, now 4 features are extracted from it:

The first one is the contrast in intensity between a pixel and its neighbors, over the whole image. It is calculated by

$$Contrast = \sum_{i=1}^{8}\sum_{j=1}^{8} |i-j|^2\, GLCM(i,j). \tag{4.66}$$

It is apparent that for an image with constant intensity, the contrast would be computed to 0.

The next feature is the correlation of a pixel with its neighbors, calculated again over the whole image. With μ_i and μ_j as the mean values and σ_i, σ_j as the standard deviations of the GLCM values in i- and j-direction, the correlation is calculated as follows:

$$Correlation = \sum_{i=1}^{8}\sum_{j=1}^{8} \frac{(i-\mu_i)(j-\mu_j)GLCM(i,j)}{\sigma_i \sigma_j} \tag{4.67}$$

The energy of the GLCM is the next feature and is the result of the sum of the squared elements:

$$Energy = \sum_{i=1}^{8}\sum_{j=1}^{8} GLCM(i,j)^2 \tag{4.68}$$

The last feature, extracted from the GLCM, is the homogeneity, as a measure for the similarity of the GLCM to a diagonal matrix.

$$Homogeneity = \sum_{i=1}^{8}\sum_{j=1}^{8} \frac{GLCM(i,j)}{1+|i-j|} \tag{4.69}$$

For a diagonal GLCM, the resulting homogeneity is 1 (due to the normalization). With more equally distributed elements, this value is < 1.

Processed images: imgStep2

4.3.2 Photometric Stereo Features

The following features are based on the photometric stereo approach, described in Section 4.2.3. To be able to evaluate the features, “ideal” surface normals are computed for every point $\mathbf{p}_i$ of a borehole, as the dimensions of the holes and the parameters of the imaging geometry are known. These surface normals are three dimensional vectors which are perpendicular to the cylindrical inside surface of the holes.

Surface Normal Distance

This feature describes the Euclidean distance d of the surface normals $\mathbf{n}_j^r$, computed with the photometric stereo method, to the “ideal” normals $\mathbf{n}_j^i$. The mean and standard deviation of the distances over all m points of the surface are used as features for both photometric stereo implementations (see Section 4.2.3).

$$d_j = \left|\left|\mathbf{n}_j^r - \mathbf{n}_j^i\right|\right| = \left|\left|\begin{matrix} x_j^r - x_j^i \\ y_j^r - y_j^i \\ z_j^r - z_j^i \end{matrix}\right|\right| \tag{4.70}$$

$$\bar{d} = \frac{1}{m}\sum_{j=1}^{m} d_j \tag{4.71}$$

$$d_\sigma = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(d_j - \bar{d})^2} \tag{4.72}$$

Surface Normal Distance 1/6,1/8

Similar to the previous feature, the distance between “ideal” and calculated surface normals is of interest. Depending on the photometric stereo implementation, the mean and standard deviation are computed separately for each image region, which results in 12 features for the first implementation and 16 features for the second implementation.

Spherical Histogram Intersection

As mentioned at the end of Section 4.2.3, the Cartesian surface normals are transformed to the spherical coordinate system to extract further information. One of the spherical features is the intersection of the real ($\mathbf{h}^r$) with the ideal histogram ($\mathbf{h}^i$) over the azimuth angle φ. It is calculated by

$$I_\varphi = \frac{1}{n}\sum_{j=1}^{m} min(h_j^r, h_j^i), \tag{4.73}$$

where m is the number of bins in the histograms and n is the number of pixels in all bins. Again, this feature is computed for both photometric stereo implementations.

Spherical Histogram Intersection 1/6,1/8

The spherical histogram intersection is also computed separately for each image region, resulting in 6 features for the first implementation and 8 features for the second implementation, respectively.

Spherical difference

This feature describes the difference in the azimuth angle φ between the photometric stereo and the "ideal" surface normals, respectively. This angle difference δ is computed using the MATLAB function *angDiff* and the result is averaged over all m borehole surface points and, in addition, the standard deviation is used a feature, for both photometric stereo implementations.

$$\delta_j = angDiff(\varphi_r,\varphi_i) \tag{4.74}$$

$$\bar{\delta} = \frac{1}{m}\sum_{j=1}^{m}\delta_j \tag{4.75}$$

$$\delta_\sigma = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\delta_j - \bar{\delta})^2} \tag{4.76}$$

Spherical difference 1/6,1/8

The previous measure is also computed for each image region separately, which means 12 features for the first and 16 features for the second implementation.

Spherical mean

The mean value of both spherical angles φ and θ of the photometric stereo surface normals, over all m borehole inside pixels, is used for this feature, for both implementations.

$$\bar{\varphi} = \text{atan2}\left(\frac{1}{m}\sum_{j=1}^{m}sin(\varphi_j), \frac{1}{m}\sum_{j=1}^{m}cos(\varphi_j)\right) \tag{4.77}$$

$$\bar{\theta} = \text{atan2}\left(\frac{1}{m}\sum_{j=1}^{m}sin(\theta_j), \frac{1}{m}\sum_{j=1}^{m}cos(\theta_j)\right) \tag{4.78}$$

Spherical mean 1/6,1/8

Similarly, the spherical means for each image region are used as features for both photometric stereo implementations.

4.4 Summary

In this chapter, the surface feature extraction process with all the implemented ideas and algorithms was described in detail. To begin with, the preparation and pre-processing of the datasets were explained. This included the imaging geometry, used for recording the boreholes, the stitching of the resulting images for different illumination, and the image segmentation which was necessary for the extraction of most of the features. Moreover, the photometric stereo approach, also used for feature extraction, was described.
Eventually, the mathematical formulations for the extracted features were presented. In the following chapter, the proposed features and methods are evaluated in detail.

5 Experiments and Evaluation

In the previous chapter, all in all 158 features for the borehole classification were described. Following, the methods and settings for the evaluation of those features regarding their correlation with the drilling parameters are explained, as well as the results of the experiments for feature selection and classifier comparison. In addition, approaches to deal with the dataset imbalance are evaluated.

5.1 Evaluation Methods and Settings

5.1.1 Dataset Balancing

The dataset processed in this work was described in Section 4.1. It contains borehole images of three different quality categories, for two types of CFRP parts, called Fabric and UD. The classes are not equally represented in both datasets, meaning that the datasets are imbalanced. This imbalance might cause the classification models to favor the majority class and therefore to produce misleading results. To avoid this, two methods for balancing the dataset are explained here and evaluated later in Section 5.4.

Class-Specific Weighting

The idea of this approach is to vary the influence of an observation on the machine learning model according to the total amount of observations for each class. Regarding classification for example, this can lead to higher misclassification costs for samples of classes which are underrepresented in the dataset. The weighting factor used in this work is calculated by

$$w_j = \sqrt{\frac{N_{max}}{N_j}}, \tag{5.1}$$

for elements of the class j, with N_{max} being the number of observations in the majority class and N_j the corresponding amount for the class j.
If not explicitly stated otherwise, this balancing approach is used for all evaluations in this work.

SMOTE

The concept behind this approach was already explained in detail in Section 3.5.4. SMOTE creates additional "synthetic" observations for the underrepresented classes to balance the dataset. These new samples are derived from "real" observations.

In this work, SMOTE is used after the separation of the data into training- and test set, to avoid bias due to too similar observations in both sets.

5.1.2 Categorization of Features

For the experiments, the features presented in Chapter 4 are divided into 4 sets which are evaluated separately in the following sections. The categories are:

- PS6
  - All features which are calculated from the first photometric stereo implementation (see Figure 4.6), in total 50
- PS8
  - All features which are calculated from the second photometric stereo implementation (see Figure 4.7), in total 64
- Standard
  - The rest of the features, in total 44
- All
  - All features together, in total 158

5.1.3 Feature Selection

The overall 158 features, described in the previous chapter, are collected in a feature matrix and normalized over all borehole instances, separately for each dataset (Fabric and UD). The dimensions of the matrix are $M \times N$, where N is the number of instances and M is the total number of features.
Next, the feature selection is applied, as to eliminate redundant features that only increase the dimensionality of the matrix and don't add further information about the data. This step leads to a reduced complexity of the classification problem. Figure 5.1 gives an overview on the feature selection process.

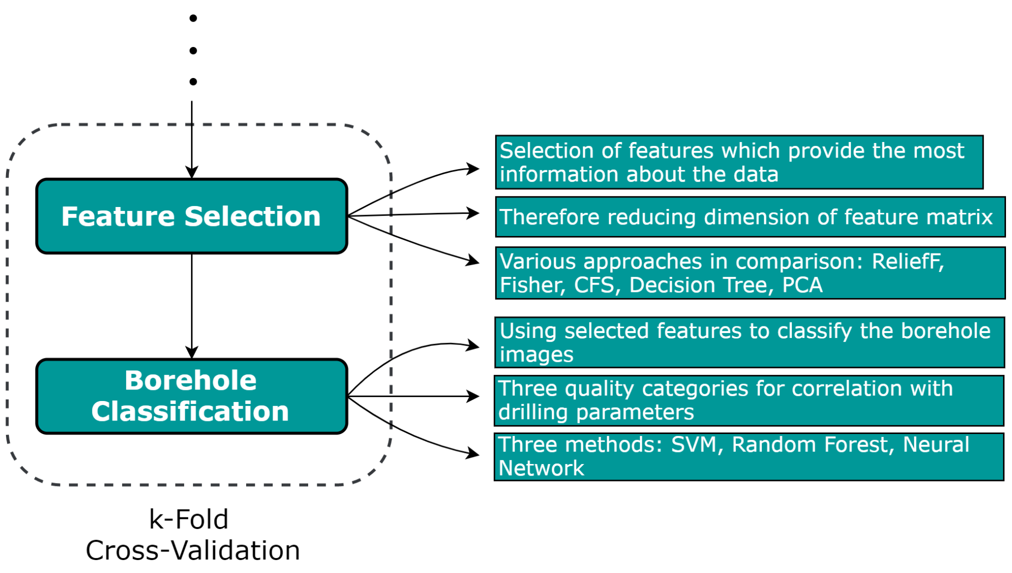

Feature Selection
Borehole Classification
k-Fold Cross-Validation
Selection of features which provide the most information about the data
Therefore reducing dimension of feature matrix
Various approaches in comparison: ReliefF, Fisher, CFS, Decision Tree, PCA
Using selected features to classify the borehole images
Three quality categories for correlation with drilling parameters
Three methods: SVM, Random Forest, Neural Network

Figure 5.1. Overview on the feature selection and classification steps.

The following feature selection methods, explained in Section 3.4, are evaluated in this work:

- ReliefF
- Fisher Score
- Correlation-based Feature Selection
- Decision Tree
- Principal Component Analysis

The whole selection process is done in MATLAB. The *reliefF* function is part of the *Statistics and Machine Learning Toolbox* (SMLT) and is used to rank the importance of features, based on the ReliefF algorithm [24].
The functions *fsfisher* and *fsCFS* are available in the *ASU feature selection repository* [55] and implement the Fisher Score [26] and the CFS method [28], respectively.
The feature selection using the Decision Tree [30] was implemented with the *fitctree* and *predictorImportance* functions which are both available in the SMLT.
The last feature selection method used in this work is PCA, described in Section 3.4.5. As explained in this section, it differs from the other algorithms, because it is unsupervised, which means that the function doesn't have knowledge of the class labels, compared to the supervised approach, which relies on this information. It is also different in a way that it doesn't search for the best features, but transforms the feature space in order to compress it and, therefore, reduces its dimensionality.

5.1.4 Classification Methods and Parameters

The most important details of the borehole classification are briefly outlined in Figure 5.1. This step of the machine learning application is used to evaluate the features which were selected with the methods mentioned in the previous section. The features are fed to three classifiers in evaluation, namely:

- Support Vector Machine
- Random Forest
- Artificial Neural Network

The underlying concepts of these approaches are explained in detail in Section 3.5. For the SVM classifier, at first a template with all the necessary parameters is created with the MATLAB function *templateSVM*. The classification model is then generated and trained using the function *fitcecoc*.
The Random Forest algorithm [23] is implemented in the function *TreeBagger*, which creates and trains the model.
The last classifier, the ANN, is carried out using the Keras Deep Learning Library [56] available in Python.

Classifier parameters

In general, the results of a classification process depend strongly on the parameter values used to create the classifier model. Thus, for every method, values for the most important parameters were experimentally determined through a grid search in an appropriate parameter range for every combination of feature- and dataset. The parameter names are chosen according to the respective classifier implementation in MATLAB and Python.
Table 5.1 shows the resulting parameter values for the SVM. As a *kernelFunction*, a polynomial was selected, with the *polynomialOrder* listed in the table. The *kernelScale* parameter is the value which every element of the feature matrix is divided by, before applying the kernel. The *boxConstraint* is a parameter which determines the weight of misclassifications, with a high value leading to a more strict separation between the data.
The parameter values for the ANN are listed in Table 5.2. The *layerCount* and *layerSize* determine the number of hidden layers and neurons per layer, respectively. The *epochs* parameter is the number of forward and backward passes the whole training set goes through. The hidden layer *activation* function used in this experiment is the rectifier.
Finally, the parameter values for the random forest approach are shown in Table 5.3. *minLeafSize* determines the minimum number of observations in a terminal node. *nrTrees* is the total amount of decision tress in the classification model and *maxNumSplits* is the maximum number of splits allowed per tree. The last parameter, *nrPredictors*, is the amount of random features selected for every tree.

Parameter	**Standard**		**PS6**		**PS8**		**All**	
	Fabric	UD	Fabric	UD	Fabric	UD	Fabric	UD
polynomialOrder	4	5	4	4	3	5	4	5
boxConstraint	5	5	5	5	5	5	5	5
kernelScale	5	4	5	6	7	5	7	5
kernelFunction	polynomial							

Table 5.1: Classification parameters for SVM.

Parameter	**Standard**		**PS6**		**PS8**		**All**	
	Fabric	UD	Fabric	UD	Fabric	UD	Fabric	UD
layerSize	$\frac{M}{2}$	$\frac{3}{4}M$	$\frac{M}{2}$	$\frac{3}{4}M$	$\frac{M}{2}$	$\frac{3}{4}M$	$\frac{M}{2}$	$\frac{3}{4}M$
layerCount	1							
epochs	700							
activation	rectifier							

Table 5.2: Classification parameters for ANN.

