**Technische Universität Ilmenau**
Fakultät für Informatik und Automatisierung
Fachgebiet Neuroinformatik und Kognitive Robotik

# Novelty Detection zur Schadstellenklassifikation auf Strassendaten
# Bachelorarbeit am FG Neuroinformatik und Kognitive Robotik

Bachelorarbeit zur Erlangung des akademischen Grades Bachelor of Science

**Ghalia Rehawi**

Betreuer: Dipl.-Inf Ronny Stricker
Verantwortlicher Hochschullehrer:
Prof. Dr. H.-M. Groß, FG Neuroinformatik und Kognitive Robotik

Die Bachelorarbeit wurde am 16.10.2017 bei der Fakultät für Informatik und Automatisierung der Technischen Universität Ilmenau eingereicht.

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

October 15, 2017 Ghalia Rehawi

# Acknowledgments

I would like first and foremost to thank my supervisor Dipl.-Inf. Ronny Stricker for giving me the opportunity of exploring such an interesting topic and supporting me with resource needed to continue this work.
I also would like to thank my parents Rwida Sawah and Maher Rehawi for their endless love and support, I could not thank you enough my guarding angels. To my two beautiful sisters Leen Rehawi and Julia Rehawi, thank you for the joy you bring to my life and for always being there for me. To my Love Hasan, your faith in me pushes me forward. Thank you for all the help and support you gave me and for bearing with me through bad times. Thank you to all my friends who helped in any small or big matter.

# Contents

# List of Figures

# Chapter 1

# Introduction and Background

Due to constant usage and heavy machinery, roads are suffering from poor pavement conditions, therefore road maintenance is an essential task in order to ensure a safe road network. Periodically, data about surface distresses is collected and inspected to identify the severity of different classes of pavement distresses. Traditionally this process is done by human operators during visual inspection sessions but this method is time consuming, costly, requires extensive human labor and unable to provide meaningful quantitative information. In the last decade automatic pavement distress detection and characterization systems have been developed, which increased the speed and efficiency of the pavement analysis process [Eisenbach et al., 2017], [Daniel and V, 2014] and others, see chapter 2. These automated systems overcome manual errors providing better outcome comparatively.

## 1.1 ASINVOS

This thesis is part of the ASINVOS project (Assistierendes und interaktiv lernfähiges Videoinspektionssystem für Oberflächenstrukturen am beispiel von Strassenbelägen und Rohrleitungen) in the department of Neuroinformatics and Cognitive Robotics Lab in TU Ilmenau [NICR, 2016]. The aim of the ASINVOS project is to automate to a high degree the process of distress detection in streets and drinking water pipes by applying machine learning techniques. The basic idea is to train a self learning

system with manually labeled data from previous inspections and by doing so the system will learn to recognize underlying patterns of distress. Once the system is able to robustly identify intact infrastructure it can reduce the human amount of work by presenting only distress candidates to the operator. This helps to significantly speed up the inspection process and simultaneously reduces the costs.

## 1.2 Purpose

The core purpose of this thesis is the comparison between the three most used novelty detection (discussed later) methods in the literature namely, Gaussian mixture models, one-class support vectors machine and k-nearest neighbors. Also different feature extraction methods are discussed and compared. The three methods are trained on the GAPs data set (The German Asphalt Pavement Distress), [Eisenbach et al., 2017], using only the intact data with the goal of detecting distress in the images. GAPs is the first freely available pavement distress dataset providing high quality images recorded by a standardized process fulfilling German federal regulations. This will provide a base for a fair comparison of researches in this field.

## 1.3 GAPs Dataset

The data presented in this paper, have been captured by the mobile mapping system S.T.I.E.R figure (1.1), which is manufactured and operated by the German engineering company LEHMANN+ PARTNER GmbH. The main components of S.T.I.E.R are an inertial navigation system, laser sensors for evenness and texture measurements, a 2D laser range finder and different camera systems for capturing both the vehicles environment and the pavement's surface [Eisenbach et al., 2017]. The data captured complies with the high German quality standards in the field of Road Monitoring and Assessment (RMA). The RMA process standardizes data acquisition and provides uniform parameters to ensure objective analyses of surface conditions. The relevant damage classes presented in the dataset are cracks, potholes, inlaid patches, applied patches and open joints. Cracks are the dominant damage class. This class comprises

all sorts of cracks like single/multiple cracking, longitudinal/transversal cracking, alligator cracking, and sealed/filled cracks. Figure (1.2) shows an example of such distress types from the dataset.

**Figure 1.1.** Mobile mapping system S.T.I.E.R [Eisenbach et al., 2017]

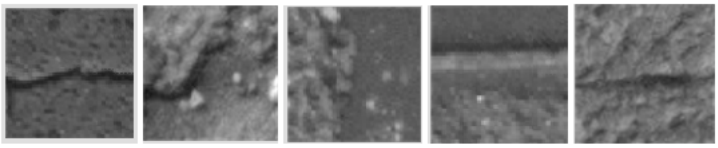


**Figure 1.2.** Distress types presented in the dataset, from left to right: crack, pothole, applied-patch, inlaid-patch and open-joints.

Moreover, the images in the dataset contain pavement of three different German federal roads. Images of two German federal roads characterized with poor pavement condition are used for training and a section of one of these roads is used for validation. For testing a third road with better pavement condition was used. The images used in this thesis are gray valued 64x64 pixels of the original GAPs images of resolution 1920x1080 pixels and the dataset is divided into Training-,Validation- and Test-data. The training

dataset contains 50k images with 30k images of intact road surface labeled '0' and 20k images of defected road surface labeled '1'. The validation and test dataset are both of size 10K with 6k images labeled '0' (intact road surface) and 4k images labeled '1'. The type of distress in the images is neglected in this work since the aim is only the detection of distress and not the classification of different distress types.

## 1.4 Challenges

As mentioned above we will be using Novelty Detection approach to detect distress in the images, that means the training is done only using data of the over represented class (see chapter 3), in our case it is class '0' containing images of only intact road surface. Throughout this thesis we will be using the term normal data to refer to this class and the term novel (or abnormal) data to refer to class '1'. In the literature there have been no reported research using Novelty techniques for road distress detection. Moreover the GAPs dataset contains a very complex and high textured images. Using normal image processing techniques and training using only normal samples is a challenging task. As mentioned above images of test dataset are extracted from a different road than the validation and training dataset, this means that we should expect a degradation in the performance on test samples even using powerful techniques such as deep learning for the distress detection, see [Eisenbach et al., 2017].

# Chapter 2

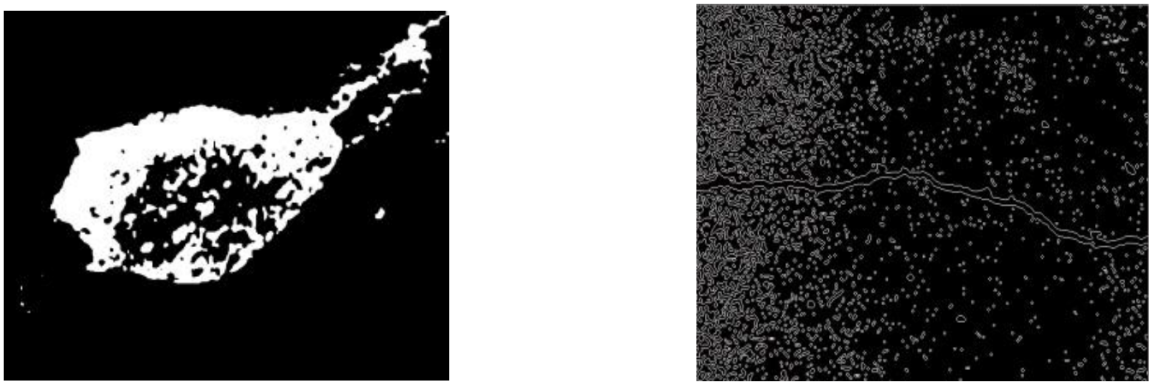
# State of the Art

In the last years, many works dealing with the detection and characterization of road pavement surface have been proposed. However as mentioned before there have been no reported research in the literature using Novelty techniques for pavement distress detection. That is why in this chapter we address some of the typical methods used regarding pavement distress detection and in the next chapter we give an overview of the available Novelty Detection methods.
Cracks are the most common distress type and they have been heavily investigated in the literature. Methods that perform crack detection, real-time crack analysis and crack classification have been proposed [Powell and Satheeshkumar, 2016]. On the other hand, researches on the detection of other surface distress such as patches and potholes are relatively less.

## 2.1 Pre-processing

There are different factors which could disturb the process of distress detection. Among those are random noise (e.g camera noise), nonuniform background illumination mostly due to the type of sensor used for image capture also nonuniform texture and shadows may affect the process considerably. So the goal of the pre-processing step is to get rid, as much as possible, of these undesirable factors while at the same time preserving the important information in the image. A wide range of pre-processing techniques were

proposed in the literature. Typically the pre-processing step starts with pixel intensity normalization in order to deal with the nonuniform background illumination. In other words we need to obtain the same average pixel intensity for each block preliminary labeled with '0' (considered as background), whereas the remaining (distress) blocks keep a lower average intensity due to the presence of relevant darker distress pixels. This step is done using histogram equalization. After that a saturation function is applied to get rid of bright pixels that could appear in images of road pavement surface due to specular reflections on some surface materials leading to similar standard deviation values in both damage and non-damage blocks. Example of a pre-propcessed images is shown in figure (2.1). This pre-processing technique is used in crack detection [Oliveira and Correia, 2013], pothole detection [Daniel and V, 2014] and in patch detection [Radopoulou and Brilakis, 2015]. For noise removal mean and median filters are two simple but efficient techniques, although median filter is more commonly used. It is used in [Koch and Brilakis, 2011] (pothole detection), [Ouyang et al., 2010] (crack detection), and in [Radopoulou and Brilakis, 2015] (patches detection). Partial differential equation (PDE) to smooth the image texture and enhance the distress is also used as a pre-processing technique for pothole detection [Lin and Liu, 2010] and for crack detection [Augereau et al., 2001].



**(a)** **(b)**

**Figure 2.1.** (a) shows a pre-processed binary image of pothole and (b) shows a pre-processed binary image of crack [Daniel and V, 2014].

## 2.2 Pothole Detection

In [Lin and Liu, 2010] a method to distinguish potholes from cracks and other pavement defects was introduced, in which the images are first segmented using partial differential equations (PDE), figure (2.2). Their algorithm makes use of the following observation: there is a granular section between the border and the sink piece in the pothole region whose texture is very different from other pavements. The method proposed aims to recognize this granular section, extract features from it such as contrast, 3-order moments, consistency and entropy and discriminate whether it is a pothole section by using Support Vector Machine (SVM).

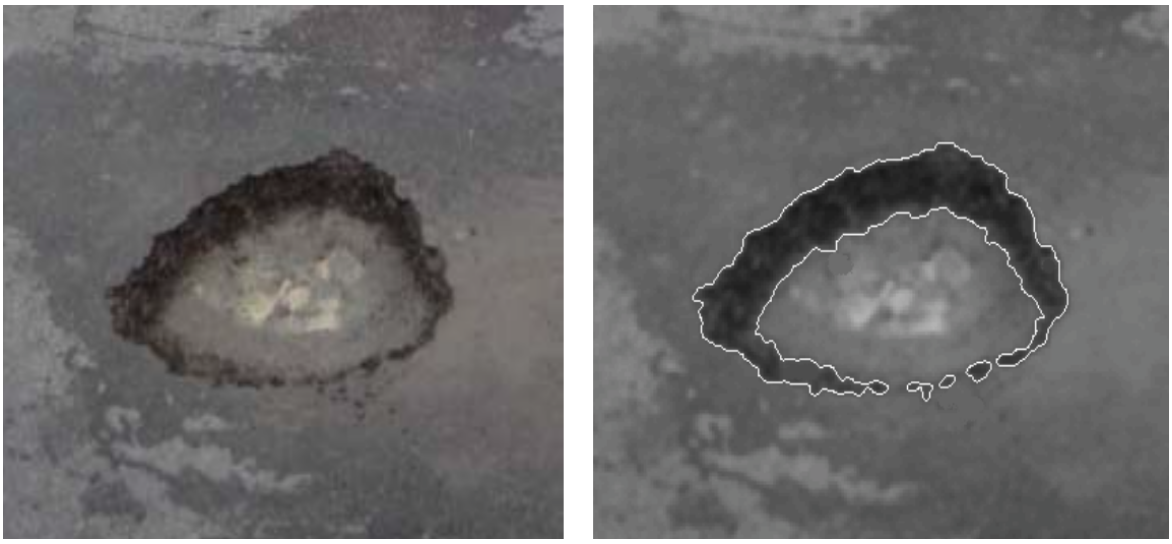


**Figure 2.2.** Image of pothole region before (left) and after (right) segmentation using PDE [Lin and Liu, 2010].

Another method proposed by Koch and Brilakis [Koch and Brilakis, 2011] starts by segmenting the images into defect and non-defect regions using histogram shape-based thresholding. After that the potential pothole shape is approximated based on the geometric properties of a defect region. The characteristics of a pothole used for the detection process are similar to the ones used in [Lin and Liu, 2010] and they are three folds: a) A pothole includes shadows that are darker than the surrounding area, b) The shape of a pothole is approximately elliptical, and c) The surface texture inside a pothole is much coarser and grainier than the surface texture of the surrounding, intact

pavement. Subsequently, the texture inside a potential defect shape was extracted and compared with the texture of the surrounding non-defect pavement to determine if the region of interest represents an actual pothole. This was done using standard deviation of gray-level intensity values as a statistical measure to describe the texture of both the inside and the outside region. From the above introduced methods we can deduce the following general flowchart for pothole detection, figure (2.3).