Parameter	**For every set**
minLeafSize	1
nrTrees	300
maxNumSplits	20
nrPredictors	$\sqrt{M}$

Table 5.3: Classification parameters for RF.

Performance measures

The classification results consist of the calculations for accuracy and recall measures which were previously described in Section 3.5. To be able to give a good estimate on the classifier performance for new data, the evaluations were done using 5-fold cross validation. For an even better estimate and to account for statistical deviations, the results are averaged over 10 iterations of cross validation for each classifier. Additionally, the results are presented via confusion matrices.

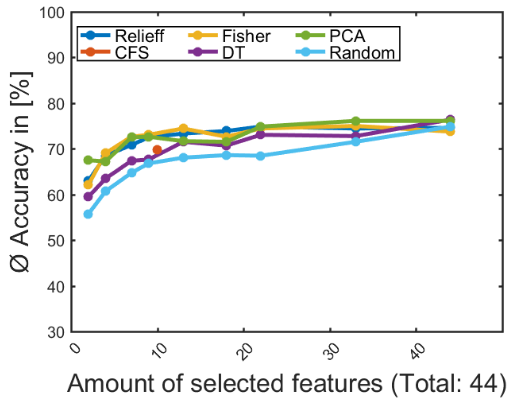
5.2 Evaluation of Feature Selection Methods

To evaluate the influence of the chosen methods for ranking and selecting the features, Figure 5.2 to Figure 5.9 show the classification results for accuracy and recall over the amount of selected features for every dataset, featureset and classifier. In the graphs, an x-value of 20 corresponds to the 20 most relevant features according to the respective feature selection method.

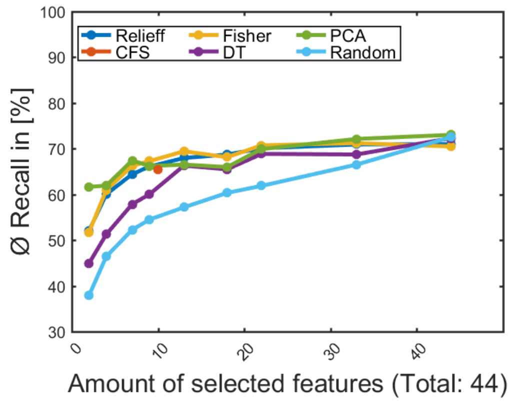
As the output of the CFS approach is the most relevant set of features and no ranking of individual features, there is only one data point for this selector in the diagram. Regarding PCA, the amount of features corresponds to the number of PCs used.
In addition to the mentioned feature selection approaches, also a series of results is plotted for randomly chosen features for comparison.

Interpretation of results

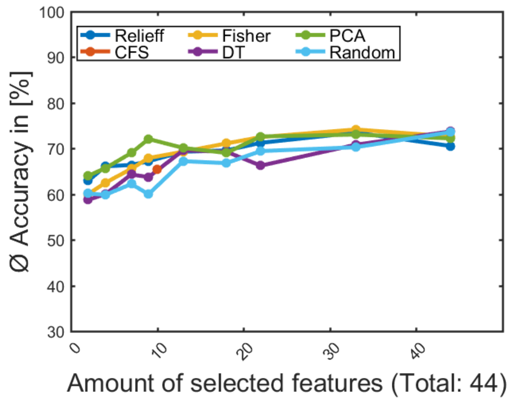
A general evaluation of the feature selection methods is not easy, as the results depend on the one hand on the feature set and on the other hand also on the dataset. In addition, the type of classifier plays a role, as Figure 5.2 to Figure 5.9 show.
An indication for a successful feature selection is an approximately horizontal line and in the ideal case, the performance drops with a higher number of features. A rising line indicates that relevant features are not ranked as such and thus are only used for classification when a high percentage of the total feature set is used.
In the result images, the left part of the graphs is important for the evaluation of the feature selection. The fewer features selected, the more influence has the quality of the feature selection method on the classifier performance. With a higher percentage of features used, the results are more similar, as most of the graphs show clearly. This is not the case for the combination of random forest classifier and PCA. Instead, in most of the graphs the performance decreases with increasing number of features. The reason for this effect is most probably a characteristic property of the random forest classifier, as it chooses a random subset of features for the creation of each decision tree. The features correspond to the PCs which differ in the variance of the data that they contain. Only a few of the PCs account for a high variance and as a result, this might lead to problems when many of the random subsets don't contain relevant PCs and are therefore not used for the classification.
In general, most of the graphs show a quite good prediction performance when around 25 % of the whole featureset is used, with a lower percentage resulting in a decrease of accuracy and a higher percentage only improving the results slightly. Especially for the recall measure it is important not to select too few features, as can be seen for most of the feature sets.
Regarding the comparison between the random feature selection and the proposed feature selection algorithms, in most of the cases the random option leads to poorer classification results when the amount of selected features is low. Thus, the use of feature selection is favorable in this work and the results of the different feature selection methods are combined for a feature ranking estimate, presented in Section 5.3.



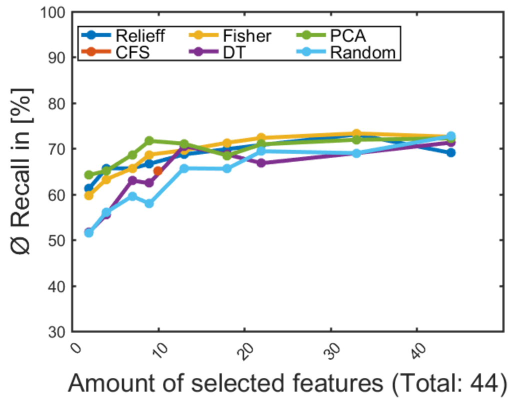
(a) SVM Accuracy



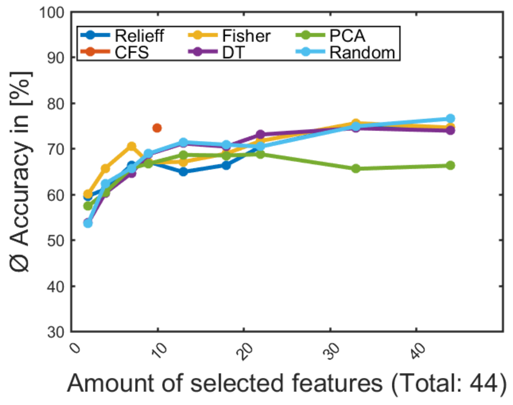
(b) SVM Recall



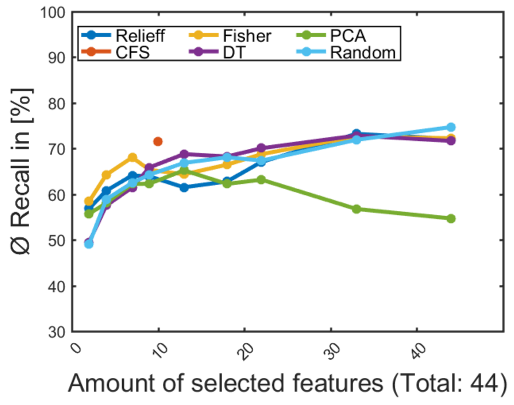
(c) ANN Accuracy



(d) ANN Recall

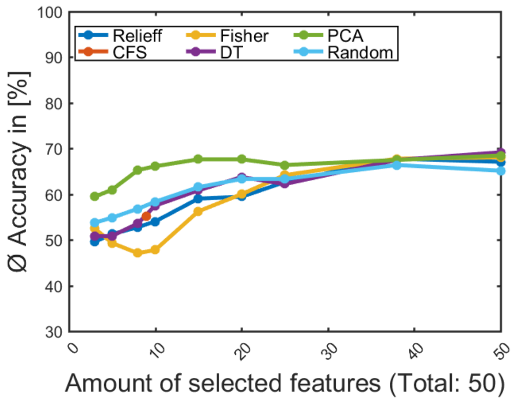


(e) RF Accuracy

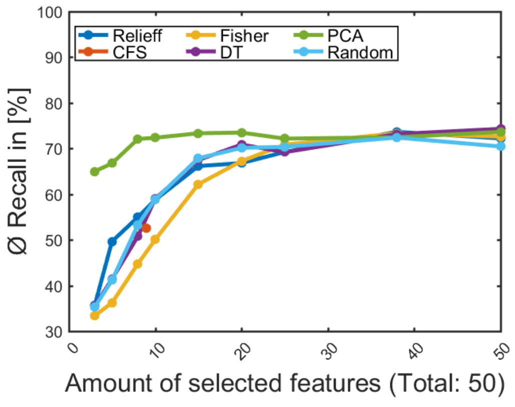


(f) RF Recall

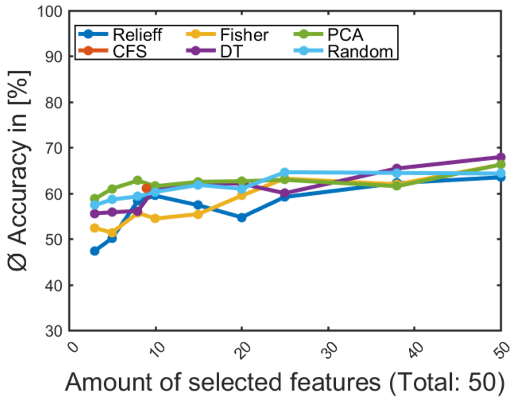
Figure 5.2. Comparison of feature selection methods for Dataset: Fabric/Featureset: Standard.



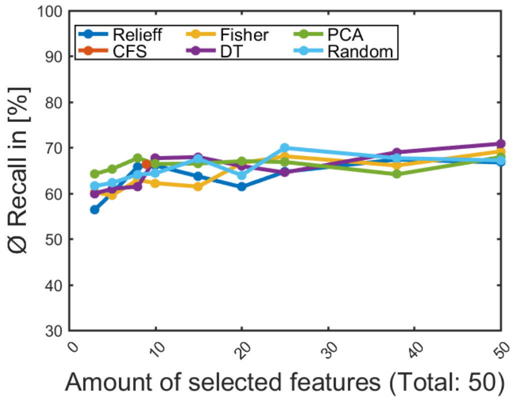
(a) SVM Accuracy



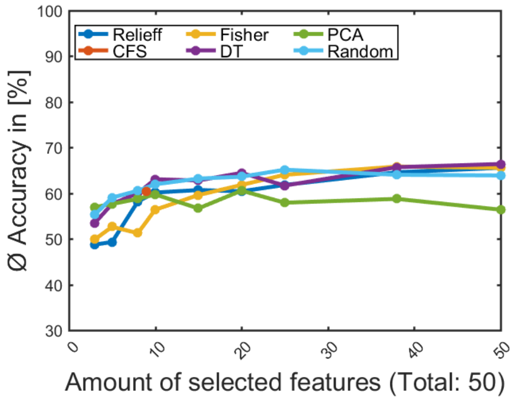
(b) SVM Recall



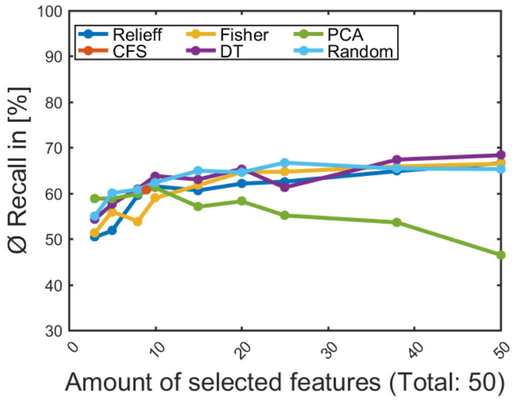
(c) ANN Accuracy



(d) ANN Recall

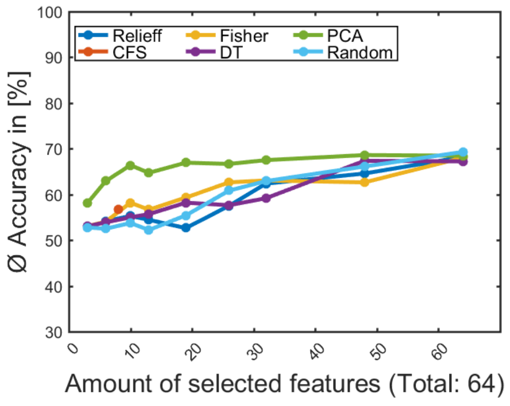


(e) RF Accuracy

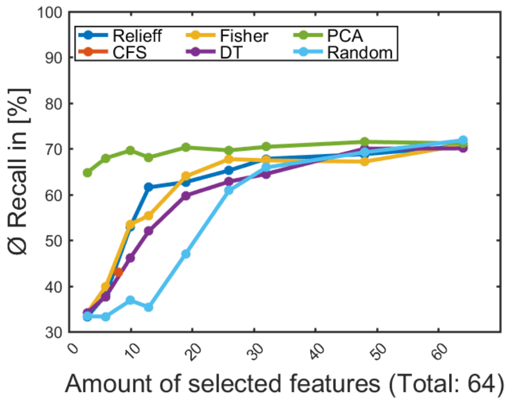


(f) RF Recall

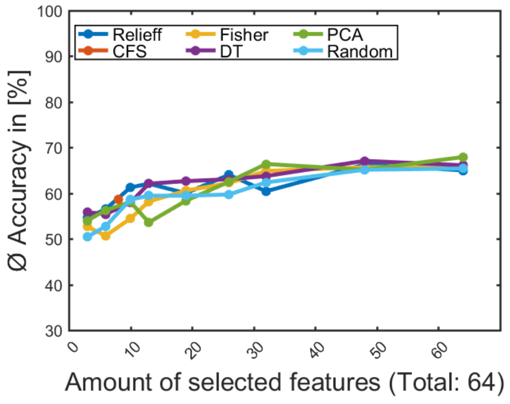
Figure 5.3. Comparison of feature selection methods for Dataset: Fabric/Featureset: PS6.