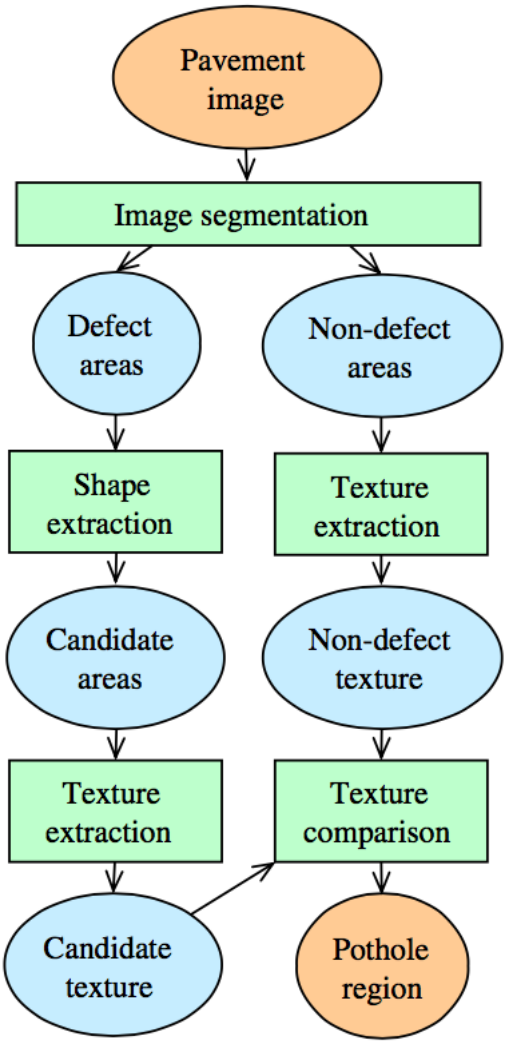



**Figure 2.3.** Pothole detection model [Koch and Brilakis, 2011].

## 2.3 Patches Detection

In [Radopoulou and Brilakis, 2015] a manual model to detect applied patches in pavement surface was introduced. As in the aforementioned pothole detection techniques this process utilizes the following main visual characteristics of an applied patch: 1) The patch consists of a closed contour, and 2) The texture of the surface of a patch is similar, if not the same, with the healthy pavement that surrounds it. After pixel intensity normalization described in the pre-processing section above, the pre-processing continues with making the image binary using histogram shape-based thresholding and finally morphological process is done to enhance the contours of the objects included in the image. To compare the textures of both a candidate patch and the healthy pavement around it standard deviation of gray-level intensity values of both regions was used. The entire process is shown in figure (2.4).

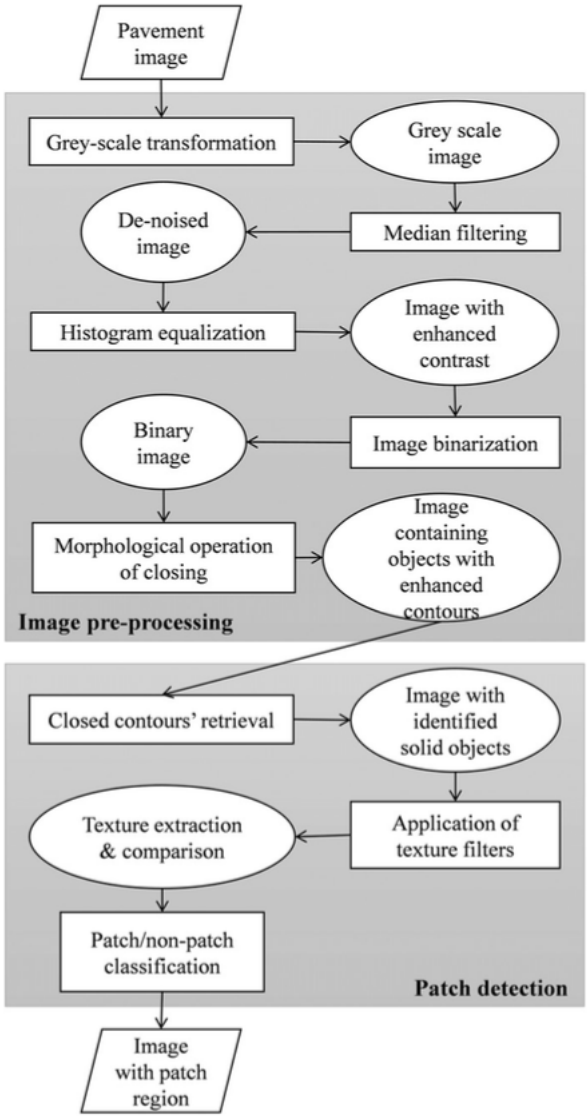



**Figure 2.4.** Patch detection model [Radopoulou and Brilakis, 2015].

## 2.4 Crack Detection

### 2.4.1 Feature extraction

This stage is considered the most critical. Depending on the features quality, i.e. the ability to distinguish distress features from non-distress features, the entire system performance could be affected. For crack detection there are tow main approaches that exist int the literature: pixel-based and block-based methods [Marques, 2012]. Pixel-based methods aim to segment the image into background (healthy pavement) and foreground (cracks) by classifying each image pixel based on its properties (e.g. intensity). Methods introduced above for pothole and patch detection fall under this category. The second approach aims to split the image into (mostly non-overlapping) blocks and to extract features from each block. Eventually a supervised learning algorithm can be trained (e.g. a neural network) to discriminate crack from non-crack blocks. As in pothole and patch detection, photometric (e.g. crack pixels are darker than the road pixels) and geometric (e.g. crack is a thin continuous object) characteristics are exploited to discriminate crack from non-crack features.

#### Pixel-based

This approach segments the image into a background (no cracks) and foreground (cracks), by classifying each pixel based on its properties. The technique reported in [Jing and Aiqin, 2010] starts with the histogram equalization (explained above) as a pre-processing step to deal with non-uniform background illumination and uses photometric properties such as the pixel grey level as crack feature. In [Nguyen et al., 2009] the above mentioned characteristics of cracks were exploited and the mean and standard deviation are used as features. After the feature extraction step, the crack detection process is carried out, see section (**??**). Other papers use frequency information from the image. Cracks existence is accompanied with a sudden transition in the image, thus associating with high frequencies. The method proposed in [Salman et al., 2013] which uses Gabor filter is an example of such approaches. A 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. In the pro-

posed method, a filter bank was generated using Gabor filter. The filter bank contains multi-orientation filter. The Gabor filter of a given orientation is then convolved with the input pre-processed image and the real component of the response is thresholded to generate a binary output image. Finally, the binary images resulting from the differently oriented filters are combined by logical OR operation to produce an output image that contains detected crack segments. See figure (2.5).

**(a)** **(b)**

**Figure 2.5.** (a) Shows the input image and (b) Shows the final output image [Salman et al., 2013].

### Block-based

This approach divides the image into (mostly non-overlapping) blocks and then extracts features from each block. Commonly used are two simple but effective local statistics: mean and standard deviation of pixel intensities of each block [Oliveira and Correia, 2013]. These two features were also used in pixel-based methods.

### 2.4.2 Crack detection

#### Pixel-based

The most used method for classification in pixel-based methods is image thresholding. With thresholding we compare the feature value of each pixel with a predefined threshold and classify accordingly which pixel corresponds to a crack and which is a background pixel [Marques, 2012]. The thresholding operation can be stated as follows:

$$L(x) = \begin{cases} 1 & \text{if } l(x) < T \\ 0 & \text{Otherwise} \end{cases} \tag{2.1}$$

$T$ is the threshold used, $l(x)$ is the feature value at position $x$ and $L(x)$ is the label assigned. An example of this is shown in figure (2.6). The method reported in [Jing and Aiqin, 2010], mentioned above, uses this thresholding technique and the threshold is set to 0.04. Morphological operators, which are capable of eliminating the most obvious errors, e.g., isolated pixels, are often used after thresholding as a post-processing to reduce the number of false positives. This was done in [Powell and Satheeshkumar, 2016] in which adaptive thresholding based on weighted means method was performed.

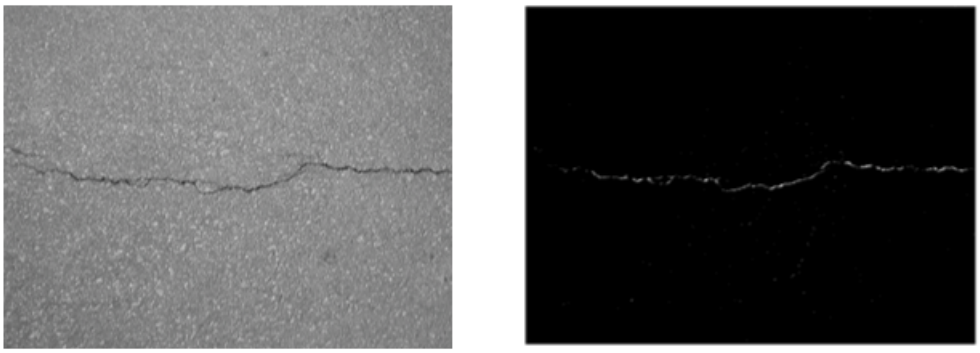


**Figure 2.6.** Thresholding example: left: original image, right: image after thresholding [Marques, 2012]

**Block-based**

Classifiers reported in the literature which falls under this category include i) neural networks [Chou et al., 1995], ii) K-nearest neighbour [Oliveira and Correia, 2009], iii) support vector machine [Li et al., 2009]. Most of them use ground truth data segmented by an expert to train a supervised recognition system which is capable of labeling the blocks in each image as crack '1' or non-crack '0'. However [Oliveira and Correia, 2013] introduced a learning from sampled paradigm with unlabeled data being used for training the CrackIT system. In the preliminary labeling process the gray-level images of size (1536x2048 pixels) were divided into 75x75 non-overlapping blocks, after that $M_m$ (mean matrices) were vertically and horizontally scanned to find cracks in each block and subsequently producing a preliminary labeling binary matrix ($plb$). For more information and deeper mathematical understanding see [Oliveira and Correia, 2013].

# Chapter 3

# Novelty detection

Unless otherwise stated all information in this chapter are taken from [Pimentel et al., 2014].

Novelty detection is the identification of new different (or novel) data or signal that a machine learning system is not aware of during training [Markou and Singh, 2003a, Markou and Singh, 2003b, Pimentel et al., 2014]. The methods of novelty detection are mostly applied to data sets where the positive (normal) examples are very well sampled whereas a sever scarcity in the negative (abnormal) examples exist for that reason the term one-class classification is often used interchangeably. The importance of novelty detection rises in modern high integrity systems in which the examples of normal behaviour of the system outnumbers the examples of abnormal behaviour [Clifton et al., 2006] and in which a number of 'abnormalities' of the system may not be known a priori making the multi-class classification techniques inappropriate [Pimentel et al., 2014]. Novelty detection introduces a solution to this problem by building a model of normality representing the positive class then testing the previously unseen data against this model. After that a novelty score is given to the test sample and is compared to a decision threshold. Finally a decision that the test data is 'abnormal' is made if this threshold is exceeded [Markou and Singh, 2003a, Pimentel et al., 2014]. In [Pimentel et al., 2014] a very comprehensive review of the different categories of novelty detection as well as the different application domains was introduced, see Figure (3.1). In the following we attempt to provide a short introduction to the five main categories.

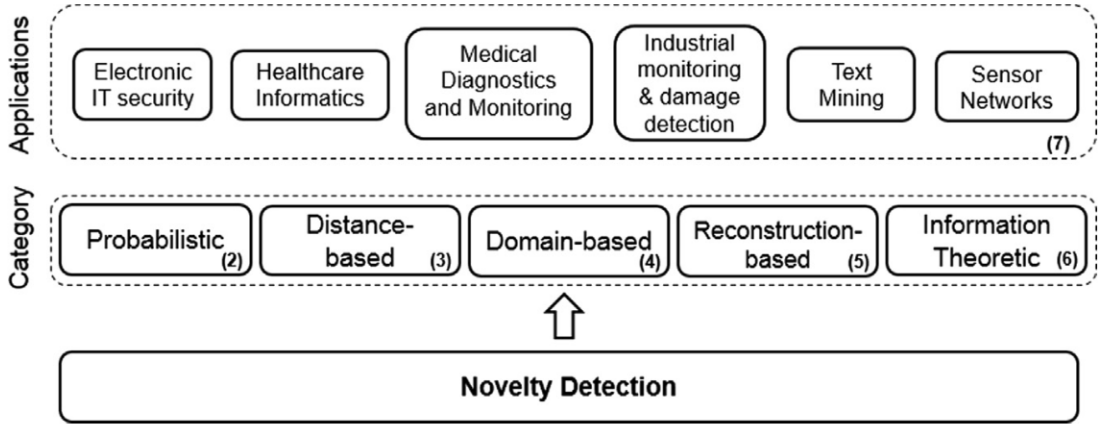



**Figure 3.1.** Categories and application domains of novelty detection [Pimentel et al., 2014]

## 3.1 Probabilistic approaches

In this approach normal data are assumed to be generated from some underlying probability density function (pdf) which can be estimated from training examples [Tarassenko et al., 2009]. The result of the estimation is seen to be representing a model of normality and can be thresholded to define the boundaries of normal data. By testing against this model we decide weather a test sample is generated from the same distribution or not hence recognised as normal or novel. We identify two different techniques in this approach parametric and non parametric.

### 3.1.1 Parametric Techniques

In these techniques assumption is made that the data comes from a known distribution and the parameters of this distribution are estimated using the training data. Most commonly used is the Gaussian distribution in which the maximum likelihood estimate **MLE** is used to estimate its parameters. If the data form is more complex then the underlying distribution can be estimated using mixture models such as Gaussian mixture models (GMM), see figure (3.2), or other mixtures. Parametric techniques impose a restrictive model on the data, as a result the chosen functional form of the

data distribution may not be a good model of the distribution that generates the data making these techniques of a little practical importance. Other methods which fall under this category are state space models which are often used for novelty detection in time-series data. Most common state-space model used for novelty detection is Hidden Markov Model (HMM).

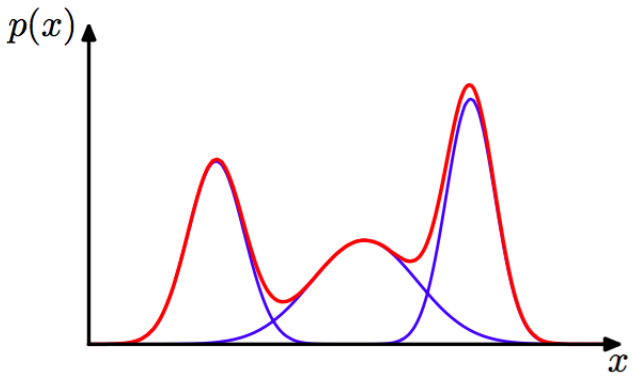



**Figure 3.2.** Example of a Gaussian mixture distribution in one dimension showing three Gaussians in blue and their sum in red [Bishop, 2006].

### 3.1.2 Non-Parametric Techniques

These techniques make very few assumptions about the form of the data distribution, thus considered to be building a more flexible model. These techniques do not assume that the structure of a model representing the data is fixed, but rather the model grows in size to fit the complexity of the data, for that reason a large number of samples is required in able to accommodate all free parameters. Nonparametric approaches include kernel density estimators, see figure (3.3), in which a kernel is placed on each data point (typically Gaussian) and then contributions of each kernel are summed. [Tarassenko et al., 2009].