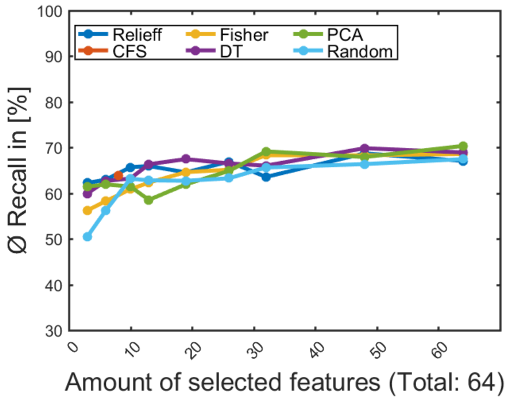
(a) SVM Accuracy



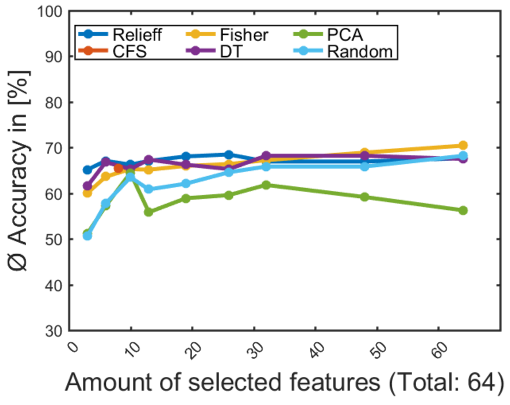
(b) SVM Recall



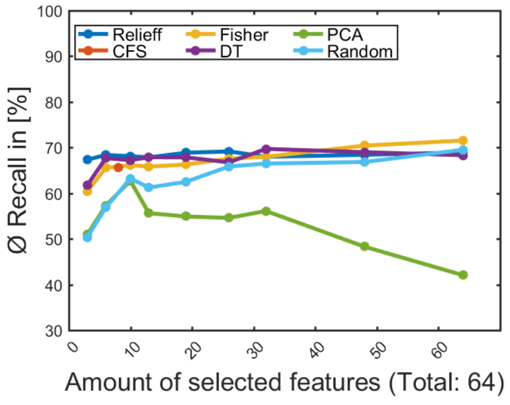
(c) ANN Accuracy



(d) ANN Recall

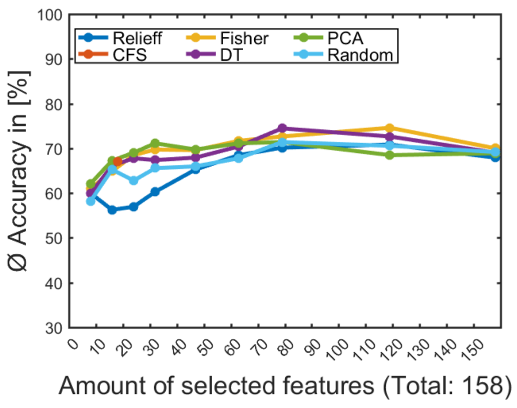


(e) RF Accuracy

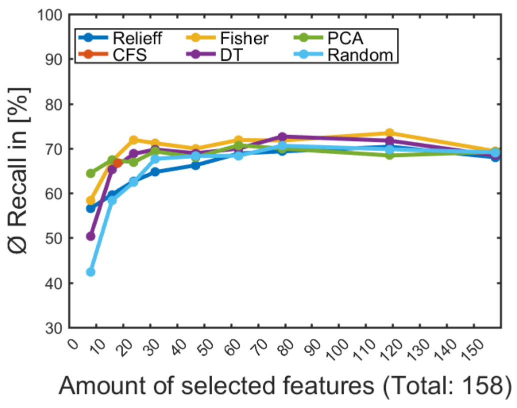


(f) RF Recall

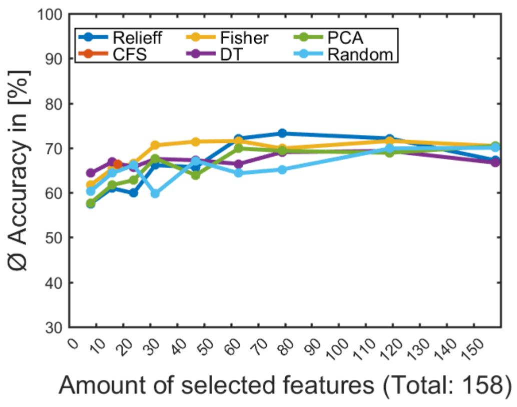
Figure 5.4. Comparison of feature selection methods for Dataset: Fabric/Featureset: PS8.



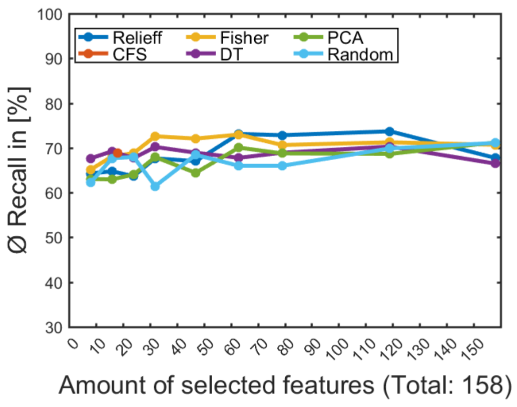
(a) SVM Accuracy



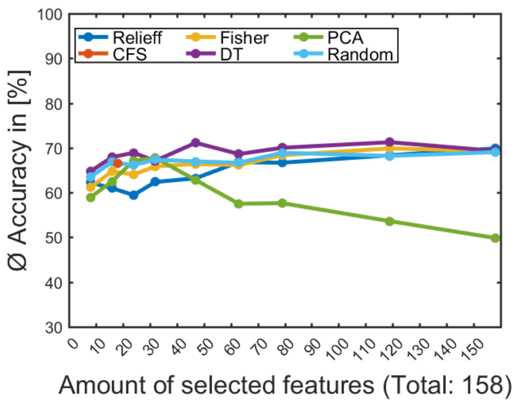
(b) SVM Recall



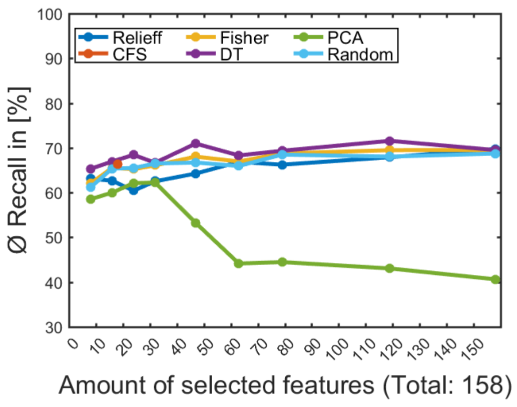
(c) ANN Accuracy



(d) ANN Recall

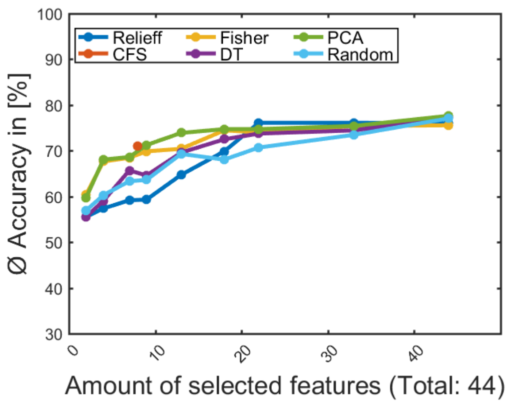


(e) RF Accuracy

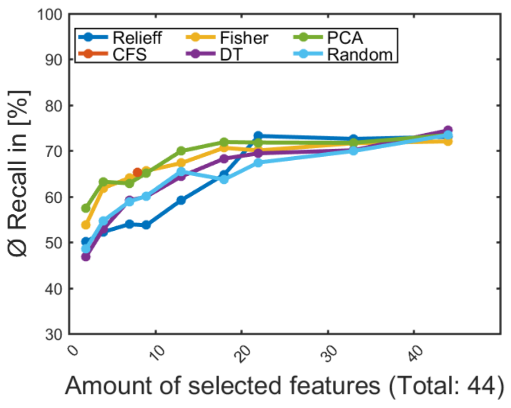


(f) RF Recall

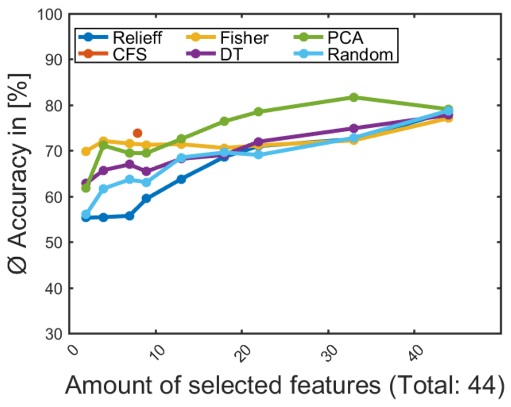
Figure 5.5. Comparison of feature selection methods for Dataset: Fabric/Featureset: All.



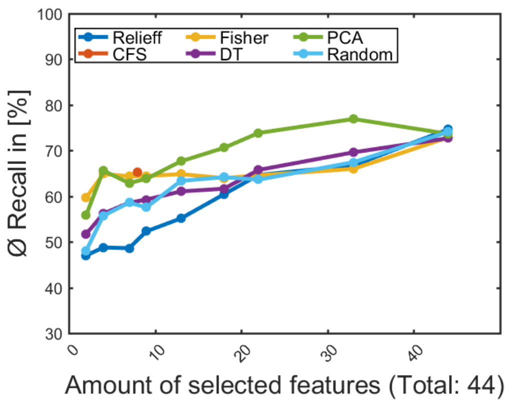
(a) SVM Accuracy



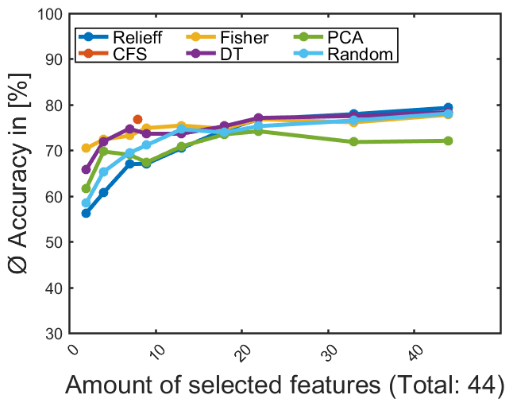
(b) SVM Recall



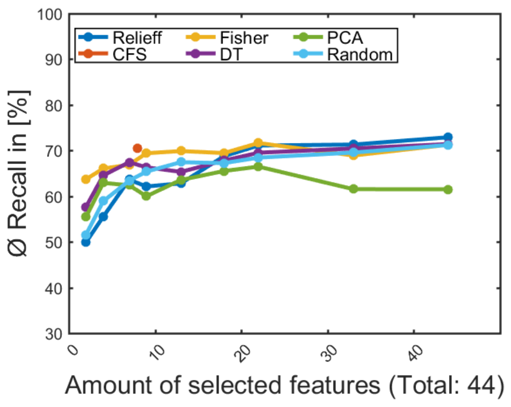
(c) ANN Accuracy



(d) ANN Recall

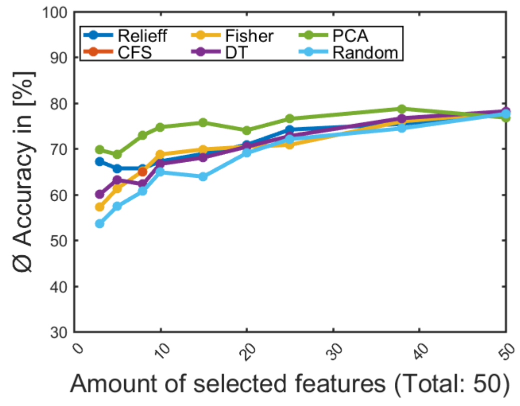


(e) RF Accuracy

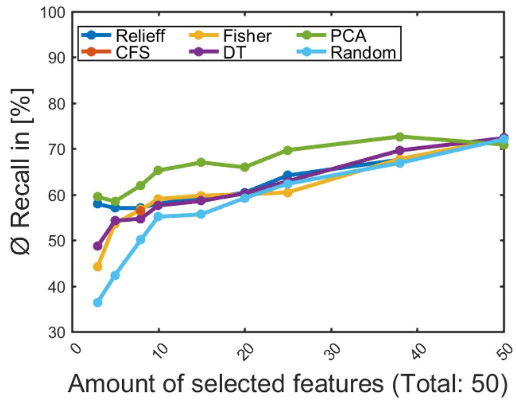


(f) RF Recall

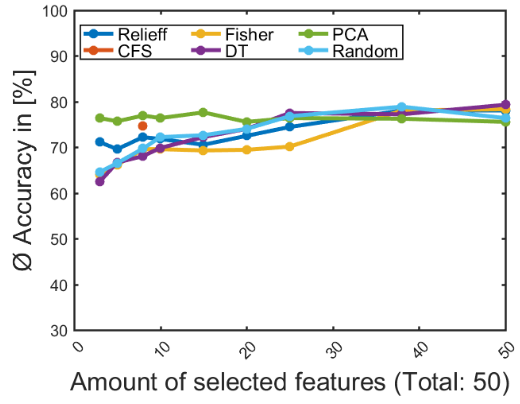
Figure 5.6. Comparison of feature selection methods for Dataset: UD/Featureset: Standard.