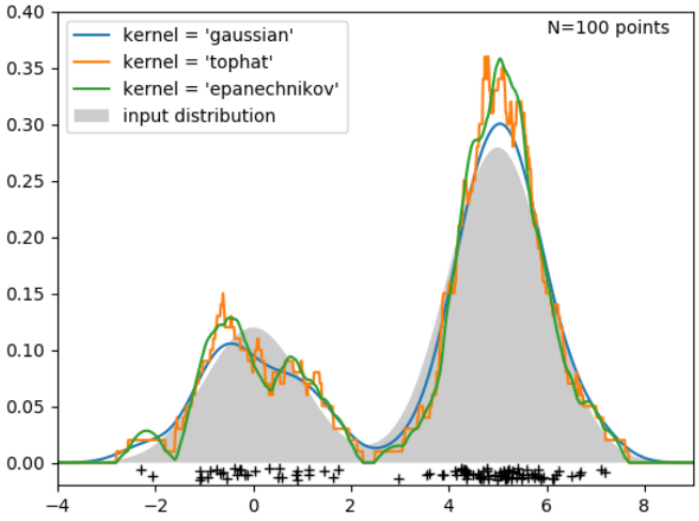



**Figure 3.3.** Example of Kernel Density Estimator with different kernels [sckikit, learn, 2011].

## 3.2 Distance-based approaches

These approaches rely on the assumption that normal data lie close to other normal data while novel data occur far from their nearest neighbours. They use well defined distance metrics to calculate the distance between two data points. This approach is further divided into two approaches i) Clustering-based: include methods such as k-means clustering where the minimum distance from a test point to its nearest prototype is compared to a predefined distance (novelty threshold) to quantify abnormality, see figure (3.4), ii) Nearest neighbour-based such as KNN explained later in details.

## 3.3 Domain-based approaches

In Domain-based approaches a decision boundary is created depending on the structure of the training data, in other words they are insensitive to the specific sampling and density of the interested class. New unseen data points are considered novel or outlier if they lie outside the novelty boundary. Support vector data description **SVDD** and One-class support vector machine, explained later in details, are examples of such

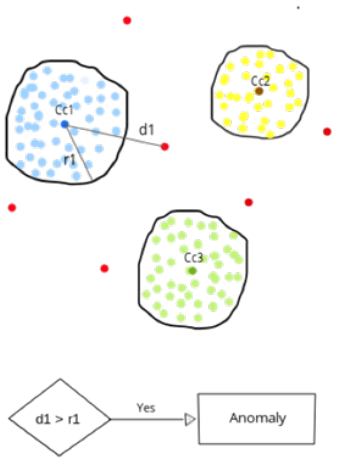



**Figure 3.4.** K-means clustering for novelty detection [Ashen, Weerathunga, 2016].

approaches.

## 3.4 Reconstruction-based approaches

Are based on training a regression model. The distance between the regression target and the actual observed value (i.e. the reconstruction error) can be related to the novelty score, which would be high when abnormal data occurs. Neural networks such as Multi-layer perceptron **MLP** and Autoassociative networks fall under this category. There is also subspace-based techniques which assume that data can be projected into a lower dimensional subspace, which makes better discrimination of normal and abnormal data.

## 3.5 Information theoretic novelty detection

Information-theoretic methods calculate the information content of a dataset with measures such as entropy, relative entropy, and Kolmogorov complexity, etc. The assumption is that novel data changes the information content of a dataset significantly. A common procedure would be to compute the metric using the entire dataset and then repeatedly eliminating the points which by elimination causes the biggest difference to

the metric value. This resulting subset of eliminated points constitute the novel data.

## 3.6 Summery and Next Step

Of the above introduced categories of novelty detection, we are interested in this work only in the fist three namely, probabilistic approaches, distance-based approaches and domain-based approaches. Information theoretic methods are not suitable to use in our case since the assumption that novelty significantly alters the information content of the otherwise 'normal' dataset, do not hold. The distress presented in the images is weakly contrasted (the road possesses a texture that hides the distress) and weakly presented in the case of cracks [Chambon and Moliard, 2011]. This makes novel data often look very similar to normal data barley effecting the data distribution. Reconstruction-based novelty detection was used for the task of distress detection in a work associated with the ASINVOS project also using the GAPs dataset, [Nitsche, 2017], and is therefor omitted in this work. Next chapter we introduce in details the methods used and how they have been implemented in the context of distress detection.

# Chapter 4

# Implemented Methods

## 4.1 Gaussian Mixture Model

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters to be estimated using training dataset. They can be used to approximate a wide variety of pdfs and are suitable for cases where single function forms (single Guassian distribution) fail. However mixture models like GMM suffer from the requirement of large numbers of training samples if the dimensionality of the data is high (curse of dimensionality). Also this method falls under the parametric techniques, in other words it makes assumptions about the underlying distribution of the data, hence imposing a restrictive model on the data. Therefore the pdf estimated from training dataset might not be a good distribution to characterize the presented data [Pimentel et al., 2014].

The multivariate unimodal normal distribution of a $D$ dimensional vector $\mathbf{x}$ is given in equation (4.1), where $\boldsymbol{\mu}$ is the mean vector and $\boldsymbol{\Sigma}$ is the covariance matrix [Bishop, 2006], see figure (4.1).

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{D}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} exp\left[ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right] \tag{4.1}$$

For a complex data distribution, the assumption of unimodality can cause a consid-

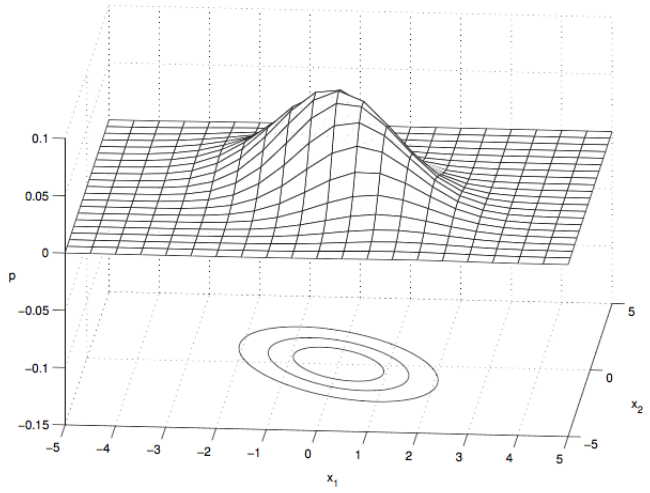



**Figure 4.1.** Example of 2-dimensional Gaussian pdf [Paalanen et al., 2006]

erable error to the estimated pdf, leading to errors in classification [Paalanen et al., 2006]. That is why we use mixture models to model multimodal random variable whose values come from different independent sources.

The probability density function of a Gaussian mixture can be defined as a weighted sum of Gaussians as expressed by equation (4.2). See figure (4.2).

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{4.2}$$

Where $C$ is the total number of components used, $\pi_c$ is the weight parameter which could be interpreted as an a priori probability that a value of the random variable is generated by the cth component, and thus, $0 \leq \pi_c \leq 1$ and $\sum_{c=1}^{C} \pi_c = 1$. The entire unknown parameters, $\boldsymbol{\theta}$ , of the Gaussian mixture probability density function is given by equation (4.3).

$$\boldsymbol{\theta} = \{\pi_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, ..., \pi_C, \boldsymbol{\mu}_C, \boldsymbol{\Sigma}_C\} \tag{4.3}$$

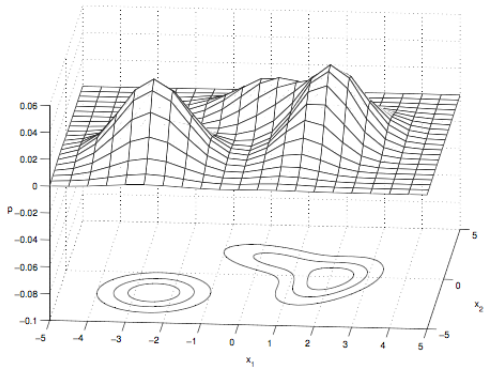



**Figure 4.2.** Example of 2-dimensional Gaussian mixture with 3 components [Paalanen et al., 2006]

### 4.1.1 Estimating Mixture Model Parameters

In the probabilistic approaches, novelty detection decides whether a test sample is normal or novel by looking if it was generated from the same underlying data generator (in this case it is assumed to be the mixture of Gaussians defined above $p(x)$) of the training data. In that sense, a test point is considered novel if $p(x_t)$ is below some predefined **novelty threshold** but more on that later. The important part of the process is estimating the parameters $\boldsymbol{\theta}$ of the learnt pdf. Bayesian methods (such as variational Bayes) and expectation-maximization **EM** are two most used methods in the literature. The principal behind variational Bayes is the same as expectation-maximization (discussed next), but variational Bayes imposes priors on model parameters, thus this method needs more hyper-parameters that might need experimental tuning. Although in this method it is possible to let the model choose a suitable number of effective components automatically, which is one advantage over EM algorithm [Tzikas et al., 2008]. In our implementation we have used EM algorithm to avoid the tuning of so many parameters accompanied with the use of variational Bayes method.

### 4.1.2 Maximum-likelihood Estimation

It seems reasonable that a good estimate of the unknown parameter $\boldsymbol{\theta}$ would be the value of $\boldsymbol{\theta}$ that maximizes the probability, that is, the likelihood of getting the data we observe. In the following we describe the process for univariate Gaussian distribution but the same can be applied to estimate the parameters of a multivariate Gaussian distribution. Using the univariate Gaussian distribution, the data is expected to be normally distributed with mean $\mu$ and variance $\sigma^2$, donated as $\mathcal{N}(\mu, \sigma^2)$ [Bishop, 2006]. For the case of single-valued variable $x$ the Gaussian distribution is given by equation (4.4).

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} exp\left[ - \frac{(x-\mu)^2}{2\sigma^2} \right] \tag{4.4}$$

Now suppose we have a dataset of $N$ observations $\mathbf{x} = (x_1, ..., x_N)^T$ where each observation is assumed to be drawn independently from a Gaussian distribution whose mean $\mu$ and variance $\sigma^2$ are unknown. To determine these parameters we start by calculating the joint probability density function which we will call the likelihood function and is given by equation (4.5). See figure (4.3).

$$\mathcal{L}(\mathbf{x}|\mu, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \sigma^2) \tag{4.5}$$

Next, we find the parameters values that maximizes the likelihood function. However, in practice it is more convenient to maximize the log of the likelihood function, equation (4.6). Because the logarithm is a monotonically increasing function of its argument, maximization of the log of a function is equivalent to maximization of the function itself [Bishop, 2006].

$$L(\mathbf{x}|\mu, \sigma^2) = \ln \mathcal{L}(\mathbf{x}|\mu, \sigma^2) = \sum_{n=1}^{N} \ln \mathcal{N}(x_n|\mu, \sigma^2) \tag{4.6}$$

Now we find the estimators $\hat{\mu}$ and $\hat{\sigma^2}$ that maximizes the log likelihood $L(\mathbf{x}|\mu, \sigma^2)$ by taking the partial derivatives of the log likelihood and setting to 0. Equations (4.7)

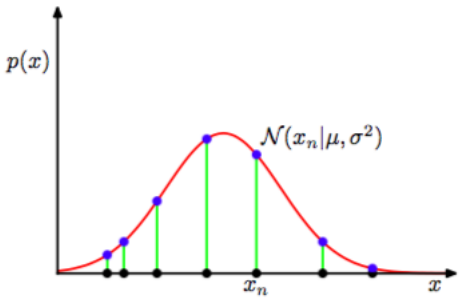



**Figure 4.3.** Illustration of the likelihood function for a Gaussian distribution, shown by the red curve. Here the black points denote a data set of values $x_n$, and the likelihood function given by equation (4.5) corresponds to the product of the blue values. Maximizing the likelihood involves adjusting the mean and variance of the Gaussian so as to maximize this product [Bishop, 2006]

and (4.8).

$$\frac{\partial}{\partial\mu}\big(L(\mathbf{x}|\mu,\sigma^2)\big) = \frac{\partial}{\partial\mu}\left( -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln(\sigma^2) - \frac{N}{2}\ln(2\pi)\right) = 0 \quad (4.7)$$

$$\frac{\partial}{\partial\sigma}\big(L(\mathbf{x}|\mu,\sigma^2)\big) = \frac{\partial}{\partial\sigma}\left( -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln(\sigma^2) - \frac{N}{2}\ln(2\pi)\right) = 0 \quad (4.8)$$

After Solving these two equations we get the estimators that maximizes the likelihood function. Equations (4.9) and (4.10).

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} x_n \quad (4.9)$$

$$\hat{\sigma^2} = \frac{1}{N}\sum_{n=1}^{N}(x_n-\hat{\mu})^2 \quad (4.10)$$

Suppose now we have a dataset of observations $(\mathbf{x_1},...,\mathbf{x_N})$, where each observation is a $D$ dimensional vector and we wish to model this data using a mixture of Gaussians.

We represent this data as a $N$x$D$ matrix $\mathbf{X}$ in which the nth row is given by $\mathbf{x_n}^T$. The unknown parameters to be found are in this case the ones introduced in equation (4.3). We determine the likelihood function as we did above. Equation (4.11).

$$\mathcal{L}(\mathbf{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x_n}|\boldsymbol{\theta}) \tag{4.11}$$

The log likelihood is then defined by:

$$L(\mathbf{X}|\boldsymbol{\theta}) = \ln \mathcal{L}(\mathbf{X}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\mathbf{x_n}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \left\{ \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{x_n}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \right\} \tag{4.12}$$

Maximizing the log likelihood function (4.12) for a Gaussian mixture model turns out to be a more complex problem than for the case of a single Gaussian. The difficulty arises from the presence of the summation over c that appears inside the logarithm in (4.12), so that the calculation of the derivatives is no longer possible. [Bishop, 2006]. One approach to solve this is by using an iterative algorithm called Expectation Maximization or **EM**.

### 4.1.3 Expectation-Maximization Algorithm

Given a Gaussian mixture function the goal is to maximize the log likelihood function with respect to the parameters $\boldsymbol{\theta}$. We start by some initial random estimate of $\boldsymbol{\theta}$ and then proceed to iteratively update $\boldsymbol{\theta}$ until convergence is detected. Each iteration consists of an E-step and an M-step. The procedure is summarized in the following steps:

1. initialize the means $\boldsymbol{\mu}_c$, the covariances $\boldsymbol{\Sigma}_c$ and the mixing coefficients $\pi_c$, and evaluate an initial value of the log likelihood function.Equation (4.12).