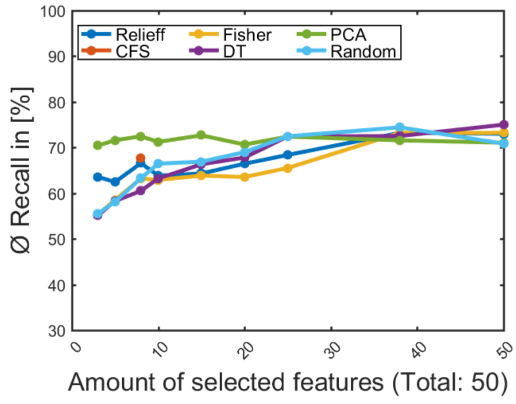
(a) SVM Accuracy



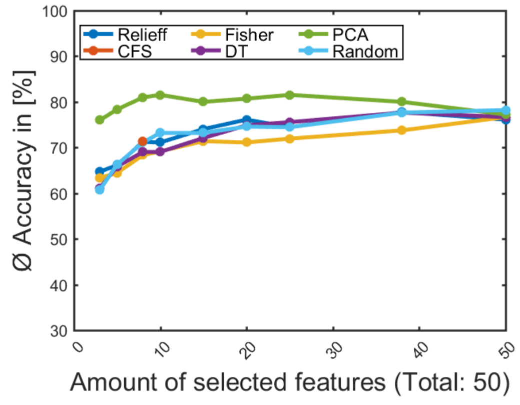
(b) SVM Recall



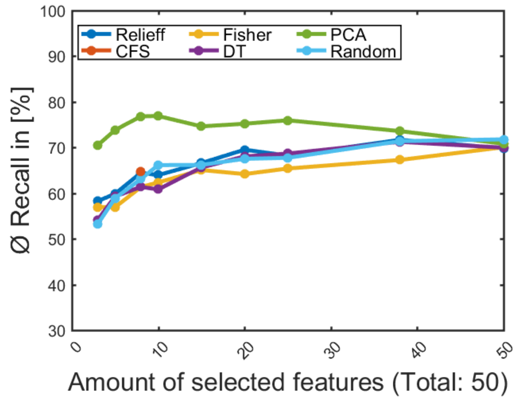
(c) ANN Accuracy



(d) ANN Recall

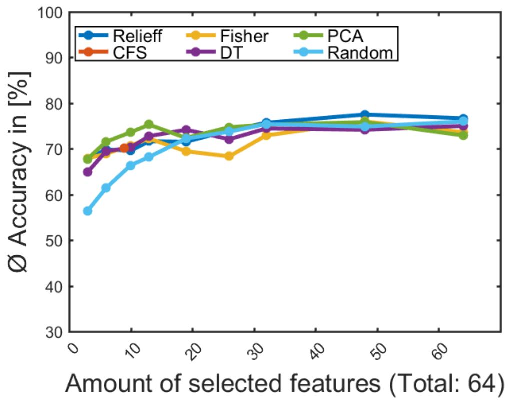


(e) RF Accuracy

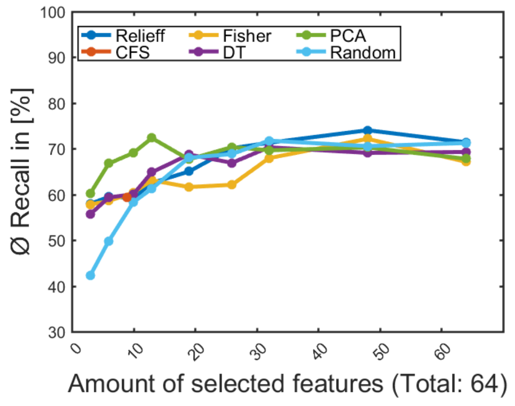


(f) RF Recall

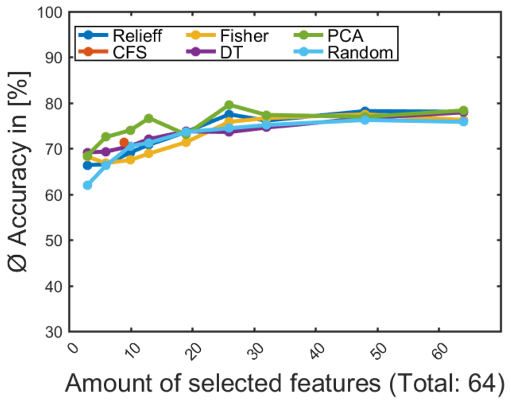
Figure 5.7. Comparison of feature selection methods for Dataset: UD/Featureset: PS6.



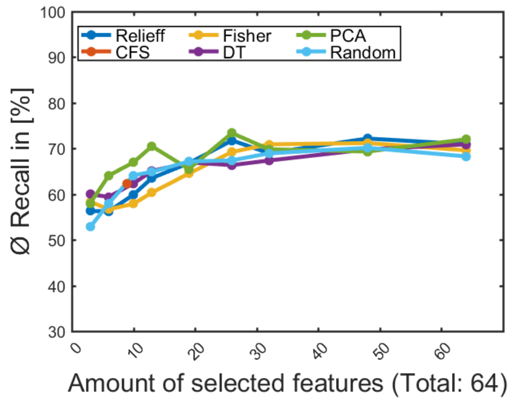
(a) SVM Accuracy



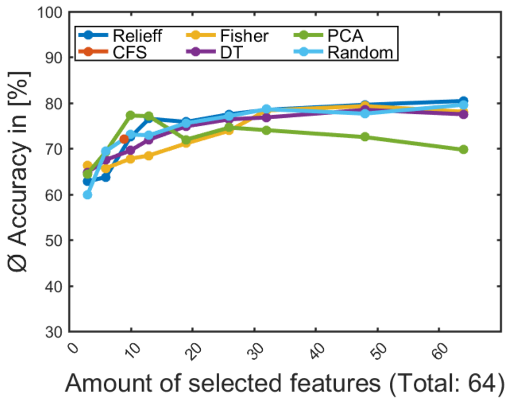
(b) SVM Recall



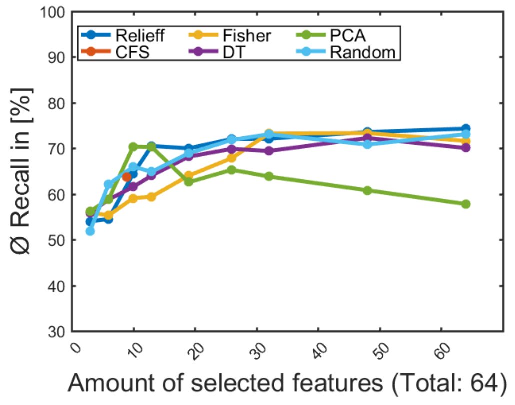
(c) ANN Accuracy



(d) ANN Recall

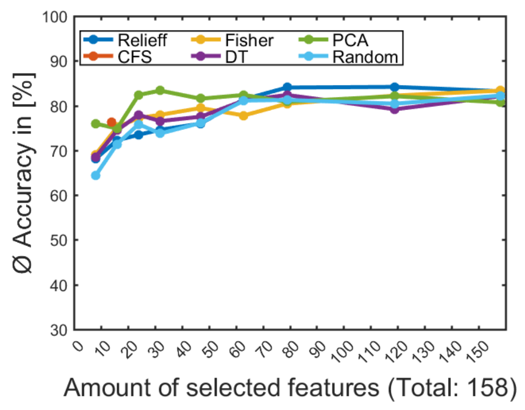


(e) RF Accuracy

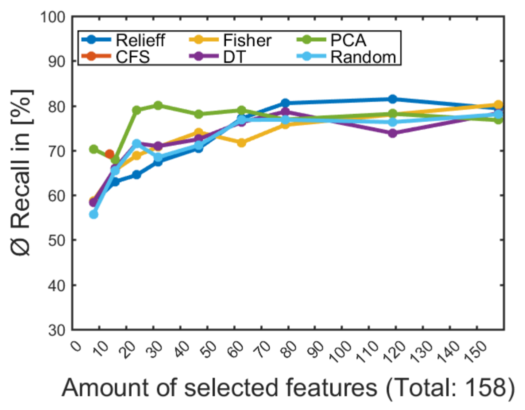


(f) RF Recall

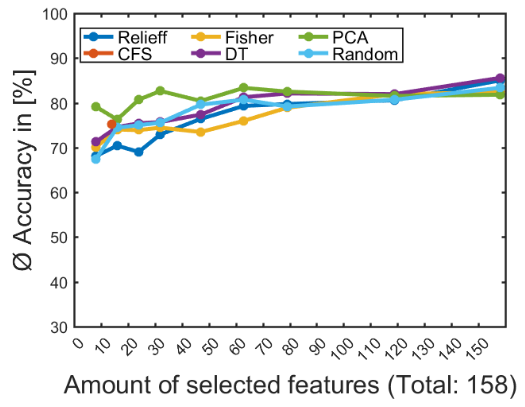
Figure 5.8. Comparison of feature selection methods for Dataset: UD/Featureset: PS8.



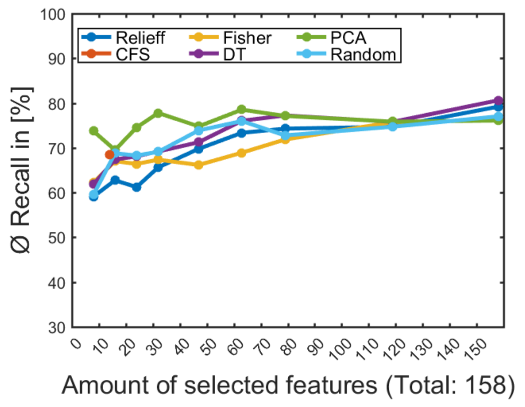
(a) SVM Accuracy



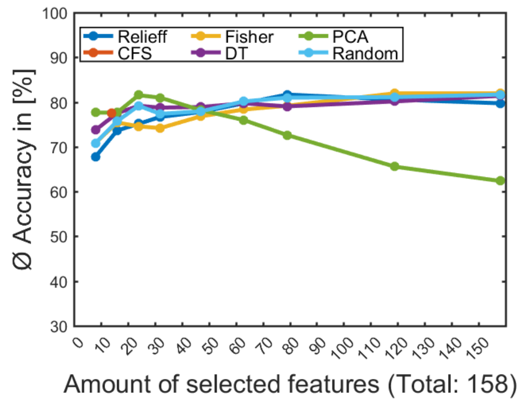
(b) SVM Recall



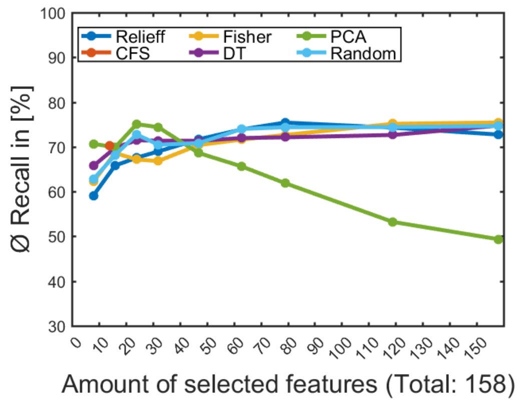
(c) ANN Accuracy



(d) ANN Recall



(e) RF Accuracy



(f) RF Recall

Figure 5.9. Comparison of feature selection methods for Dataset: UD/Featureset: All.

5.3 Evaluation of Relevant Features

In this section, the relevance of the individual features, based on the results of Section 5.2, is discussed. The relevance of a feature in this work is determined through counting the occurrences of every feature within the best ranked 40 % of the total features for every feature selection method (except for CFS where the single data point was used, and PCA) and for every dataset-featureset combination. Therefore, the highest possible value is 40 which is the amount of iterations (10) multiplied by the number of feature selections methods used for this evaluation (4). The assumption of this approach is that over all feature selections, the relevant features are picked more frequently than less important ones.
The results for these calculations are shown in Figure 5.10 to Figure 5.17 where the frequency of the individual features is plotted over their names. For a better visualization, only the highest ranked features are shown, except for the set with all features where also the least relevant ones are listed.