2. **E step**. Evaluate the responsibilities $\gamma_{nc}$, that is, the membership weights for all data points $\mathbf{x_n}^T$ for $0 \leq n \leq N$ and for all components $1 \leq c \leq C$ using the following equation (4.13):

$$\gamma_{nc} = \frac{\pi_c \mathcal{N}(\mathbf{x_n}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}{\sum_{j=1}^{C} \pi_j \mathcal{N}(\mathbf{x_n}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{4.13}$$

3. **M step**. Re-estimate the parameters using the current responsibilities. By setting the derivative of the log likelihood function, equation(4.12), with respect to $\boldsymbol{\mu}_c$, $\boldsymbol{\Sigma}_c$ and the mixing coefficients $\pi_c$ to 0 we get the following, where $N_c$ can be interpreted as the effective number of points assigned to component $c$.

$$\boldsymbol{\mu}_c^{new} = \frac{1}{N_c}\sum_{n=1}^{N}\gamma_{nc}\mathbf{x_n} \tag{4.14}$$

$$\boldsymbol{\Sigma}_c^{new} = \frac{1}{N_c}\sum_{n=1}^{N}\gamma_{nc}(\mathbf{x_n} - \boldsymbol{\mu}_c^{new})(\mathbf{x_n} - \boldsymbol{\mu}_c^{new})^T \tag{4.15}$$

$$\pi_c^{new} = \frac{N_c}{N} \text{ where: } N_c = \sum_{n=1}^{N}\gamma_{nc} \tag{4.16}$$

4. Re-evaluate the log likelihood function, equation (4.12), and check for the convergence of either the parameters or the log likelihood. The convergence occurs when the change in the log likelihood function or the in the values of the parameters fall below some threshold. If not converged return to step 2.

The convergence of **EM** is certain, because the log likelihood functions is guaranteed to increase in each iteration, see figure (4.4).

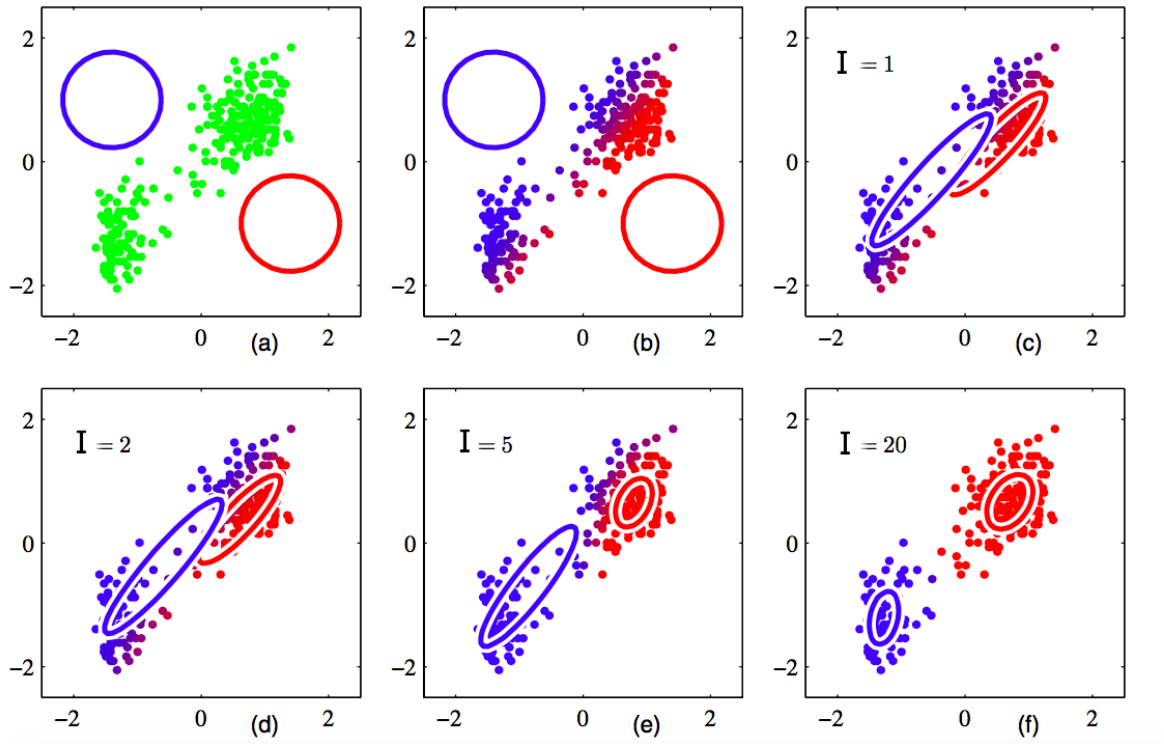



**Figure 4.4.** Illustration of the EM algorithm using two Gaussian components. The figure shows the result of the iterative procedure until convergence, starting from (a) and finishing with (f). In (a) the data points are shown in green along with standard-deviation contours for two Gaussian components shown as red and blue circles. After that green points are assigned blue, red and purple colors representing the probabilities of being generated by the red or blue Gaussian component, with purple color meaning that points belong to the two components with low probability. [Bishop, 2006].

In figure (4.4) plot (a), the data points are shown in green and the standard-deviation contours for two Gaussian components are shown as blue and red circles. Plot (b) shows the result of the initial E step, in which each data point is depicted using a proportion of blue ink equal to the probability that the blue component was responsible of generating that point (responsibility), the same applies for the points depicted using a proportion of red ink. Thus, points appearing in purple belong to the two components with low probability. Plot (c) shows the situation after the first M step, we can see that the mean of the blue component has moved to the mean of the dataset, weighted by the

probabilities of each data point belonging to the blue component (i.e moved to the center of the set containing blue points), equation (4.14). Also the covariance of the blue component is now equal to the covariance of the blue ink, equation (4.15). Plots (d), (e) and (f) show the results after 2, 5 and 20 cycles of the EM algorithm.

### 4.1.4 GMM in Novelty Detection

In novelty detection, information from only one class is used, we can call this class also the target class. The model of the target class is estimated using GMM normally using fewer kernels than the number of patterns in the dataset. The most essential part in novelty detection is setting a suitable novelty threshold. Under probabilistic approaches such as GMM novelty scores of new unseen data is defined using the pdf in equation (4.2). Typical methods of setting novelty threshold are heuristic methods such as the one applied here. In this method we change the threshold repeatedly to be equal to the novelty scores calculated for each new data point and choose of those the threshold which reflects the best performance of the classifier. Another method for choosing the best novelty threshold is Extreme Value Statistics, [Pimentel et al., 2014]. As in [Zorriassatine et al., 2005], threshold was first set to be the minimum log likelihood of the training data and validation dataset was used to adjust this threshold using the heuristic process. If, subsequently, a test vector $\mathbf{x}$ belongs to a region of input space for which $p(\mathbf{x}|\boldsymbol{\theta})$ is below this threshold, then that vector is deemed to be novel or 'abnormal'. Algorithm (1) summarizes the whole implemented method of GMM.

### 4.1.5 Limitations of Gaussian Mixture Models

1. More kernels are better to model a complex underlying distribution of the data, but more kernels means more parameters to be set and subsequently the need for a larger data set (Curse of dimensionality).

2. Parametric methods such as GMM impose a restrictive model on the data, as mentioned in section 4.1 the data is assumed to be generated from a mixture

of Gaussians. However, in many real-life scenarios, no a priori knowledge of the data distributions is available, and so parametric approaches may be problematic if the data do not follow the assumed distribution.

**Algorithm 1** `GMM method`

1: Fitting a mixture of Gaussian kernels to the training dataset (this includes learning the model parameters using EM algorithm see section 4.1.3). In this step we need to provide the number of components to be used.

2: Calculate the log likelihood of each point in the training dataset and set the minimum as the novelty threshold.

3: For each new unseen data point, the log likelihood of belonging to the model is calculated and compared with the novelty threshold. If it exceeds the novelty threshold then the data point is normal otherwise it is novel.

4: The validation data set was used to tune the number of components used to a value which gives the best detection results of the distress presented in the dataset.

## 4.2 One-class SVM

Unless otherwise stated all information and equations in this chapter are taken from [Hofmann, 2006]. One-class SVM belongs to the domain-based methods of novelty detection. In these methods we are interested in creating a boundary around the target class, and as in the original two-class SVM, one-class SVM determines the location of the boundary only using the data which lie closest to it (support vectors) and all other data from the training set are not considered. By doing this, we discard the distribution of the presented data, which is seen as not solving a more general problem than is necessarily [Pimentel et al., 2014]. Since One-class SVM is a special variant of the original SVM, we will first start with an overview of the original SVM which is ideally suited for binary pattern classification of the data that are linearly separable, and then take a look at the adjusted SVM which is suited for classification of non-linearly separable data.

### 4.2.1 Basic Concepts of Support Vector Machine

considering a dataset $X = \{(\mathbf{x_1}, y_1), ..., (\mathbf{x_n}, y_n)\}$; where $\mathbf{x_i} \in \mathbb{R}^N$ and $y_i \in \{-1, +1\}$ are the labels of the two classes. The traditional two-class support vector machine uses a hyperplane that maximizes the separating margin between two classes. It implements a maximum-margin training algorithm in order to find a decision function so that for new unseen test sample $\mathbf{x}$ the corresponding label would be assigned.
If the data is linearly separable, then there exists a hyperplane of the form:

$$\omega^T \mathbf{x} + b = 0 \tag{4.17}$$

Separating the positive from the negative training samples such that:

$$\begin{aligned} &\omega^T \mathbf{x_i} + b \geq 0, \text{ for } y_i = +1, \\ &\omega^T \mathbf{x_i} + b < 0, \text{ for } y_i = -1, \end{aligned} \tag{4.18}$$

where $\omega$ is the normal to the hyperplane and $b$ is the perpendicular distance of the hyperplane to the origin, see figure (4.5). The decision function $f(\mathbf{x}) = \omega^T \mathbf{x_i} + b$ can be interpreted as the distance between a data point and the hyperplane.

Nevertheless, there exist many separating hyperplanes to separate the two classes as illustrated in figure (4.6). To find the hyperplane that separate the two classes with maximum margin we apply the maximum-margin algorithm which solves the following optimization problem:

$$\begin{aligned} &\max \frac{2}{\|\omega\|} \\ &\text{subject to} \quad \omega^T \mathbf{x_i} + b \geq +1 \quad \text{for } y_i = +1, \\ &\qquad\qquad \omega^T \mathbf{x_i} + b \leq -1 \quad \text{for } y_i = -1, \end{aligned} \tag{4.19}$$

The constraints state that each data point must lie on the correct side of the margin. Solving this optimization problem using Lagrange multipliers via numerical optimization method would lead to finding the optimal hyperplane. The optimal decision function is given in equation (4.20), where $\alpha_i$ are the optimal Lagrange multipliers found.

$$f(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i}^T \mathbf{x} + b) \tag{4.20}$$

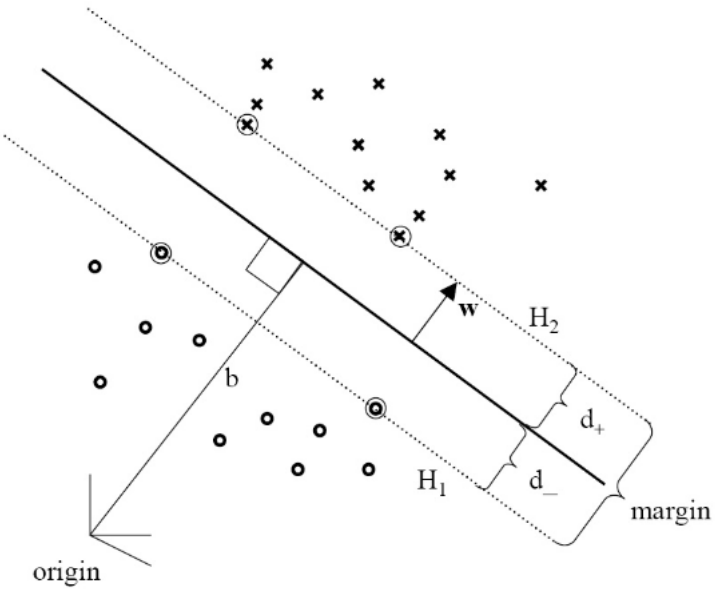



**Figure 4.5.** Optimal separating hyperplane with maximum margin [Hofmann, 2006].

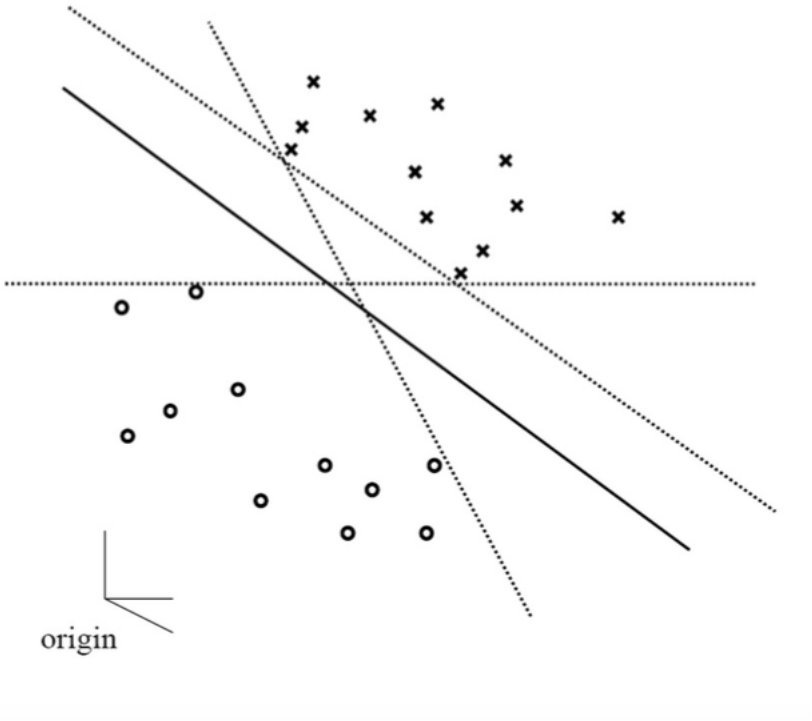



**Figure 4.6.** Suboptimal (dashed) and optimal (bold) separating hyperplanes [Hofmann, 2006].

### 4.2.2 Dealing with Non-linear separable Data

One nice feature of SVM is that it could be adapted to create a non-linear decision boundary in case the data is linearly non-separable. This is done by projecting the data through a non-linear function $\Phi$ into a higher dimensional linear separable feature space. Figure (4.7) shows a linearly nonseparable dataset that is easily separable by a hyperplane when lifted into a higher dimension. When that hyperplane is projected back to the original input space, it would have the form of a non-linear curve, figure (4.8).

We solve the following minimization problem (with quadratic programming) using

Lagrange multipliers :

$$\begin{aligned} \underset{w,b,\xi_i}{\text{minimize}} \quad & \frac{\|\omega\|^2}{2} + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(\omega^T\Phi(\mathbf{x_i}) + b) \geq 1 - \xi_i \quad \text{for all } i = 1, .., n. \\ & \xi_i \geq 0 \quad \text{for all } i = 1, .., n. \end{aligned} \tag{4.21}$$

[Vlasveld, 2013]. Where $\xi_i$ are slack variables introduced to allow some data points to lie within the margin in order to prevent SVM from over-fitting with noisy data, and the constant $C > 0$ determines the trade-off between maximizing the margin and the number of training data points that are allowed to fall within the margin (training errors). When this minimization problem is solved we get the following decision function, where $\alpha_i$ are the optimal Lagrange multipliers found.

$$f(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{n}\alpha_i y_i \Phi(\mathbf{x_i})^T\Phi(\mathbf{x}) + b) \tag{4.22}$$

We summarize the process of dealing with nonseparable data with the following algorithm:

**Algorithm 2** `Non-linear SVM`

1: Transform the training set $X$ to $X'$ with $\Phi$
2: Train a linear SVM on $X'$ to get the classifier $f$
3: Transform new test sample $\mathbf{x}$ to $\mathbf{x}' = \Phi(\mathbf{x})$
4: Determine the label for the test sample $\mathbf{x}'$ using the classifier $f$

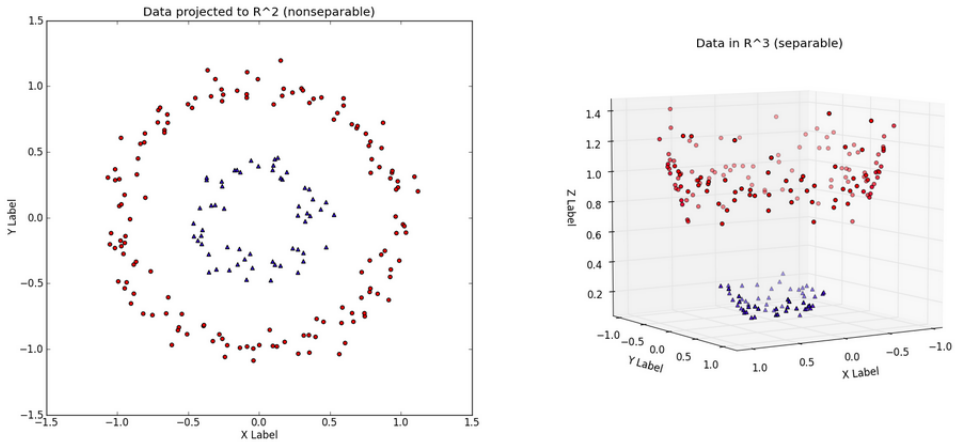



**Figure 4.7.** (Left) A dataset in $\mathbb{R}^2$, not linearly separable. (Right) The same dataset transformed to a higher space [Vlasveld, 2013].

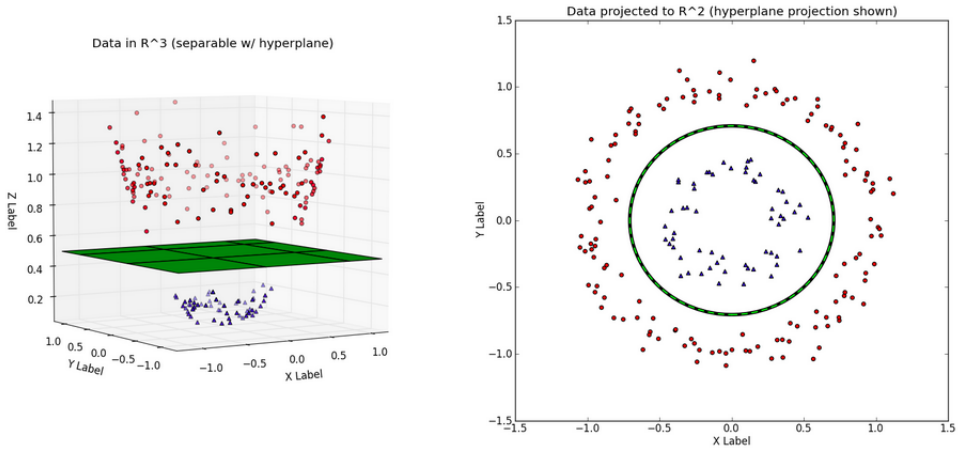



**Figure 4.8.** The created decision boundary after projecting back to the original space [Vlasveld, 2013].

### 4.2.3 The kernel Trick

It turns out that we can actually get to the same results without the need to transform the data into a higher dimensional space at training or testing time. We note that equation (4.22) depend on the mapped training data only through dot products in some higher dimensional feature space, let this one be $\mathbb{R}^M$. Here the Kernel trick comes in handy. It turns out that there are functions called **Kernel functions** $k : \mathbb{R}^N$x

$\mathbb{R}^N \to \mathbb{R}$, which given two vectors $v$ and $w$ in $\mathbb{R}^N$ implicitly computes the dot product between those vectors in a higher dimension $\mathbb{R}^M$ without explicitly transforming $v$ and $w$ to $\mathbb{R}^M$. In other words the explicit coordinates in $\mathbb{R}^M$ and even the mapping function $\Phi$ become unnecessary when we define a function $k(v, w) = \Phi(\mathbf{v})^T\Phi(\mathbf{w})$. Thus the decision function becomes:

$$f(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x_i}, \mathbf{x}) + b) \tag{4.23}$$

By using the Kernel trick, the feature space $\mathbb{R}^M$ can be of unlimited dimension and thus the hyperplane separating the data can be very complex. Popular choices of the Kernel function include: **polynomial, radial basis function,** and **sigmoid** kernels.

### 4.2.4 Schölkopf Methodology

In [Schölkopf et al., 2000], an adaptation on the classical SVM was introduced to the one-class classification problem. They framed the problem in the following way: "suppose there is a dataset drawn from an underlying probability distribution $P$ and we want to estimate a 'simple' subset $S$ of input space such that the probability that a test point drawn from $P$ lies outside of $S$ equals some a priori specified value $\nu$ between 0 and 1". To approach this problem they estimated a function $f$ which captures regions in the input space where the probability density of the data lives, in other words the function is positive on $S$ and negative on the complement, equation (4.23) [Manevitz and Yousef, 2001].

$$f(\mathbf{x}) = \begin{cases} +1 & \text{if } \mathbf{x} \in S \\ -1 & \text{if } \mathbf{x} \in \bar{S} \end{cases} \tag{4.24}$$

In our case, the subset $S$ resembles the training data containing the 'normal' class (intact data). The following algorithm should create a boundary around the normal class and estimate a decision function $f$ that is positive if a test data point is normal and negative if it is novel. But first let us introduce terminology and notation conventions.

Let $\mathbf{x_1}, ..., \mathbf{x_l} \in \mathcal{X}$ be the training data where $l \in N$ is the number of observations, and let $\Phi : \mathcal{X} \to F$ be a non-linear function which maps the input space into a dot product space $F$ such that the dot product in the image of $\Phi$ can be alternatively computed using a suitable kernel function: $k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x})^T \Phi(\mathbf{y})$ [Schölkopf et al., 2000]. In our implementation we used Gaussian kernel or **RBF** because it allows any datapoint to be separated from the origin in $F$. It is given by $k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$, with $\gamma > 0$ as free parameter. The effect of this parameter is discussed later in this chapter.

**Algorithm 3** `Schölkopf One-class SVM`

1: Map the training data into $F$ corresponding to the kernel
2: Treat the origin as the only member of the second class, figure (4.9)
3: Separate the training data from the origin with maximum margin by solving the minimization problem, equation (4.25)
4: Obtain the optimal classifier $f$, equation (4.26)
5: For a new point $\mathbf{x}$ the value of $f(\mathbf{x})$ is determined by evaluating which side of the hyperplane it falls on in $F$

$$\begin{aligned} \underset{w \in F, \rho \in \mathbb{R}, \xi_i \in \mathbb{R}^l}{\text{minimize}} \quad & \frac{\|\omega\|^2}{2} + \frac{1}{\nu l} \sum_{i=1}^{l} \xi_i - \rho \\ \text{subject to} \quad & (\omega . \Phi(\mathbf{x_i})) \geq \rho - \xi_i \qquad \text{for all } i = 1, .., l. \\ & \xi_i \geq 0 \qquad \text{for all } i = 1, .., l. \end{aligned} \tag{4.25}$$

In the previous formulation the parameter $C$ determined the smoothness. In this formula it is the parameter $\nu$. $\nu$ does two things [Schölkopf et al., 2000]:

1. It sets an upper bound on the fraction of the training data allowed to fall outside of the description of the 'normal' class.

2. It sets a lower bound on the fraction of support vectors used.

After solving this minimization problem, again by using Lagrange techniques, a hyperplane characterized by the normal $\omega$ and the offset $\rho$ is created and the the decision

function $f$ is derived:

$$f(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{l} \alpha_i k(\mathbf{x_i}, \mathbf{x}) - \rho) \tag{4.26}$$

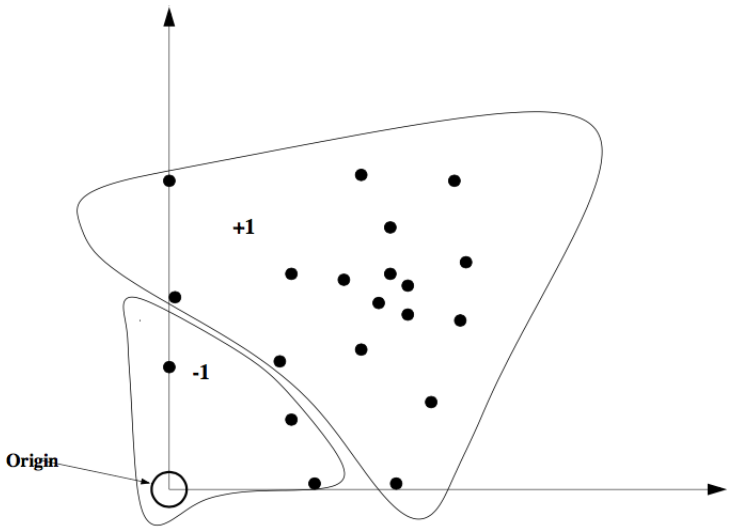



**Figure 4.9.** One-class classification using SVM, the origin is the only original member of the second class [Manevitz and Yousef, 2001].

### 4.2.5 One-class SVM in Novelty Detection

In our implementation, we have used the Schölkopf methodology and considered the origin the only member of the negative class. The novelty threshold occurs at $f(\mathbf{x}) = 0$ [Clifton et al., 2011]. Novelty score for a previously unseen data point is in this case the distance to the decision boundary (novelty threshold). It is positive if the data point is normal and negative if it is novel. Because of the importance of the parameter $\nu$ the one-class SVM method is often referred to as $\nu$-SVM. For the Gaussian kernel it is important to choose an appropriate value of parameter $\gamma$. Small values of gamma result in smoother decision boundary which tend to exhibit lower variance (better generalization ability), but increased bias. Conversely, larger values of gamma provide decreased bias (closer fit to the 'normal' data space), but at the expense of high variance (less able to generalize to previously unseen data) [Clifton et al., 2011].

### 4.2.6 Limitation of One-class SVM

Domain-based approaches determine the location of the novelty boundary using only the data which lies nearest to it, so the distribution of the data in the training set is neglected which is seen as not solving a more general problem [Pimentel et al., 2014]. The one-class SVM however suffers from the complexity associated with the computation of the kernel functions. There is also problem of setting a suitable parameter $\nu$ which controls the trade off between maximizing the margin and the percentage of training errors. By focusing on the decision boundary, this method is often influenced by outliers in the training set.

## 4.3 K-Nearest Neighbours

K-Nearest Neighbours method falls under distance-based approaches of novelty detection, see section (3.2). K-NN approach is based on the assumption that normal data points have close neighbours in the 'normal' training set, however novel points lie far from their neighbours [Pimentel et al., 2014]. Similarity measure (distance) between two data points can be calculated using well-defined distance metrics, Euclidean distance is a popular measure which was implemented in this work and delivered better results than Mahalanobis distance.

### 4.3.1 Problem Formulation

So our goal is to find for a given new unseen data point whether or not it shows a noticeable deviation from its neighbouring points figure (4.10).

### 4.3.2 K-NN in Novelty Detection

To achieve our goal, we use a distance threshold $T$ (novelty threshold) and assign each data point, whose distance to its kth nearest point in the training data set exceeds this threshold, the label '1' as in being a novel data point otherwise it is normal and we assign it the label '0' [Upadhyaya and Singh, 2012]. The following algorithm summarizes the one in this work implemented K-NN method.

To obtain the best threshold at which the classifier gives the best performance, a heuristic method implemented such as the one explained in the GMM method, see section 4.1.4.

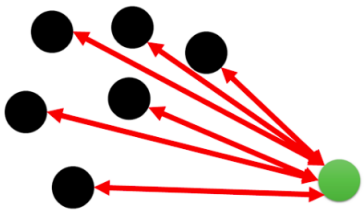


**Figure 4.10.** Distances from a data point to its k-nearest data points.

**Algorithm 4** `K-NN method`

1: For each point in the training data set, calculate the distance to its kth-nearest data point.
2: Set the the mean of these distances as the novelty threshold.
3: For each new data point (validation or test point), calculate the distance to its kth-nearest point in the training data set. These are referred to as novelty scores.
4: If, for each new data point, this distance exceeds the novelty threshold then this data point is deemed to be novel, otherwise it is normal.
5: Validation data set was used in order to set the number $K$ in such a way that gives the best performance of the classifier.

Methods using K-NN can be divided into two groups: distance-based, like the one applied in this work, explained above, and density-based. The latter method estimates the density of the neighborhood of every data instance. An instance lying in a low density neighborhood is considered to be novel while those lying in a dense neighborhood are deemed normal [Upadhyaya and Singh, 2012]. A density-based K-NN method [Cabral et al., 2007] was also implemented but showed inefficiency in execution time and did not add any improvement to the performance that's why the first method was favored. The results obtained using this method is represented briefly in the appendix section (A.4).

### 4.3.3 Limitation of K-NN

Distance based approaches of novelty detection do not require a priori knowledge of the data distribution. However, they rely on distance metrics to calculate similarity (distance) between two data points. On one hand this causes the results of the classifier to depend significantly on the metric used, and on the other hand in high-dimensional data sets this process becomes computationally expensive and as a result this technique lacks scalability. Another drawback is speed. Since space pruning must be done to find the nearest neighbours, the process becomes slower in high dimensions.

# Chapter 5

# Feature Extraction Methods

## 5.1 Gray Level Co-occurrence Matrix

The GLCM texture extraction methods provides information about shape, i.e., the spatial relationships of pixels in an image. The GLCM is defined as the joint probability of occurrence of two grey level values at a given offset both in terms of distance and orientation. In other words the GLCM is a matrix which calculates how often a pixel with gray-level value $i$ occurs at a distance $d$ and at an angle theta (spatial relationship) from grey-level pixel $j$ [Hall-Beyer, 2000]. The two pixels are called the **reference** and the **neighbour** pixel. In figure (5.1), the neighbour pixel (shown in blue) is chosen to be the one to the east (right) of each reference pixel (shown in red). This can also be expressed as a (1,0) relation: 1 pixel in the x direction, 0 pixels in the y direction.