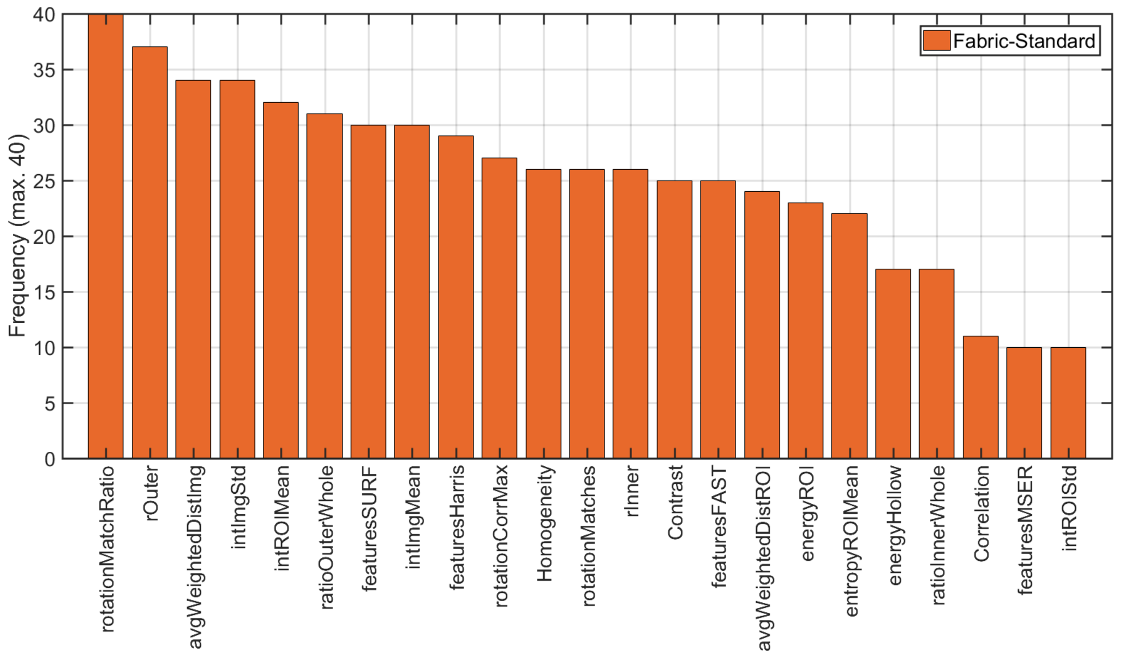


Figure 5.10. Relevance of features for Fabric-Standard

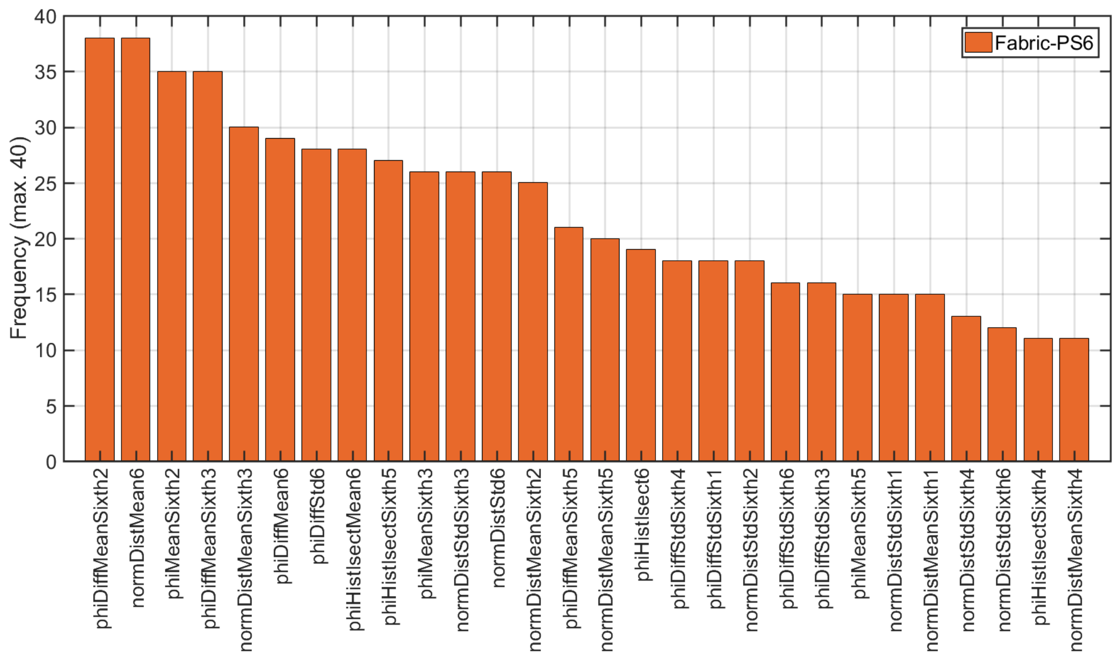


Figure 5.11. Relevance of features for Fabric-PS6

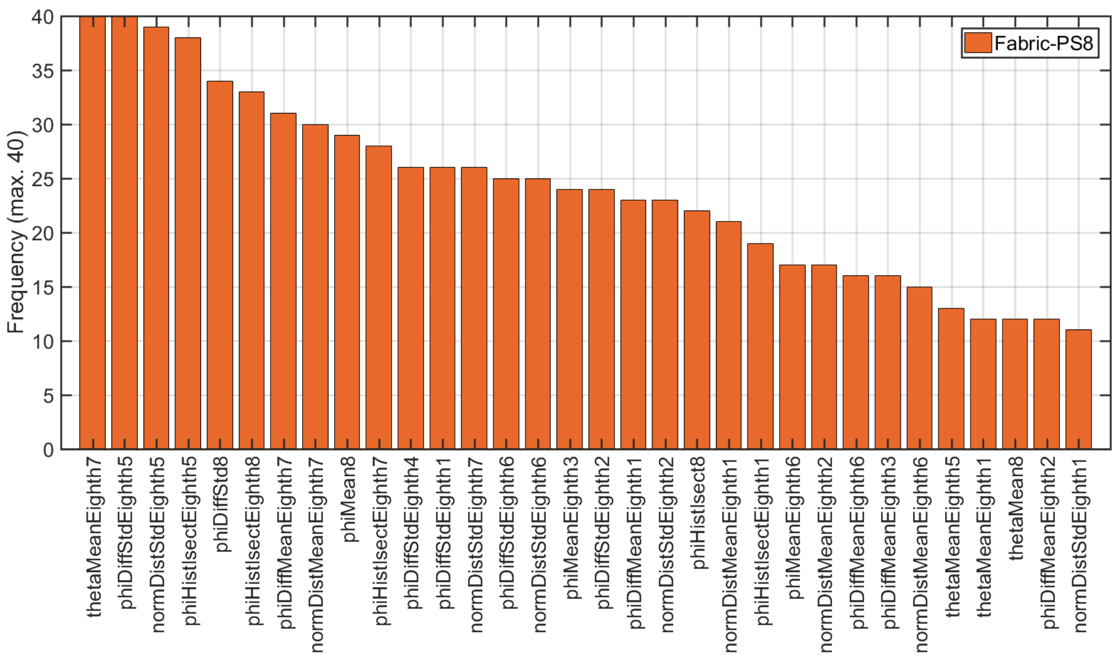


Figure 5.12. Relevance of features for Fabric-PS8

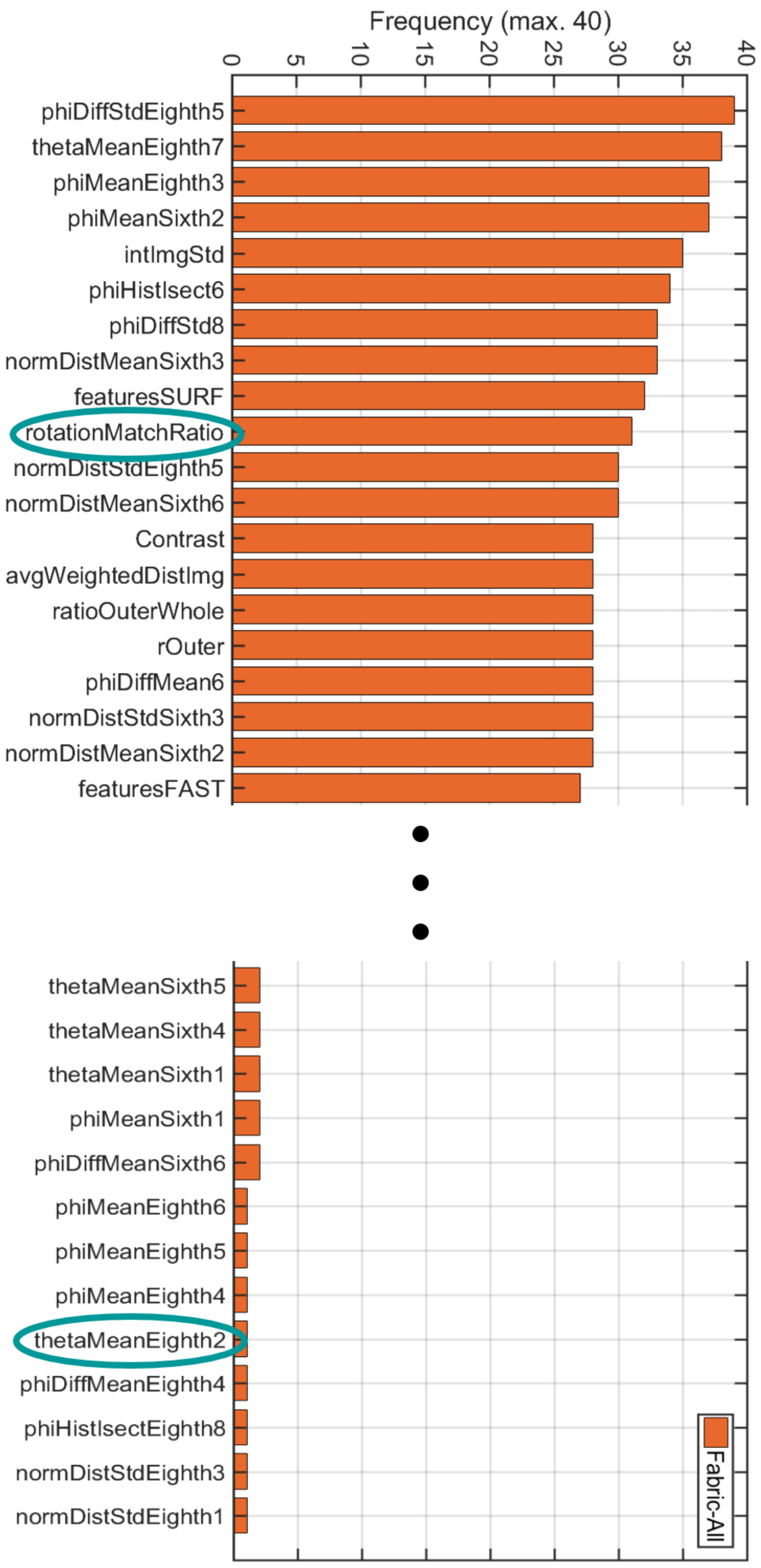


Figure 5.13. Relevance of features for Fabric-All

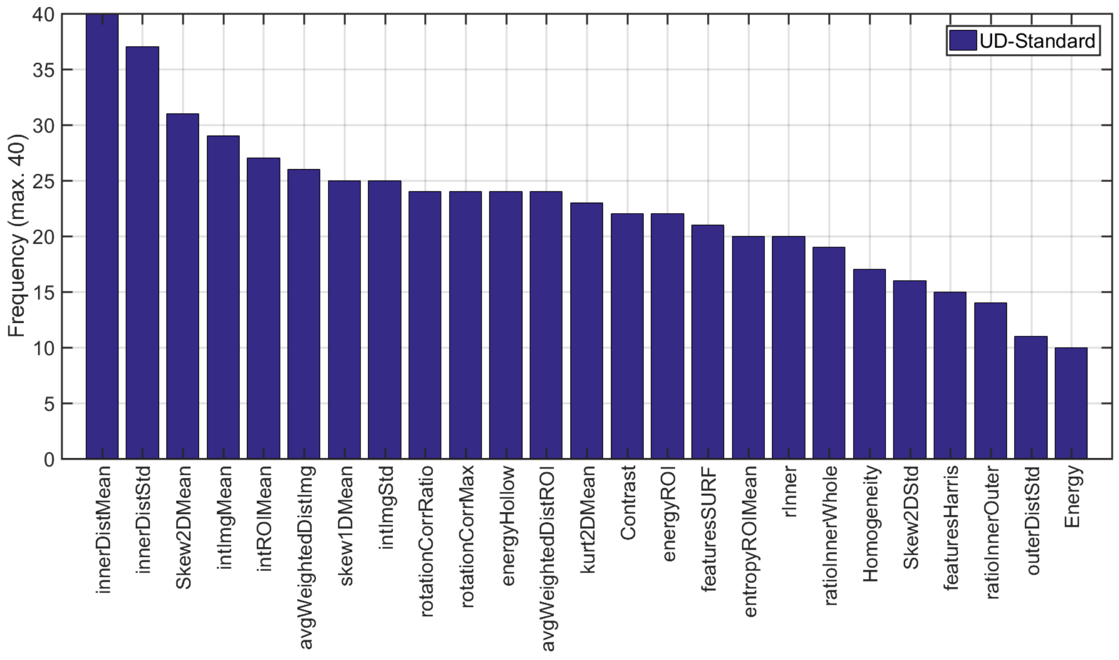


Figure 5.14. Relevance of features for UD-Standard

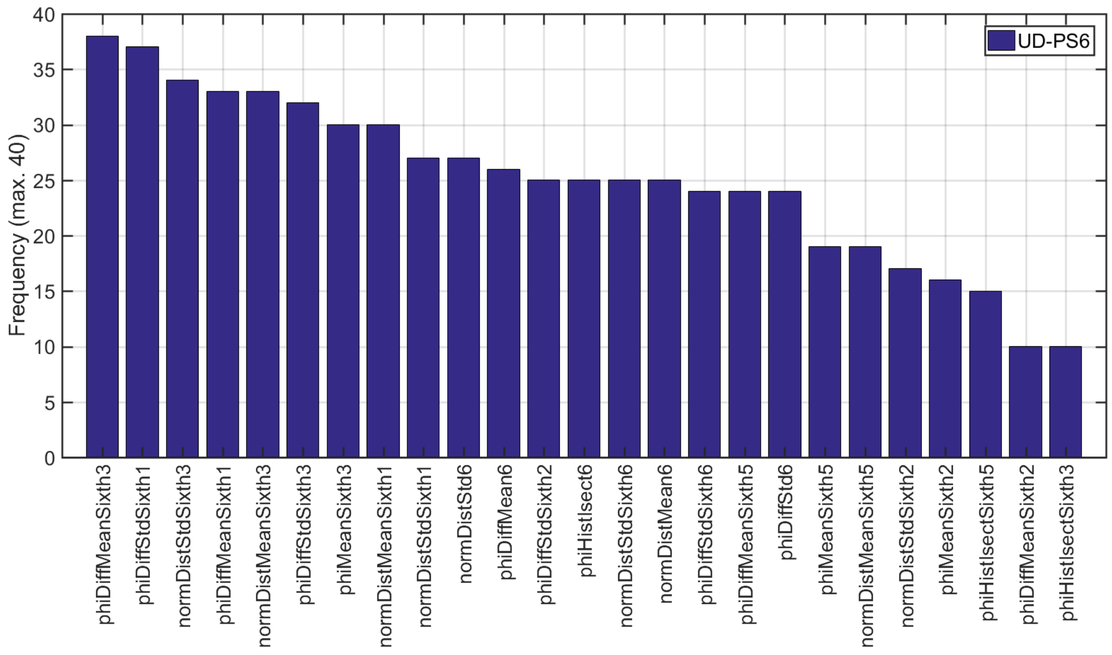


Figure 5.15. Relevance of features for UD-PS6

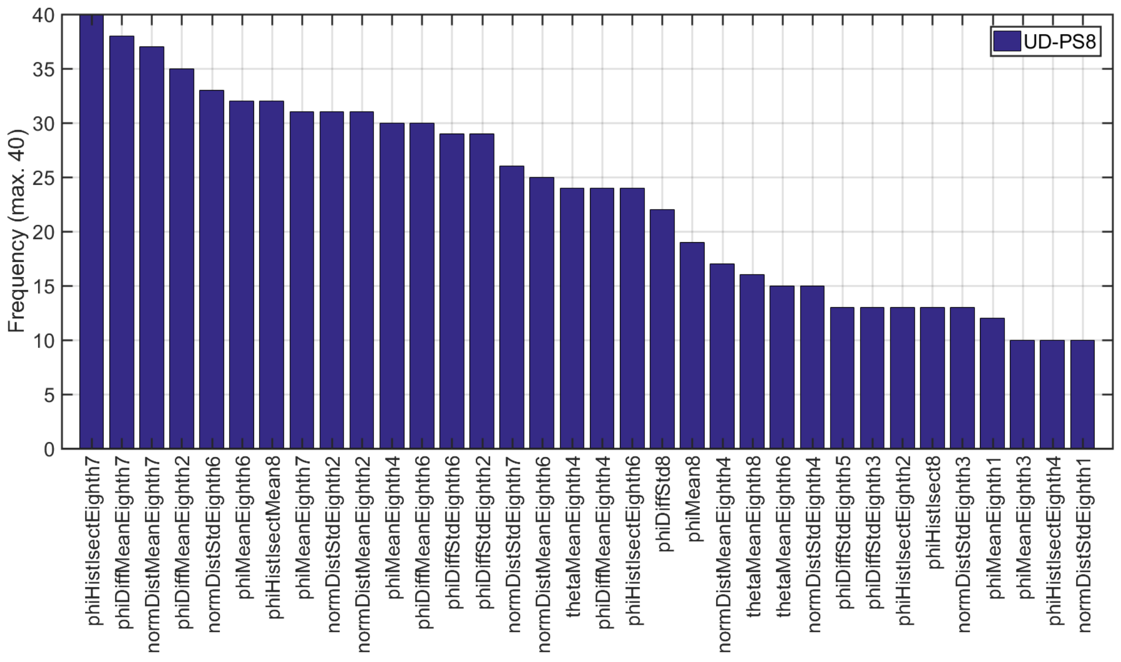


Figure 5.16. Relevance of features for UD-PS8

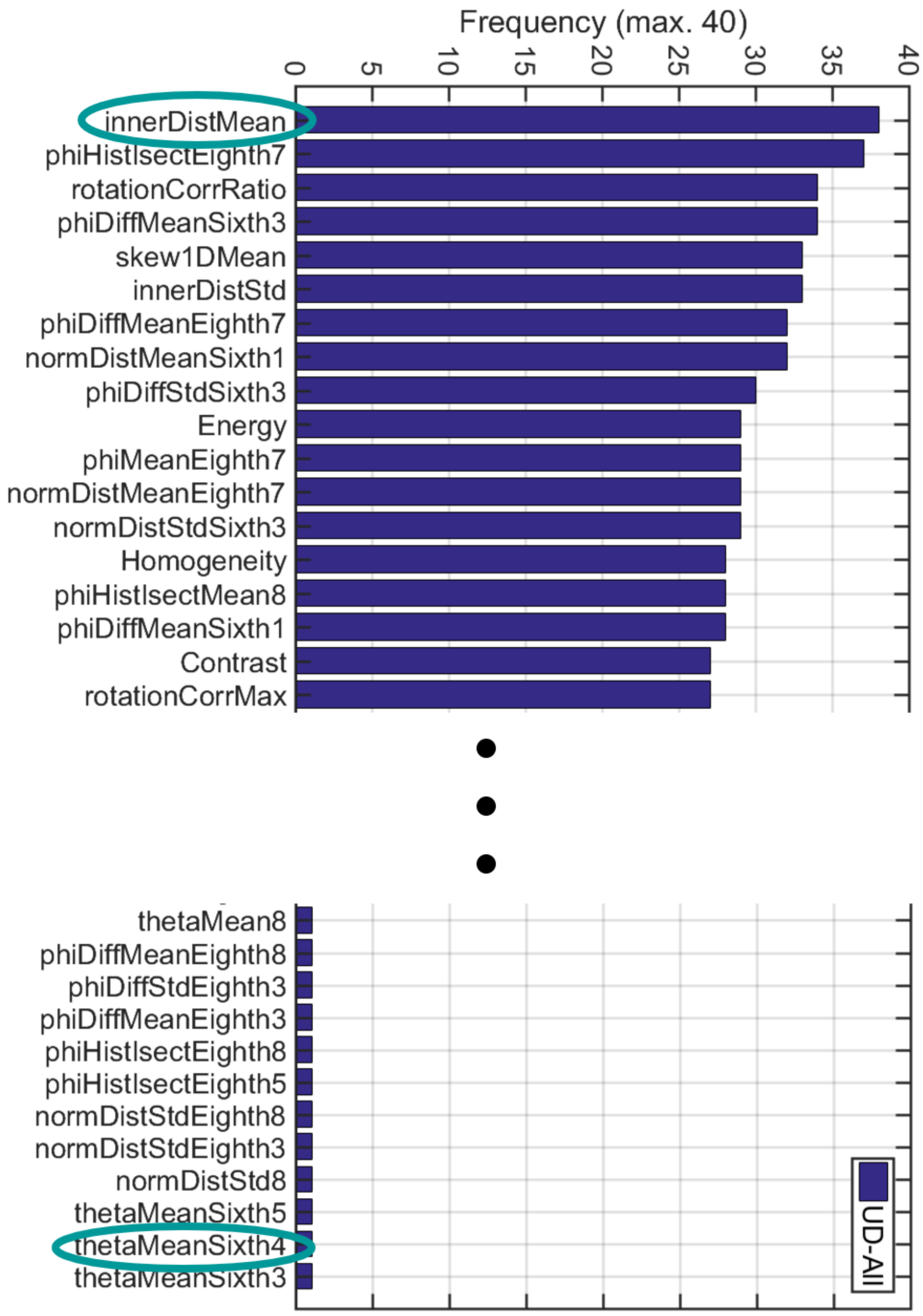


Figure 5.17. Relevance of features for UD-All

To evaluate the approach for ranking the features presented on the last few pages, the Figures 5.18 and 5.19 show the classification results plotted over the amount of selected features for both datasets. If a feature is not ranked by any method within the best 40 %, it is not considered in the evaluation. Thus, the total amount of features for each set differs from the original number.
The first feature that is selected is the most relevant feature which means it has the highest frequency according to Figure 5.13 for Fabric and 5.17 for UD. The higher the number of selected features is, the lower is the frequency of the features that are added to the selection.

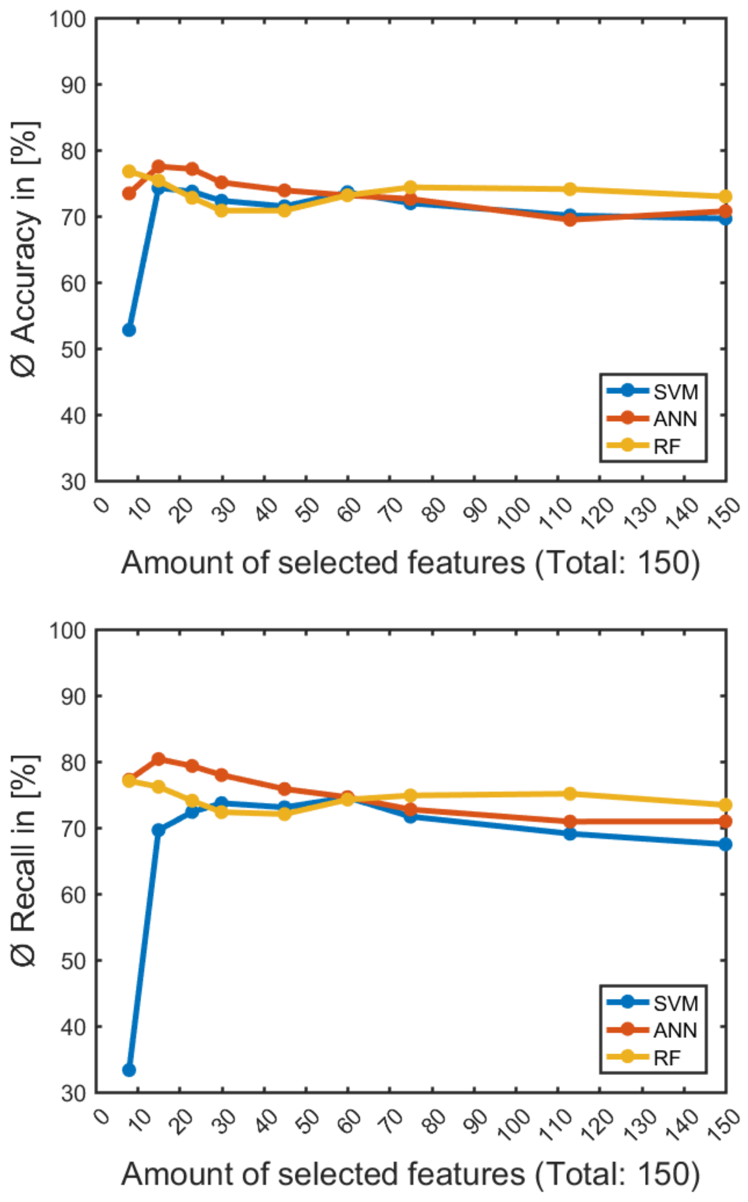


Figure 5.18. Evaluation of relevant features for Dataset: Fabric/Featureset: All.

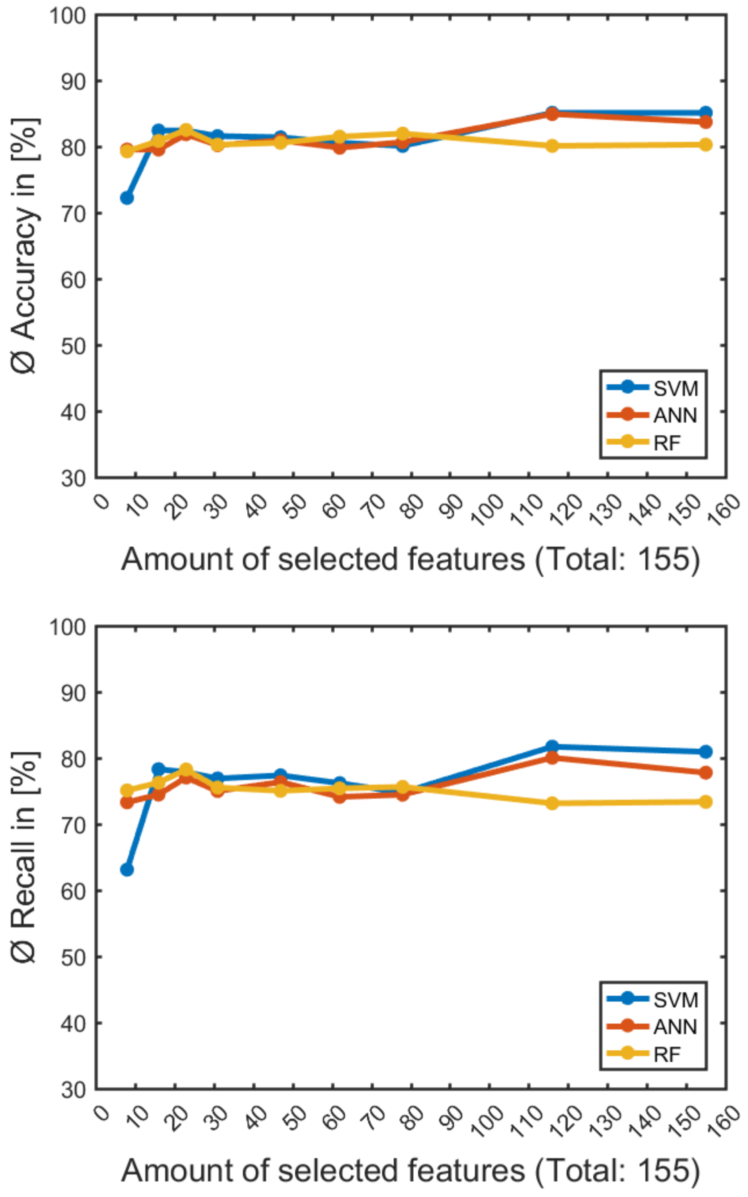


Figure 5.19. Evaluation of relevant features for Dataset: UD/Featureset: All.

Interpretation of results

From Figure 5.13, two features, for which the normalized values over the number of the observation are plotted in Figure 5.20 and Figure 5.21, are selected as examples (encircled).
In the first image, a quite important feature is shown, where a high variance between category 1 and category 3 is apparent. Therefore, this feature can be a good indicator whether an observation corresponds to class 1 or 3, respectively. In comparison, the second image shows a less relevant feature which is indicated by the lower variance between the categories. Thus, these two features are examples which encourage the assumption of the feature ranking approach.

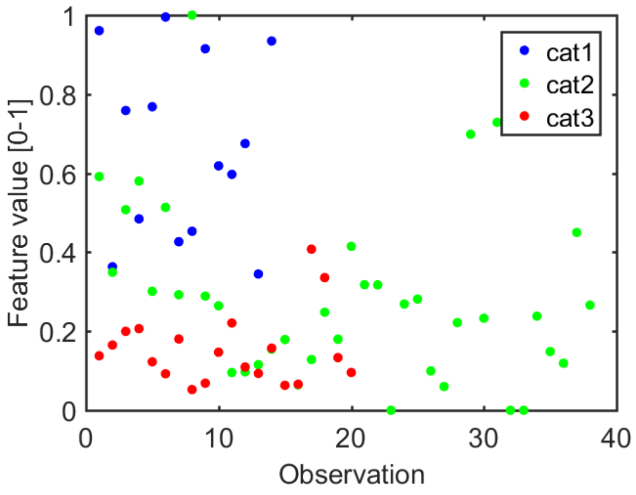


Figure 5.20. Sample data for the quite relevant feature *rotationMatchRatio* for the Fabric dataset.

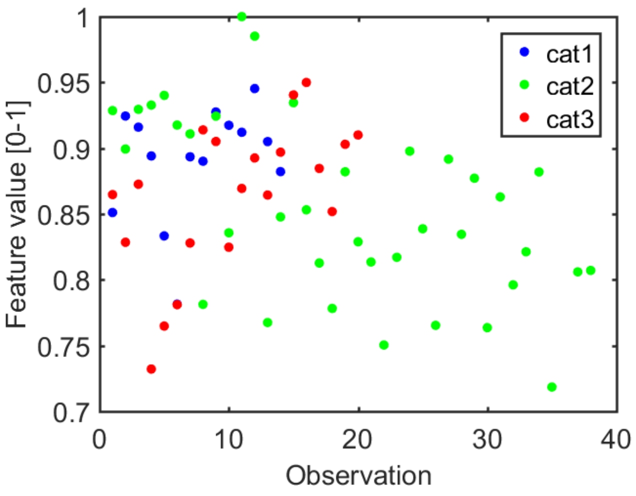


Figure 5.21. Sample data for the rather not relevant feature *thetaMeanEighth2* for the Fabric dataset.

As for the Fabric dataset, also for the UD dataset two features are selected exemplarily from Figure 5.17 for comparison, plotted in Figure 5.22 and Figure 5.23. The first image shows a feature ranked as relevant. The variance between the third class and the other two classes is very high, which probably is the reason that it is selected quite frequently. Nevertheless, the variance between class 1 and 2 is very low and thus the feature is only helpful for a distinction between class 3 and the rest.
In Figure 5.23, the interclass variance is rather low, therefore the feature is no good pick to distinguish between the classes.

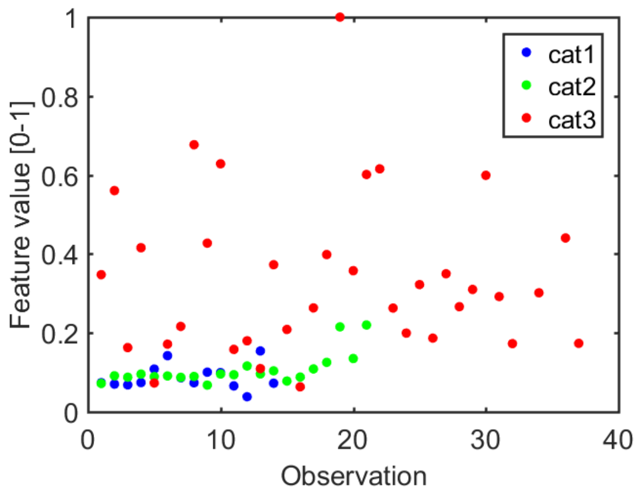


Figure 5.22. Sample data for the quite relevant feature *innerDistMean* for the UD dataset.