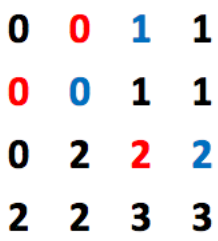



**Figure 5.1.** Example of (1,0) spatial relationship in an original grey-level image [Hall-Beyer, 2000].

We note that a different co-occurrence matrix exists for each spatial relationship. For eight bit data (256 possible gray-level values) and a certain spatial relationship we get a 256x256 square matrix. To avoid this big amount of calculations most of the programs uses scaling to reduce the number of intensity values in grayscale image from 256 to eight, see example of calculating a GLCM for scaled image in figure (5.2), where each cell in the constructed matrix contains how often the corresponding pixels $i$ and $j$ occur together with respect to the specified offset. The gray-level co-occurrence matrix can reveal certain properties about the spatial distribution of the gray levels in the texture image. For example, if most of the entries in the GLCM are concentrated along the diagonal, the texture is coarse with respect to the specified offset.

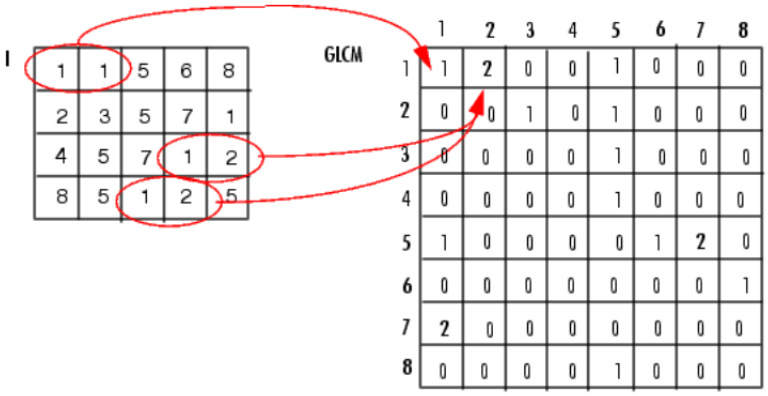



**Figure 5.2.** Example of calculating the GLCM for an image with 8 gray-level values [Matlab, ].

A number of texture features may be extracted from the GLCM, among those are: Entropy, correlation, contrast, etc. (see [Haralick et al., 1973]). In figure (5.3) an illustration of setting the offset (distance and orientation) is presented.

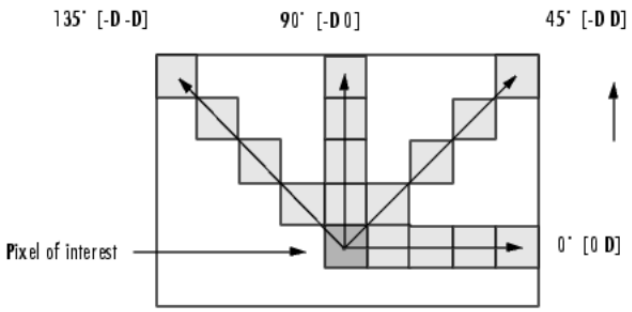



**Figure 5.3.** Clarification of offset calculation [Matlab, ].

## 5.2 Local Binary Pattern

Local Binary Pattern (LBP) is a simple yet powerful method for image feature extraction in pattern recognition and image processing. In LBP, the feature of each pixel mainly depends on its neighboring pixels. As a result, the LBP may emphasize on local information too much [Cheung and Deng, 2014]. So unlike GLCM, LBP computes a local representation of texture. The LBP implemented in this work is the one proposed by Ojala [Ojala et al., 2002], It is a gray-scale texture operator that calculates the local spatial structure of the image texture [Cheung and Deng, 2014]. Given a pixel $g_c$, a LBP is computed by comparing its value with those of it's neighbouring pixels $g_i (i = 0..P-1)$. Equation (5.1), see figure (5.4).

$$LPB_{P,R} = \sum_{i=0}^{P-1} s(g_i - g_c).2^i \tag{5.1}$$

where $P$ is the number of neighbors, $R$ is the radius of the neighborhood and $s$ is defined as following. Equation (5.2).

$$s(x) = \begin{cases} 1 & \text{if } \geq 0 \\ 0 & \text{if } < 0 \end{cases} \tag{5.2}$$

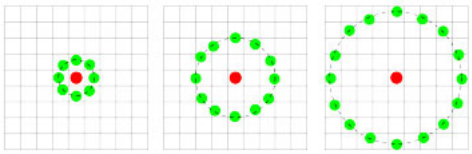


**Figure 5.4.** Three neighborhood examples with varying P and R used to construct Local Binary Patterns.[Adrian, 2015].

To understand how LBP works. let us take a look at the original LBP descriptor which operates on a fixed 3 x 3 neighborhood of pixels just like in figure (5.5).

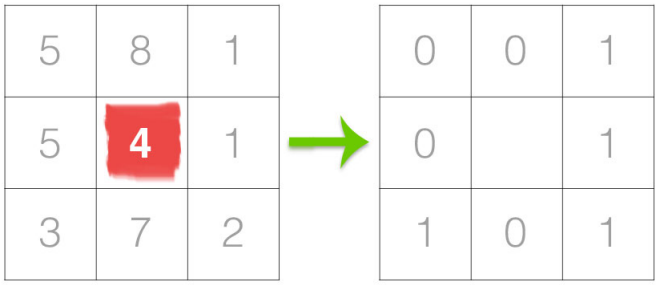



**Figure 5.5.** 8 pixel neighborhood surrounding a center pixel is taken and thresholded to construct a set of 8 binary digits [Adrian, 2015].

Starting from any neighboring pixel and working our way clockwise or counter-clockwise, if the intensity of the center pixel is greater-than-or-equal to its neighbor, then we set the value to 1; otherwise, we set it to 0. The results of this binary test are stored in an 8-bit array, which we then convert to decimal, figure (5.6). This decimal value is stored in the output LBP 2D array as shown in figure (5.7).

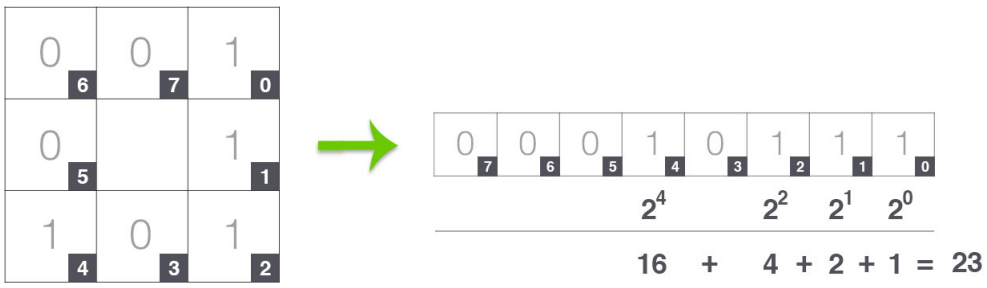



**Figure 5.6.** Converting the 8-bit binary neighborhood of the center pixel into a decimal representation [Adrian, 2015].

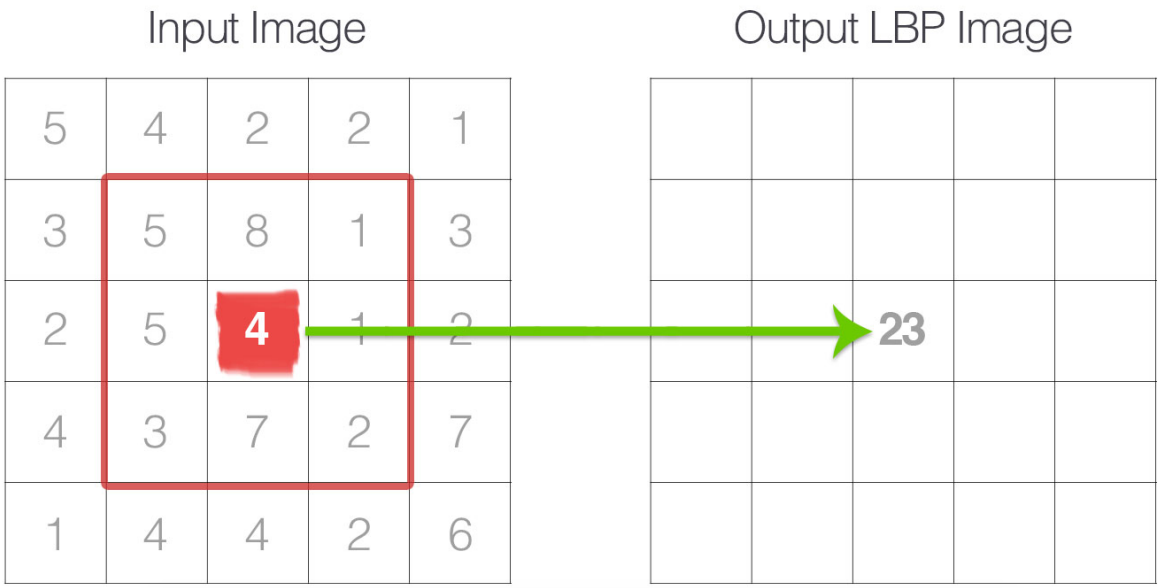



**Figure 5.7.** The calculated LBP value is stored in an output array with the same width and height as the original image [Adrian, 2015].

With 8 surrounding pixels, we have a total of $2^8 = 256$ possible combinations of LBP codes. Hence the histogram which constitute our final feature vector will have 256 bins [Adrian, 2015]. To reduce the size of this feature vector the concept of LBP uniformity was introduced. A LBP is considered to be uniform if it has at most two 0-1 or 1-0 transitions. For example, the pattern (00001000) is uniform while the pattern (01010010) is not. The number of uniform prototypes in a Local Binary Pattern is dependent on the number of points $P$. If $P$ increases so will the dimensionality of the feature vector. As a rule of thump if $P$ is the number of neighboring points specified then the number of uniform patterns will be $P + 1$, for more details please refer to [Ojala et al., 2002]. The final dimensionality of the histogram is thus $P + 2$, where the added entry refer to all patterns that are not uniform.

## 5.3 Gabor Filter

The Gabor filter is proven to be a highly potential technique for multidirectional crack detection [Salman et al., 2013]. Gabor filters are well recognized as a joint

spatial/spatial-frequency representation for analyzing images containing specific frequency and orientation characteristics. Frequency and orientation representations of Gabor filters are similar to those of the human visual system, that is why they are particularly appropriate for texture representation and discrimination. A 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal wave [Salman et al., 2013], as in equation (5.3).

$$g(x, y) = s(x, y).w(x, y) \tag{5.3}$$

where $w(x, y)$ is s a complex sinusoid, known as the **carrier**, and $s(x, y)$ is a 2D Gaussian-shaped function known as the **envelope** given in equations (5.4) and (5.5) respectively.

$$w(x, y) = \exp(j2\pi\omega_0 x' + \psi)) \tag{5.4}$$

$$s(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\Big( -\frac{1}{2}\Big[(\frac{x'}{\sigma_{x'}})^2 + (\frac{y'}{\sigma_{y'}})^2\Big]\Big) \tag{5.5}$$

Where $x' = x\cos\theta + y\sin\theta$ and $y' = -x\sin\theta + y\cos\theta$, $\omega_0$ is the magnitude of the spatial frequency and $\theta$ is the angle between the direction of the sinusoidal wave and the x-axis (orientation). $\sigma_x$ and $\sigma_y$ are the standard deviations of the Gaussian envelope in the direction of the wave and orthogonal to it.

High spatial frequencies represent abrupt spatial changes in the image, such as edges, and generally correspond to featural information and fine detail. Since text is rich in high frequency components, Gabor filter have been used for handwritten digits and alphabet recognition [Salman et al., 2013]. Figure (5.8) shows the result of applying Gabor filter to a Chinese OCR along with the image resulting from the composition of these transformed images.

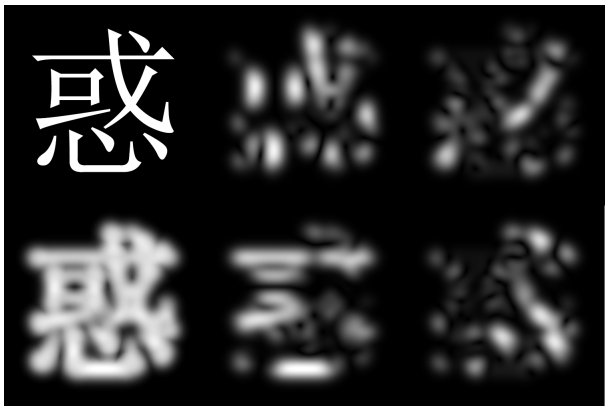


**Figure 5.8.** Four orientations are shown on the right 0°, 45°, 90°and 135°. The original character picture and the superposition of all four orientations are shown on the left [Wikipedia, the free encyclopedia, 2017].

So loosely speaking when we add up the results of applying Gabor filer with different orientation, frequencies and standard deviation (multiple kernels) we will get the original image.

Figure (5.10) shows an example of applying Gabor filter at 0°and 90°orientations on an image, figure (5.9), containing distress of type crack. This is taken from the work of [Salman et al., 2013] were Gabor filter was used as feature extraction method to detect pavement cracks.

**Figure 5.9.** Original image of crack on asphalt [Salman et al., 2013].

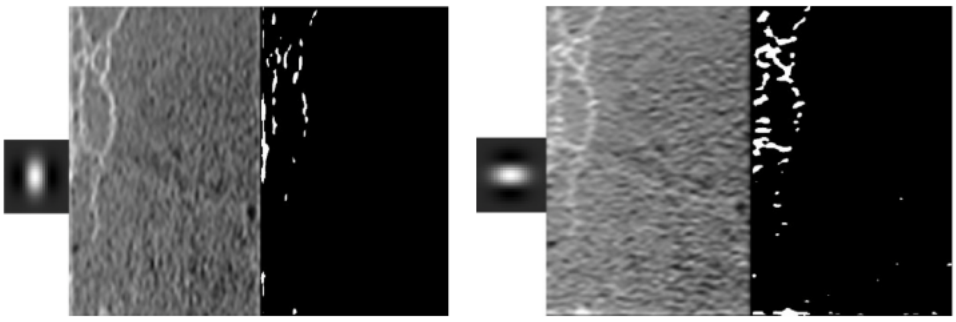


**Figure 5.10.** result of Gabor filter at 0°(left) and 90°(right) orientation. In both images the sub-image on the right is the real part of the filter response and the left one is the thresholded version [Salman et al., 2013].

# Chapter 6

# Experiments and Comparisons

Implementation of the three chosen Novelty Detection methods along with the used feature extraction methods was done using the python library Scikit Learn [Pedregosa et al., 2011].

## 6.1 Dataset and Pre-processing

As stated in the introduction section (1.3), we have used 64x64 pixels gray valued images of the GAPs dataset partitioned into Training- Validation- and Test-datasets. For the training we use 30k images from the normal class (intact). Validation and test-datasets are both of size 10k each containing 6k images of normal (intact) samples and 4k images of abnormal (distress) samples. For the pre-processing median filter was applied which is a simple but efficient technique for noise removal.

## 6.2 Performance Measures

For the evaluation of the classifiers the following metrics were favored: the F1 score which is derived from the PR curve, and the Area Under Curve (AUC) which is derived from the ROC curve. The F1 score can be interpreted as the harmonic mean of the precision and recall, equation (6.1), where F1 score reaches its best value at 1 and worst score at 0. By implementing the F1 measure the average type was set to **macro**

instead of **binary**. Binary averaged F1 score focuses on the positive class (distress), its magnitude is mostly determined by the number of true positives. That is why it is mostly useful when using imbalanced dataset as was the case in [Eisenbach et al., 2017], where the full GAPs dataset with 6.3 M Patches (of which only 0.7 M are distress) was used to train a deep neural network for the distress detection purpose. However, using F1-binary isn't useful in our case since the number of data points in the positive class (distress) which is 4k is close to the number of data points in the negative class (intact) which is 6k. Macro averaging F1 on the other hand gives equal weight to each class by calculating metrics (precision, recall) for each class, and find their unweighted mean, equations (6.2), (6.3). So the importance of the two classes are considered the same.

$$F1 = 2 * \frac{precision.recall}{precision + recall} \tag{6.1}$$

$$Macro - average - precision = \frac{P1 + P2}{2} \tag{6.2}$$

$$Macro - average - recall = \frac{R1 + R2}{2} \tag{6.3}$$

In the above equations (6.2), (6.3) $P1$, $R1$, $P2$ and $R2$ are the precision and recall for the first and second class respectively. Another metric used to evaluate the performance is the Matthews correlation coefficient, equation (6.4). It could be seen a perfect way of describing the confusion matrix by a single number, its highest value is 1 representing perfect prediction while the value 0 is equivalent to random prediction.

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{6.4}$$

Where $TP$ is the number of true positves, $TN$ is the number of true negatives, $FP$ is the number of false positives and $FN$ is the number of false negatives.

## 6.3 Feature Vectors

In our implementation of Gabor filter, frequency was set to 0.1, however other values $\{0.05, 0.25\}$ and combinations of them were tested. $\sigma_x$ and $\sigma_y$ have values of $\{1, 3\}$ ($\{1, 3, 5\}$ were also tested) and orientation is set to have the values (0°, 45°, 90°, 135°). After the convolution of those 8 resulting kernels with the image we extract both **mean** and **variance** and end up with a feature vector of size 16.

In the implementation of LBP, the number of neighboring points $P$ was set to 32 (tests with different number of neighbours were carried out e.g. 24 ), thus the final feature vector has the length of $P + 2 = 32 + 2 = 34$. As for the GLCM feature extraction method three measures were used namely, **contrast**, **correlation** and **homogeneity** (dissimilarity measure was also added but did not add much improvement and is omitted to reduce the size of the feature vector). Offset was specified as follows: for the distance $d$ the values $(1, 2, 3)$ was used and for theta the values $(0°, 45°, 90°, 135°)$. Moreover the number of intensity values in the gray scale image was not reduced, but rather 256 was used. see chapter 5 for more details on the parameters of these methods.

| Feature Extraction Method | Feature vector size |
|---|---|
| GLCM | 36 |
| LBP | 34 |
| Gabor filter | 16 |

**Table 6.1.** Used features

## 6.4 Evaluation of Feature Extraction Methods

Features from the 3 texture extraction methods GLCM, LBP and Gabor filter were used alone as well as combined on different classifiers. The aim of the this section is to evaluate these feature extraction methods on our dataset. We present the results of each classifier using the different feature extraction methods on both test and validation datasets.

### 6.4.1 Parameter Settings

Table (6.3) shows the results of each classifier using the different feature extraction methods on test dataset. For each classifier, the validation dataset was used to estimate the parameters which gives the best classifier performance (list of tested parameters can be found in the appendix, table (A.1)) and then the best threshold which reflects the highest value of F1-Score (macro) was used on test dataset to calculate the predicted labels. Table (6.2) shows the parameters settings for each classifier.

| Method | Feature-Extraction Method | Parameters |
|---|---|---|
| GMM | GLCM | nr-components=2000 |
| | LBP | nr-components=500 |
| | Gabor filter | nr-component =1000 |
| OC-SVM | GLCM | $\nu = 0.2, \gamma = 0.01$ |
| | LBP | $\nu = 0.2, \gamma = 0.01$ |
| | Gabor filter | $\nu = 0.2, \gamma = 0.1$ |
| KNN | GLCM | $K = 10$ |
| | LBP | $K = 500$ |
| | Gabor filter | $K = 50$ |

**Table 6.2.** Parameter settings for the classifiers for each feature extraction method.

| Feature-Extraction Method | Metric | GMM | OC-SVM | KNN |
|---|---|---|---|---|
| GLCM | F1-Score | 0.42 | 0.5 | 0.43 |
| | AUC | 0.59 | 0.52 | 0.56 |
| | Matthews-Coef | 0.12 | 0.02 | 0.09 |
| LBP | F1-Score | 0.3 | 0.42 | 0.32 |
| | AUC | 0.6 | 0.55 | 0.57 |
| | Matthews-Coef | 0.05 | 0.05 | 0.06 |
| Gabor filter | F1-Score | **0.59** | **0.49** | **0.65** |
| | AUC | **0.64** | **0.58** | **0.71** |
| | Matthews-Coef | **0.18** | **0.04** | **0.3** |

**Table 6.3.** Evaluation of different feature extraction methods on test dataset.

In the literature, these feature extraction methods were used for road crack detection as in [Hu et al., 2010] where features extracted using GLCM and two shape descriptors were used as discriminate features against irregular texture and uneven illumination. Gabor filter is also used in the context of crack detection as in [Salman et al., 2013] achieving 95% precision even when the pavement have very complex and random texture. Although these methods were mostly used for crack detection , however, they have also been used for the detection of other types of distress. For example GLCM method was used for pothole detection in [Wang et al., 2017], were it was compared with wavelet energy field, and for patches detection as in [Hadjidemetriou et al., 2016]. LBP was used for pothole detection in [Naidoo et al., 2015]. This should give us an intuition that the used feature extraction methods should be suitable to use for our task.

However the GAPs dataset contains a very complex textured images, thus we should not expect such techniques to preform well.

Figures (6.3), (6.4) and (6.5) show a t-SNE [van der Maaten and Hinton, 2008] representation of the features extracted from the validation dataset using GLCM, LBP and Gabor filter. T-SNE is a non-linear dimensionality reduction algorithm used for exploring high-dimensional data. It maps multi-dimensional data to two or

more dimensions suitable for human observation. It is obvious from the figures the difficulty of separating the two classes, normal (intact) and abnormal (distress) from each other.

The used features can of course be enhanced. For example [Rampun et al., 2013] suggest carrying out a further processing on each feature extracted from the GLCM in order to improve the results when dealing with noisy, complex textures and unclear boundaries. Also Gabor filter have many parameters which could be tuned to get better features however this process need extensive experimentation as in [Salman et al., 2013].

To compare between the used feature extraction methods we take a look at table (6.3) nad figure (6.1), in which three ROC curves are drawn each one is the result of using one of these feature extraction methods on a GMM classifier. It is obvious from the results that features extracted using Gabor filter perform better. This could be due to the fact that both GLCM and LBP only provides spatial information form an image, with an extra disadvantage of emphasizing too much on local information when using LBP , see section (5.2). On the other hand Gabor filter provides both spatial and frequency information from an image. Same conclusion is induced when using the validation dataset, see figure (6.2) and section (A.2) in the appendix.

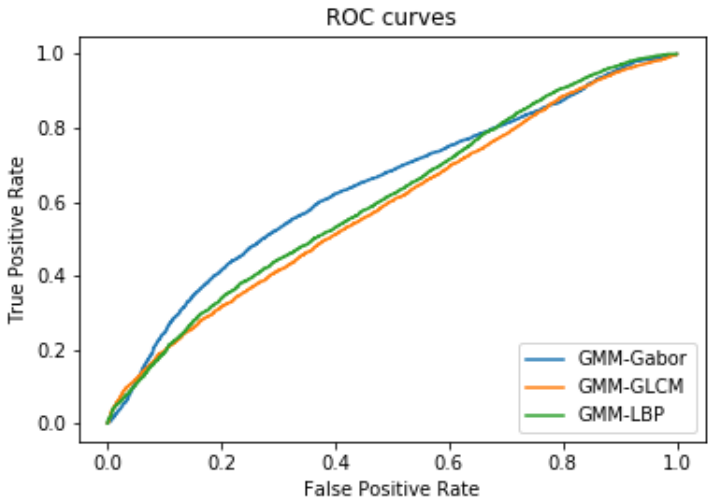



**Figure 6.1.** ROC curves on test dataset using Gaussian mixture model as a classifier and different feature extraction methods.

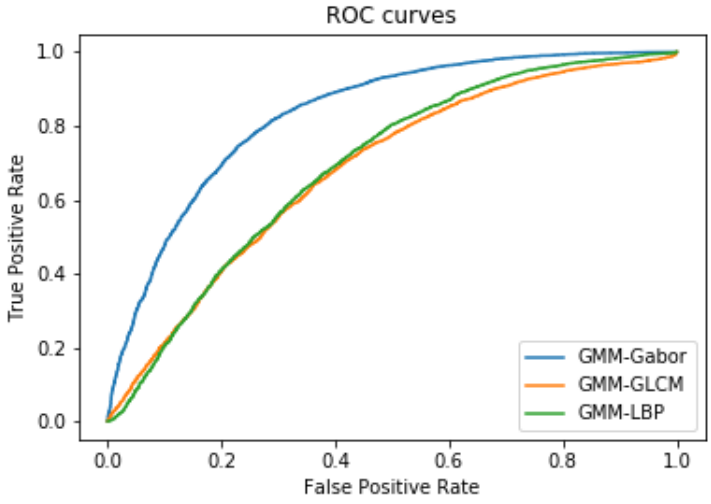



**Figure 6.2.** ROC curves on validation dataset using Gaussian mixture model as a classifier and different feature extraction methods.

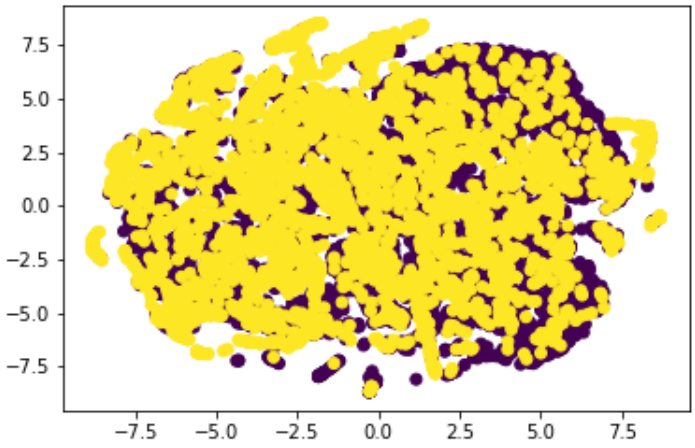



**Figure 6.3.** TSNE representation of validation samples feature vector using GLCM feature extraction method.

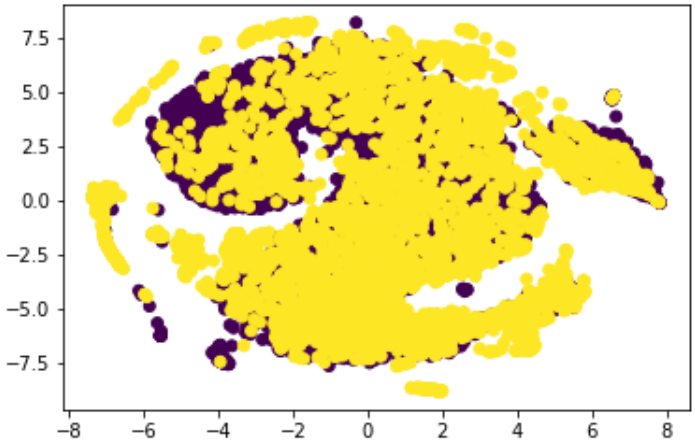



**Figure 6.4.** TSNE representation of validation samples feature vector using LBP feature extraction method.

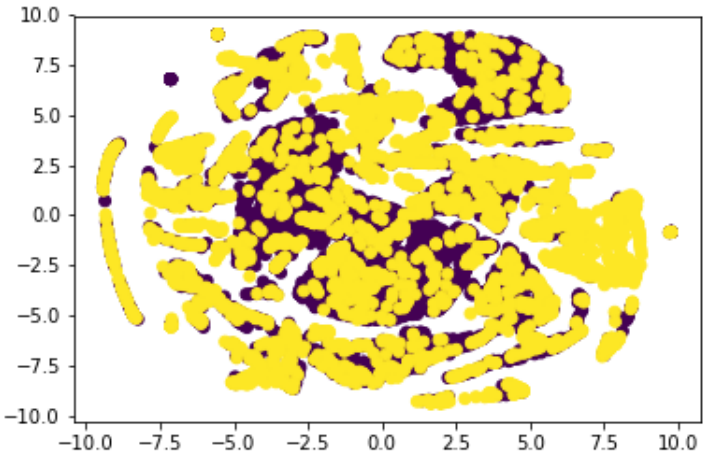



**Figure 6.5.** TSNE representation of validation samples feature vector using Gabor filter feature extraction method.

## 6.5 Effect of Test Dataset

One important observation is that results on validation dataset differ considerably from the results on test dataset, see table (6.4) and figure (6.6) in which two ROC curves on both validation and test dataset, using Guassian mixture model as a classifier and Gabor filter as a feature extraction method, are drawn. This subsequently reflects the classifiers incapability of **generalization**. As mentioned in the Introduction (chapter 1), images of two German federal roads are used for training and a section of one of these roads is used for validation. Whereas images in the test dataset were extracted from a third different road with a better pavement condition than the first two roads. These results comply with the results of [Eisenbach et al., 2017] where they used the full GAPs dataset to train a Convolutional Neural Network (CNN). Refer also to table (6.3) and table (A.2) for comparison between results on test and validation datasets.

| Dataset | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Test | 0.59 | 0.64 | 0.18 |
| Validation | 0.74 | 0.83 | 0.51 |

**Table 6.4.** Comparison of results on test and validation datasets using Gaussian mixture model as a classifier and Gabor filter as feature extraction method.