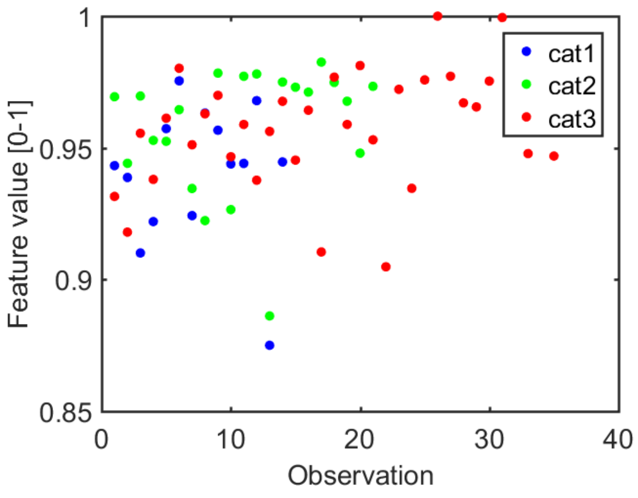


Figure 5.23. Sample data for the rather not relevant feature *thetaMeanSixth4* for the UD dataset.

It is also noteworthy that the results in Figure 5.10 to Figure 5.17 show that features which are within the top ranked for the feature subsets (Standard, PS6 and PS8) are also determined to be relevant for the whole feature set. In the experiments in the following sections, the results of the ranking is used to create a new set with the most relevant elements of the whole featureset, where relevant means that the frequency of the feature is higher than 24.
Regarding Figure 5.18 and Figure 5.19 an approximately horizontal or falling line in the graphs indicates a successful feature selection, as already mentioned earlier in this thesis. The graphs in evaluation show comparably good results when only a few of the features are selected and don't go up noticeably or even fall if the amount is increased. This finding shows that it is not only possible to reduce the dimensionality of the feature space while accepting minimal accuracy losses, but even to improve the prediction quality.

5.4 Evaluation of Balancing Methods

Section 5.1.1 lists the methods used in this work for dealing with the imbalanced datasets. In the current section, these methods are evaluated regarding the resulting classification accuracy and recall for both datasets, with all features and also only the relevant feature set, prepared in Section 5.3.
The following tables 5.4 and 5.5 compare the balancing outcomes for each classifier-featureset combination. The best results for accuracy and recall are highlighted for every row.
Additionally, confusion matrices in Figure 5.24 - Figure 5.29 show the results for the dataset balancing for the relevant features with SVM as classifier, to underline the influence of balancing on the prediction performance for the individual classes.

Method	**Weighting**		**SMOTE**		**w/o balance**	
	Acc.:	**Rec.:**	**Acc.:**	**Rec.:**	**Acc.:**	**Rec.:**
SVM/Rel.	75.4 %	76.3 %	74.8 %	76.5 %	**77.0 %**	**77.0 %**
SVM/All	66.2 %	66.2 %	**68.8 %**	**68.5 %**	67.1 %	66.5 %
ANN/Rel.	75.2 %	**77.6 %**	74.2 %	77.5 %	**76.8 %**	**77.6 %**
ANN/All	70.0 %	70.6 %	69.9 %	**70.9 %**	**70.1 %**	69.0 %
RF/Rel.	72.3 %	**74.3 %**	71.8 %	74.0 %	**72.9 %**	72.9 %
RF/All	71.0 %	**71.4 %**	69.6 %	70.2 %	**73.2 %**	71.1 %

Table 5.4: Comparison of balancing methods and classifiers for dataset Fabric.

Method	**Weighting**		**SMOTE**		**w/o balance**	
	Acc.:	**Rec.:**	**Acc.:**	**Rec.:**	**Acc.:**	**Rec.:**
SVM/Rel.	82.8 %	78.9 %	82.8 %	**79.8 %**	**82.9 %**	77.7 %
SVM/All	**85.7 %**	**81.6 %**	85.2 %	81.0 %	85.5 %	81.2 %
ANN/Rel.	80.7 %	76.5 %	**81.8 %**	**78.2 %**	79.2 %	72.9 %
ANN/All	83.3 %	77.7 %	**83.7 %**	**78.6 %**	83.3 %	77.7 %
RF/Rel.	81.7 %	**77.3 %**	**81.8 %**	77.2 %	81.7 %	77.2 %
RF/All	80.0 %	73.6 %	**81.6 %**	**76.0 %**	80.0 %	72.7 %

Table 5.5: Comparison of balancing methods and classifiers for dataset UD.

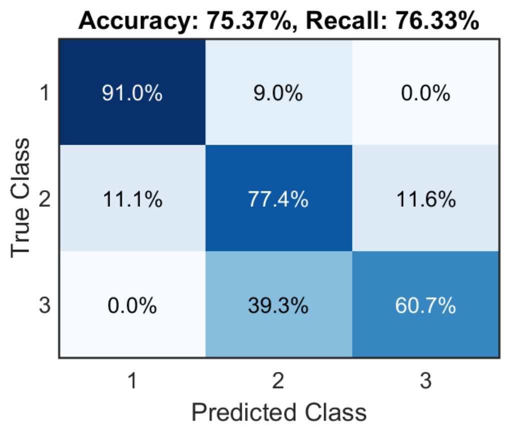


Figure 5.24. Evaluation of relevant features for
Dataset: Fabric/Balancing: Weights

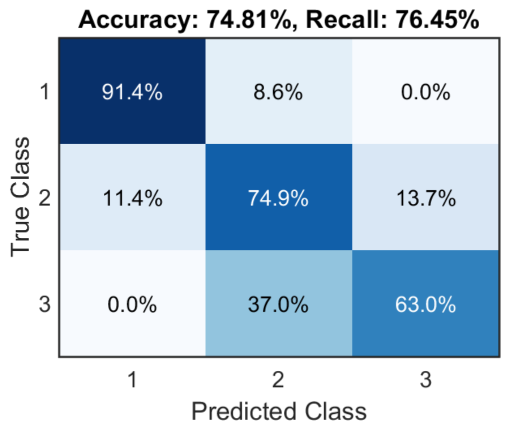


Figure 5.25. Evaluation of relevant features for
Dataset: Fabric/Balancing: SMOTE

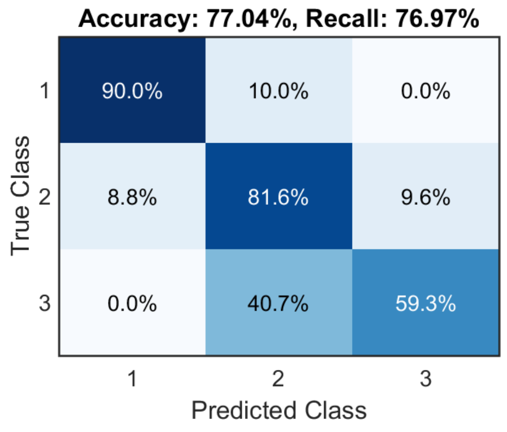


Figure 5.26. Evaluation of relevant features for
Dataset: Fabric/Balancing: -

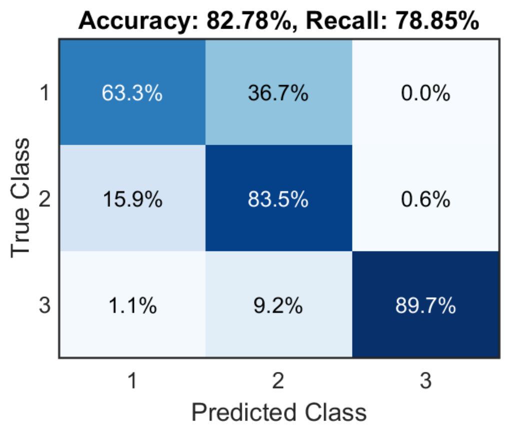


Figure 5.27. Evaluation of relevant features for Dataset: UD/Balancing: Weights

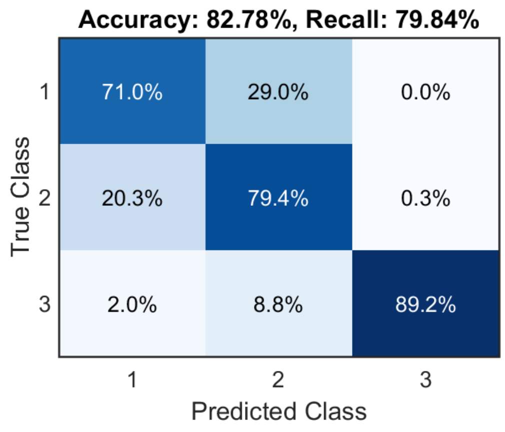


Figure 5.28. Evaluation of relevant features for Dataset: UD/Balancing: SMOTE

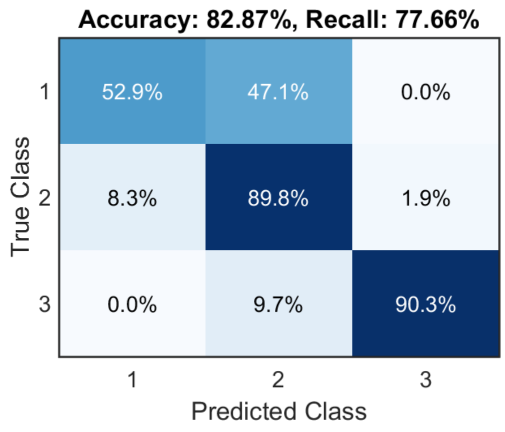


Figure 5.29. Evaluation of relevant features for Dataset: UD/Balancing: -

Interpretation of results

Regarding the results for the Fabric dataset, the number of correct predictions is the highest without any balancing method in all cases except one. As already pointed out earlier, the accuracy measure is a bit misleading for an imbalanced dataset and therefore the recall is a more accurate measure. It improves with balancing the datasets, while the accuracy decreases. Looking at the confusion matrices, the predictions for classes 1 and 3, which are the underrepresented classes for the Fabric dataset, are better with balancing. Looking at Table 5.4, it is also noticeable that the results for the relevant feature set are quite better than for the models with all features.
The results for the UD data are a bit different from Fabric, as the accuracy is in almost every case better for a balanced dataset. In addition, the improvements in recall are apparent, especially for SMOTE. The confusion matrices show that for category 1, the predictions are quite better with balancing methods. Comparing the featuresets, the results for the relevant features are slightly worse.
To sum up, the influence of balancing methods depend on the dataset, but in general they improve the prediction performance, especially for the underrepresented classes.

5.5 Evaluation of Classifiers

Finally, the classification algorithms are evaluated on the relevant features of both Fabric and UD dataset by using confusion matrices shown in Figure 5.30 to Figure 5.35. To balance the datasets, class-specific weighting is used.

Interpretation of results

Regarding the results for the Fabric dataset, the classifiers perform very similar. The predictions for the observations of category 1 are the most accurate, followed by category 2. Between classes 1 and 3, there are no confusions over all classifiers, which indicates a good separability of optimal boreholes (class 1) and samples where the feed during the drilling process was comparably high (class 3). The decision boundary for the classes 2 and 3 is not as distinct, which could be due to the fact that both classes share non-optimal drilling characteristics. Finally, the results regarding the separation between class 1 and 2 show that to a quite high extent it is possible to differ between observations with optimal and non-optimal drilling tools. Comparing the classifiers, the SVM and ANN perform similarly, whereas the Random Forest produces slightly worse results for this dataset.
For the UD dataset, the predictions are slightly better than for Fabric, especially for the random forest classifier. For this dataset, most of the samples from category 3 are classified correctly, whereas the results for class 1 are comparably bad. Nevertheless, the separability between 1 and 3 is again almost perfect. It is noticeable that a quite high percentage of observations from category 1 is predicted to be category 2 for all classifiers. This difference, compared to the Fabric dataset, probably origins from the drilling parameters as they are not exactly the same for both datasets.

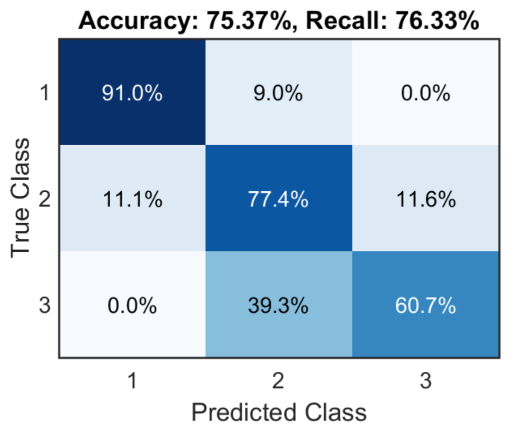


Figure 5.30. Evaluation of relevant features for Dataset: Fabric/Classifier: SVM

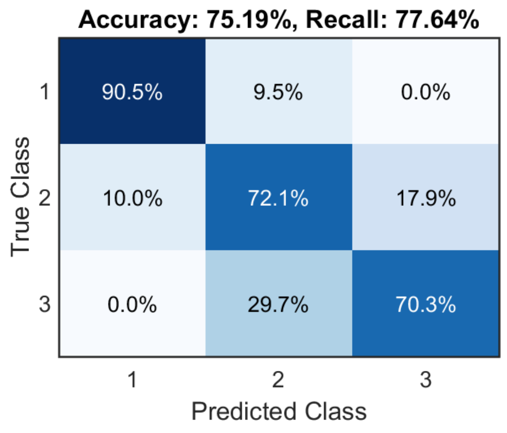


Figure 5.31. Evaluation of relevant features for Dataset: Fabric/Classifier: ANN

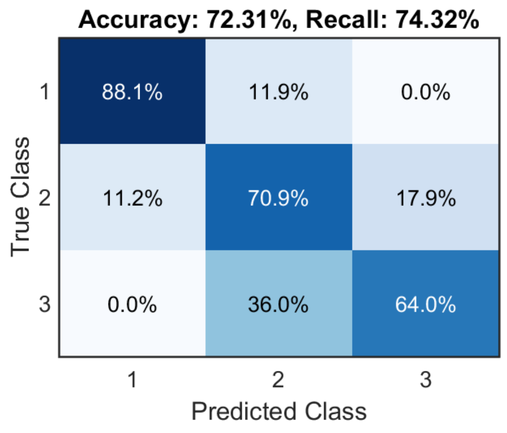


Figure 5.32. Evaluation of relevant features for Dataset: Fabric/Classifier: RF

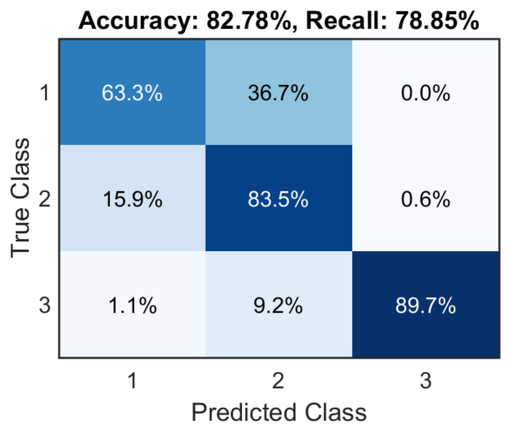


Figure 5.33. Evaluation of relevant features for Dataset: UD/Classifier: SVM

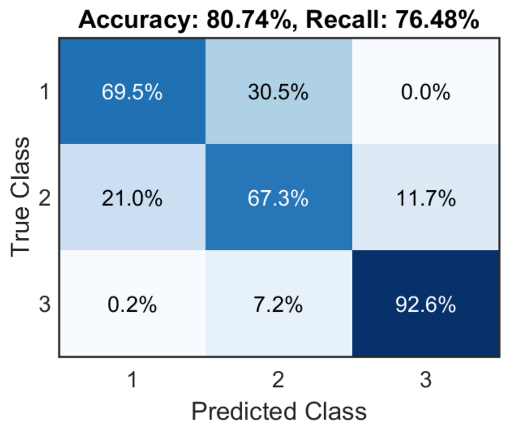


Figure 5.34. Evaluation of relevant features for Dataset: UD/Classifier: ANN

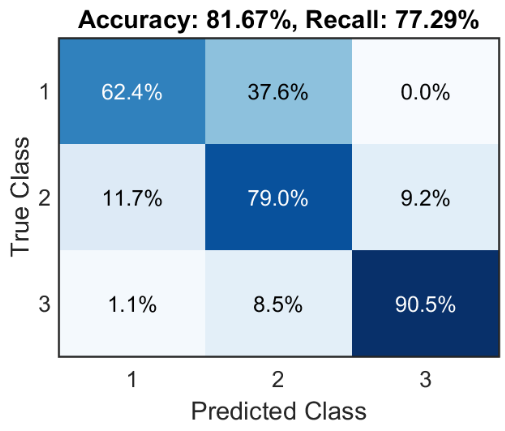


Figure 5.35. Evaluation of relevant features for Dataset: UD/Classifier: RF

6 Conclusion and Future Work

This chapter summarizes the findings of the feature extraction and the subsequent evaluation of the correlation of these features with the parameters in the drilling process. The most important results are mentioned, as well as possible further improvements for the different processing steps of this work.

6.1 Conclusion

In this thesis, we presented a machine learning approach to determine the correlation between surface features of boreholes in CFRP parts and the state of the tools and parameters values used in the drilling process. It is based on processing images of the insides of the bores, acquired by an endoscope with additional glass fiber illumination. From these images, information is extracted in terms of visual features which are then used as an input for the classification model. The result is a prediction for the quality of the drilling process and thus can be used as an indicator for replacing worn tools or adjusting drilling parameters.
We proposed several surface features for this application, for example textural features like the entropy of the image and higher level features, like corner detection. Additionaly, features were derived from a standard photometric stereo approach, which gives an estimate on the surface normals. These characteristics were later on evaluated regarding their relevance and informative content, using different feature selection approaches. In the end, the most relevant features were used to create the classification model. Based on the results of the machine learning process, we evaluated the feature selection approaches and prediction methods.

In general, the evaluations showed that with the results of the visual inspection of the boreholes, the predictability for the quality category of the drilling process is up to 77% for the Fabric dataset and even better for UD dataset, with slightly over 80%. The results vary throughout the three quality categories, as well as between the datasets. What is noticeable for both datasets, is the good separability between class 1 and 3 which indicates a very distinct classification boundary between optimal and very poor drilling conditions, like fractured tools or very high feed. The category 2, which lies in between the other categories regarding the drilling conditions, is sometimes confused with one of the other two classes, which is an indication for the correlation between the results of the visual inspection and the quality of the drilling process.
The results in Section 5.2 and Section 5.3 emphasize the importance of selecting relevant features, as they show an improvement of the classification performance

for the Fabric dataset over all classifiers. The predictions for the UD dataset were slightly worse for SVM and ANN, taking only the relevant features into account. Nevertheless, the relevant feature set only contains around 30 features, compared to the 158 in total, which underscores an important characteristic of the feature selection, the dimensionality reduction. This can already be achieved with a simple unsupervised method like PCA.
Regarding the proposed dataset balancing methods, the evaluations show the improvement in prediction performance for the individual classes over the imbalanced dataset.
The classification approaches perform very similar throughout the datasets and featuresets in evaluation. One slight difference is that the random forest improves for every dataset if only relevant features are used. As already mentioned in the previous chapter, this is due to the random feature subset selection, which negatively affects the performance if there are too many unimportant features in the set to choose from.
To sum up, it is possible to derive the quality of the drilling process to a certain extent from the resulting boreholes by means of visual inspection, using the proposed machine learning approach.

6.2 Future Work

The methods proposed in this work allow an acceptable correlation between surface features and parameters in the drilling process. Nevertheless, to implement this approach into an industrial inspection application, some prediction performance improvements might be necessary.
A possible starting point for future work could be the photometric stereo approach which in its present implementation is not very accurate regarding the exact surface normals. The reason for this is that on the one hand, the assumption of diffuse reflection of the CFRP material is not realistic. On the other hand, it is not easy to take the influence of the differences in glassfiber illumination into account, which are for example varying fiber cutting angles or diameters. In addition, the dimensions of the imaging geometry are very small, lying in the range of only a few millimeters, which restricts the possibilities for photometric stereo. To improve the proposed method, different approaches for the reflective characteristics of the material could be evaluated. In addition, the effect of the imaging geometry could be examined more detailed.
On top of the approaches for feature extraction presented in this thesis, future work could evaluate more complex features, like detecting certain geometric shapes in the images due to fraying or delamination, though, here the problem might be that the exact shape of such defects is rather random, which would require a more complex implementation.
With the proposed improvements, the correlation of surface features with the quality of the drilling process will be an important step towards preventing defective boreholes in the first place, in order to accomplish the goal of ZDM.

Bibliography

[1] PROFACTOR GmbH. (2018), [Online]. Available: `https://www.profactor.at/`.

[2] J. F. Halpin, *Zero Defects: A New Dimension in Quality Assurance.* New York City: McGraw-Hill, 1966.

[3] PROFACTOR GmbH. (2018). Hscan, [Online]. Available: `https://www.profactor.at/hscan/`.

[4] T. S. Newman and A. K. Jain, "A Survey of Automated Visual Inspection," in *Computer Vision and Image Understanding*, vol. 61, 1995, pp. 231–262.

[5] F.-C. Tien, C.-H. Yeh, and K.-H. Hsieh, "Automated visual inspection for microdrills in printed circuit board production," in *International Journal of Production Research*, vol. 42, 2004, pp. 2477–2495.

[6] W.-C. Wang, S.-L. Chen, L.-B. Chen, and W.-J. Chang, "A Machine Vision Based Automatic Optical Inspection System for Measuring Drilling Quality of Printed Circuit Boards," in *IEEE Access*, vol. 5, 2017, pp. 10 817–10 833.

[7] X.-W. Zhang, Y.-Q. Ding, Y.-y. Lv, A.-y. Shi, and R.-y. Liang, "A vision inspection system for the surface defects of strongly reflected metal based on multi-class SVM," in *Expert Systems with Applications*, vol. 38, 2011, pp. 5930–5939.

[8] R. Schmitt, T. Fürtjes, B. Abbas, P. Abel, W. Kimmelmann, P. Kosse, and A. Buratti, "Real-Time Machine-Vision-System for an Automated Quality Monitoring in Mass Production of Multiaxial Non-crimp Fabrics," 2016, pp. 769–782.

[9] R. Woodham, "Photometric Method for Determining Surface Orientation from Multiple Images," in *Optical Engineering*, vol. 19, 1992.

[10] L. Wang, K. Xu, and P. Zhou, "Online Detection Technique of 3D Defects for Steel Strips Based on Photometric Stereo," in *International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, vol. 8, 2016, pp. 428–432.

[11] W. Palfinger, S. Thumfart, and C Eitzinger, "Photometric stereo on carbon fiber surfaces," 2011.

[12] S. H. Chen and D. B. Perng, "A Molding Surface Auto-Inspection System," in *International Journal of Industrial and Manufacturing Engineering*, vol. 8, 2014.

[13] D. Weimer, B. Scholz-Reiter, and M. Shpitalni, “Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection,” in *CIRP Annals - Manufacturing Technology*, vol. 65, 2016.

[14] D. Weimer, H. Thamer, and B. Scholz-Reiter, “Learning Defect Classifiers for Textured Surfaces Using Neural Networks and Statistical Feature Representations,” in *Procedia CIRP*, vol. 7, 2013, pp. 347–352.

[15] S. Ravikumar, K. I. Ramachandran, and V. Sugumaran, “Machine learning approach for automated visual inspection of machine components,” in *Expert Systems with Applications*, vol. 38, 2011, pp. 3260–3266.

[16] J.-K. Park, B.-K. Kwon, J.-H. Park, and D.-J. Kang, “Machine Learning-Based Imaging System for Surface Defect Inspection,” in *International Journal of Precision Engineering and Manufacturing-Green Technology*, vol. 3, 2016, pp. 303–310.

[17] Y.-J. Cha, W. Choi, and O. Büyüköztürk, “Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks,” in *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, 2017, pp. 361–378.

[18] W. Ford, “Chapter 16 - Least-Squares Problems,” in *Numerical Linear Algebra with Applications*, Boston: Academic Press, 2015, pp. 321 –349.

[19] M. Sugiyama, “Chapter 22 - Least Squares Regression,” in *Introduction to Statistical Machine Learning*, Boston: Morgan Kaufmann, 2016, pp. 245 –256.

[20] R. Szeliski, *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag, 2010.

[21] B. Horn, “Obtaining Shape from Shading Information,” in *Shape from Shading*, 1989, pp. 123–171.

[22] U. Staczyk, B. Zielosko, and L. C. Jain, *Advances in Feature Selection for Data and Pattern Recognition*, 1st ed. Springer Publishing Company, 2017.

[23] L. Breiman, “Random Forests,” in *Machine Learning*, vol. 45, 2001.

[24] I. Kononenko, E. Šimec, and M. Robnik-Šikonja, “Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF,” in *Applied Intelligence*, 1997.

[25] K. Kira and L. A. Rendell, “The Feature Selection Problem: Traditional Methods and a New Algorithm.,” in *Proceedings 10th National Conference on Artificial Intelligence*, 1992, pp. 129–134.

[26] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. John Wiley & Sons, New York, 2001.

[27] Q. Gu, Z. Li, and J. Han, “Generalized Fisher Score for Feature Selection,” in *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 2011, pp. 266–273.

[28] M. A. Hall and L. A. Smith, "Feature Selection for Machine Learning: Comparing a Correlation-based Filter Approach to the Wrapper," in *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, 1999, pp. 235–239.

[29] M. A. Hall, "Correlation-Based Feature Selection for Machine Learning," PhD thesis, Department of Computer Science, University of Waikato, 1999.

[30] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees.* Taylor & Francis, 1984.

[31] H. Bittencourt and R. Clarke, "Feature Selection by using Classification and Regression Trees (CART)," 2004.

[32] C. Gini, "Variabilità e mutabilità," in *Memorie di Metodologica Statistica*, 1912.

[33] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," in *Philosophical Magazine*, vol. 2, 1901, pp. 559–572.

[34] H. Hotelling, "Analysis of a Complex of Statistical Variables into Components," in *Journal of Educational Psychology*, vol. 24, 1933, pp. 417–441.

[35] I. Jolliffe, *Principal Component Analysis*, 2nd ed. Springer, 2002.

[36] A. Tharwat, "Principal Component Analysis - a Tutorial," in *International Journal of Applied Pattern Recognition*, vol. 3, 2016.

[37] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version*, 11th ed. Wiley Global Education, 2013.

[38] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed. Prentice Hall, 2011.

[39] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, 2nd ed. Springer, 2009.

[40] G. Bonaccorso, *Machine Learning Algorithms: A Reference Guide to Popular Algorithms for Data Science and Machine Learning.* Packt Publishing, 2017.

[41] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms.* New York, USA: Cambridge University Press, 2014.

[42] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," in *Journal of Arfiticial Intelligence Research*, vol. 16, 2002, pp. 321–357.

[43] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, 1979.

[44] K. Zuiderveld, "Contrast Limited Adaptive Histograph Equalization," in *Graphic Gems IV*, 1994, pp. 474–485.

[45] E. Rosten and T. Drummond, "Machine Learning for High Speed Corner Detection," in *9th European Conference on Computer Vision*, vol. 1, 2006, pp. 430–443.

[46] E. Rosten, R. Porter, and T. Drummond, "Faster and Better: A Machine Learning Approach to Corner Detection," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, 2010, pp. 105–119.

[47] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[48] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF:Speeded Up Robust Features," in *Computer Vision and Image Understanding (CVIU)*, vol. 110, 2008, pp. 346–359.

[49] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," in *Int. J. Comput. Vision*, vol. 60, 2004, pp. 91–110.

[50] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[51] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions," in *Proceedings of British Machine Vision Conference*, 2002, pp. 384–396.

[52] M. Donoser and H. Bischof, "Efficient Maximally Stable Extremal Region (MSER) Tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 553–560.

[53] R. K. McConnell, "Method of and Apparatus for Pattern Recognition," in *U. S. Patent Nr. 4,567,610*, 1986.

[54] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 886–893.

[55] Z. Zhao, S. Sharma, A. Anand, F. Morstatter, S. Alelyani, and H. Liu, "Advancing Feature Selection Research - ASU Feature Selection Repository," 2010.

[56] F. Chollet. (2015). Keras, [Online]. Available: `https://keras.io`.

Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit gemäß dem Code of Conduct, insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien, am 7. März 2019

Johannes Huemer