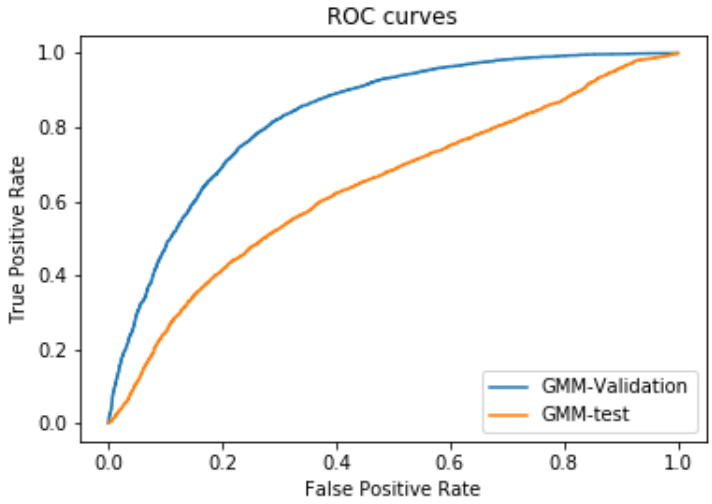



**Figure 6.6.** ROC curves on validation and test datasets using Gaussian mixture model as a classifier and Gabor filter as feature extraction method.

## 6.6 Novelty detection vs Normal Two-class Training

To study the effect of implementing a novelty detection approach, we compare the results of a normal two-class SVM classifier with that of one-class SVM. In order to train a two class SVM on the dataset, we have split the validation dataset into training and validation sets with each containing 3k samples of intact road surface and 2k samples of distressed road surface. Table (6.5) shows the results of comparison on the validation dataset for both classifiers using Gabor filter as features extraction method. See also ROC curves in figure (6.7).

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| SVM | 0.8 | 0.89 | 0.63 |
| OC-SVM | 0.7 | 0.7 | 0.44 |

**Table 6.5.** Comparison of results on validation dataset using SVM and OC-SVM classifiers and Gabor as feature extraction method.

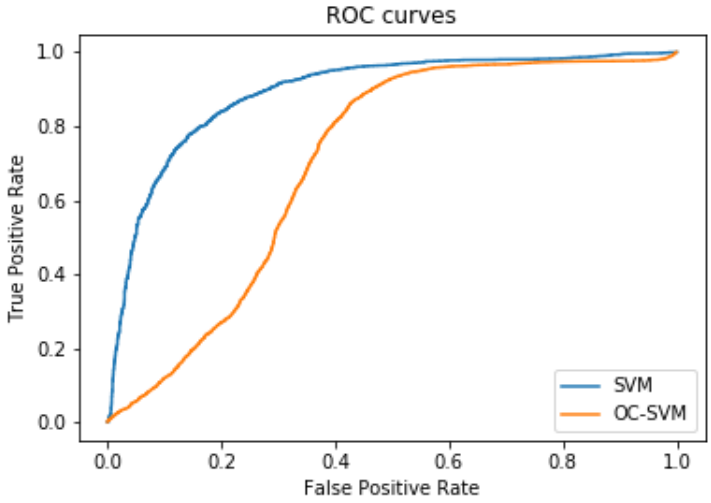



**Figure 6.7.** ROC curves for SVM and OC-SVM classifiers on validation data using Gabor filter.

As mentioned before there have been no reported research using Novelty Detection for pavement distress detection [Pimentel et al., 2014]. Novelty Detection is used in the task of classifying test data that differ in some respect from the data that are available during training (normal dataset), see chapter (3). However, the model of normality built to characterize the normal dataset (whether it is probability-based as in GMM, domain-based as in OC-SVM or distance-based as in KNN) might not be a good model to characterize a complex underlying data structure as the case in the GAPs dataset. The results above show a considerable improvement of the classifiers performance on validation data when trained on both normal (intact) and abnormal (distress) samples.

## 6.7 Methods Comparison

To figure out which are the best features to use, different combinations of features from the three feature extraction methods were tested on validation dataset. A mixture of features from Gabor filter and GLCM delivered the best results for all three classifiers, see table (6.6) and figure (6.8). Also refer to the appendix, section (A.3), for the full tests on validation dataset. It is obvious that GMM and KNN methods are delivering the best performance. GMM belongs to the probability-based approaches of novelty detection which are based on estimating the generative probability density function of the data. This estimate can then be thresholded to define the boundaries of normality. KNN which belongs to the distance-based approaches rely on similar assumptions as GMM. Both probability- and distance-based approaches of Novelty Detection attempt to characterize the area of the data space occupied by normal data. Probabilistic techniques assume that normal data points occur in high density regions and distance techniques like KNN assume that normal data occur close to each other in the feature space while novel data lie far from their nearest neighbours. The main difference between the two approaches is the computational complexity and scalability of the proposed techniques [Pimentel et al., 2014]. This similarity may interpret the close results on validation dataset. On test dataset KNN delivered the worst performance when using features of GLCM and Gabor filter, table (6.7) and figure (6.9). This is due to the inability of KNN to deal with high dimensional data efficiently, because distance measures in high dimensions are not able to differentiate between normal and abnormal data points (see table (6.3) to compare with the performance when using lower dimension feature vector). Domain-based approaches like one-class SVM try to describe a domain containing the normal data by defining a boundary around the normal class but they are insensitive to the specific sampling and density of that class. Moreover these methods are often influenced by outliers in the training dataset. Also the position of the boundary, and hence the performance, can differ significantly upon setting the parameters of the model to different values, see section (4.2).

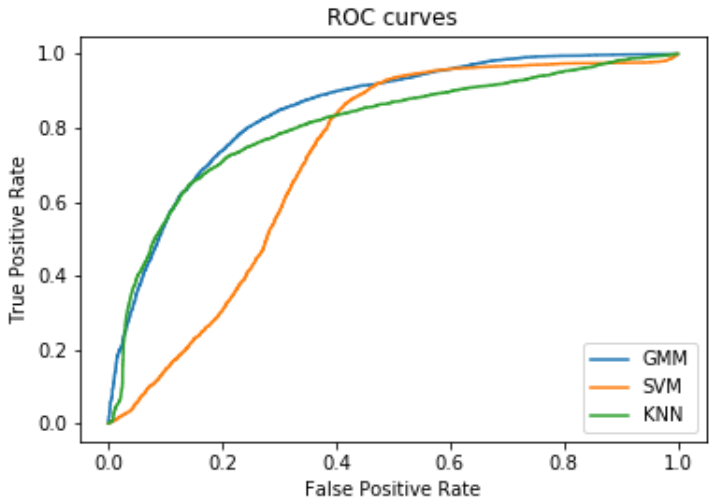



**Figure 6.8.** ROC curves for GMM, OC-SVM and KNN classifiers on validation data using a mixture of features from Gabor filter and GLCM.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Gaussian Mixture Model | **0.77** | **0.85** | **0.55** |
| OC-SVM | 0.72 | 0.72 | 0.45 |
| KNN | **0.74** | **0.8** | **0.5** |

**Table 6.6.** Comparison of results on validation dataset using a mixture of features from Gabor filter and GLCM.

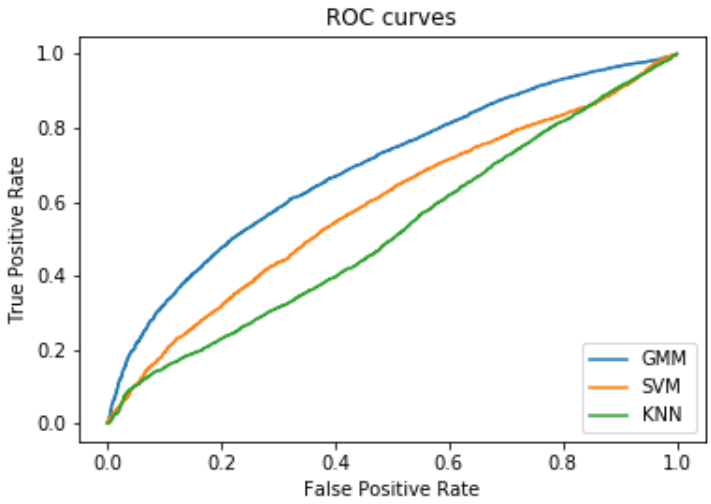



**Figure 6.9.** ROC curves for GMM, OC-SVM and KNN classifiers on test data using a mixture of features from Gabor filter and GLCM.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Gaussian Mixture Model | **0.53** | **0.69** | **0.25** |
| OC-SVM | 0.5 | 0.58 | 0.06 |
| KNN | 0.5 | 0.51 | 0.012 |

**Table 6.7.** Comparison of results on test dataset using a mixture of features from Gabor filter and GLCM.

## 6.8 Comparison with Deep Learning

In [Eisenbach et al., 2017] the GAPs dataset was introduced along with an evaluation of the state of the art in pavement distress detection. The authors used a convolutional neural network for the distress detection purpose with an alternating convolutional and max-pooling layers and two fully-connected layers. Table (6.8) shows the results using the same dataset used in this thesis, see details in section (1.3) on both validation and test datasets.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Validation-dataset | 0.8901 | 0.947 | 0.7812 |
| Test-dataset | 0.8369 | 0.8988 | 0.6759 |

**Table 6.8.** Results on test and validation datasets using deep learning approach on the GAPs dataset.

# Chapter 7

# Conclusion

The aim of this work was to investigate different Novelty techniques that are mostly used in the literature and implement them in our task which is detecting distress in pavement surface. Road maintenance is an essential task in order to ensure a safe road network, and the effort to automate this process has been rapidly increasing in the last decade. Novelty techniques that were implemented include Gaussian mixture model which belongs to the probabilistic approaches of Novelty Detection and which makes use of the distribution of the training data to determine the location of the novelty boundary. We also have implemented one-class SVM method which belongs to the domain-based approaches of Novelty Detection. Domain-based methods determine the location of the novelty boundary using only those data that lie closest to it, and do not make any assumption about data distribution. Finally a distance-based approach namely, K-nearest neighbour was also implemented in which the definition of an appropriate distance measure for the given data is required. These methods assume that normal data lie close to each other in the feature space while novel data points lie far from their nearest neighbours. Moreover features form three texture extraction methods: Gray-level co-occurrence matrix, Gabor filter and local binary pattern were extracted from the data and tested on the validation dataset alone as well as combined to investigate the best features that characterizes the data. When comparing the three feature extraction methods, features from Gabor filter delivered the best performance on both validation and test datasets on all classifiers. This is due to the ability of

Gabor filter to extract both spatial and frequency information from an image while GLCM and LBP only provide spatial information. The performance of GMM and KNN methods on validation dataset were very close and better than the performance of the one-class SVM method. We due this behaviour to the fact that both probability- and distance-based approaches of Novelty Detection attempt to characterize the area of the data space occupied by normal data and they both assume that normal data occur in high density regions. On the other hand domain-based approaches such as OC-SVM try to describe a domain containing the normal data by defining a boundary around the normal class but they are insensitive to the specific sampling and density of that class.

Another important note is that the results on test dataset are much worse than on the validation, because images used for testing were extracted from a different road than the one used for training and validation. Finally from all the introduced results we conclude that using Novelty Detection approach with the aforementioned techniques of feature extraction did not deliver good results. Features could of course be enhanced by doing for example post-processing procedures and performing a more extensive tuning of the parameters in the features extraction method. However the GAPs dataset contains a very complex textured images, thus we should not expect such techniques to preform well. Moreover, comparing our method which is training only using the normal class with a two-class classification method indicated the poor performance of the one-class classification methods.

# Appendix A

## A.1 Tested Parameters for the Classifiers

Table (A.1) shows all tested parameters for the classifiers. Those resulted in the best performance were selected. In the Gaussian mixture model method the covariance type parameter which describes how the covariance matrix for each component is calculated is set to 'Full' and is tested against all other variations: 'tied', 'spherical' and 'diag'.

| Method | Parameters | Values |
|---|---|---|
| GMM | nr-components | 60, 100, 500, 1000, 1500, 2000 |
| OC-SVM | $[\nu, \gamma]$ | [0.1,0.1],[0.01,0.1],[0.01,0.01], [0.001,0.0001],[0.001,0.1],[0.001,0.001], [0.1,0.001],[0.2,0.1],[0.3,0.1],[0.2,0.01] |
| KNN | $K$ | 10, 50, 100, 150, 200, 300, 500 |

**Table A.1.** Tested Parameters for the classifiers.

## A.2 Evaluation of Feature Extraction Methods: Results on validation dataset

| Feature-Extraction Method | Metric | GMM | OC-SVM | KNN |
|---|---|---|---|---|
| GLCM | F1-Score | 0.63 | 0.65 | 0.63 |
| | AUC | 0.68 | 0.7 | 0.72 |
| | Matthews-Coef | 0.27 | 0.31 | 0.3 |
| LBP | F1-Score | 0.64 | 0.63 | 0.57 |
| | AUC | 0.69 | 0.67 | 0.7 |
| | Matthews-Coef | 0.29 | 0.28 | 0.23 |
| Gabor filter | F1-Score | **0.74** | **0.71** | **0.76** |
| | AUC | **0.83** | **0.70** | **0.81** |
| | Matthews-Coef | **0.51** | **0.44** | **0.51** |

**Table A.2.** Evaluation of different feature extraction methods on validation dataset.

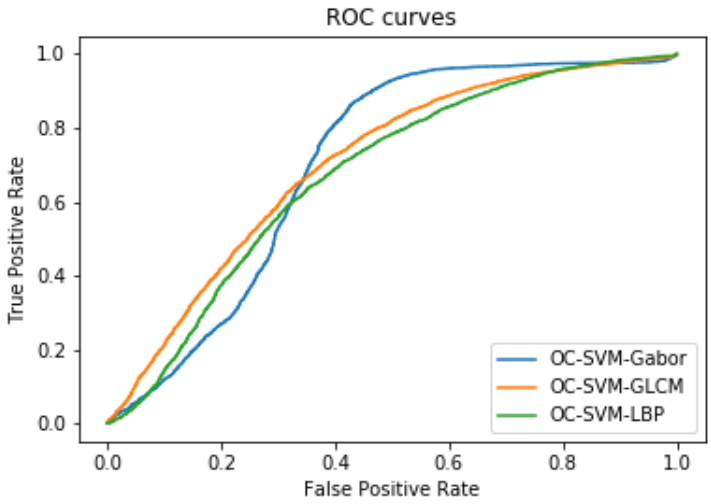



**Figure A.1.** ROC curves on validation dataset using OC-SVM as a classifier and different feature extraction methods.

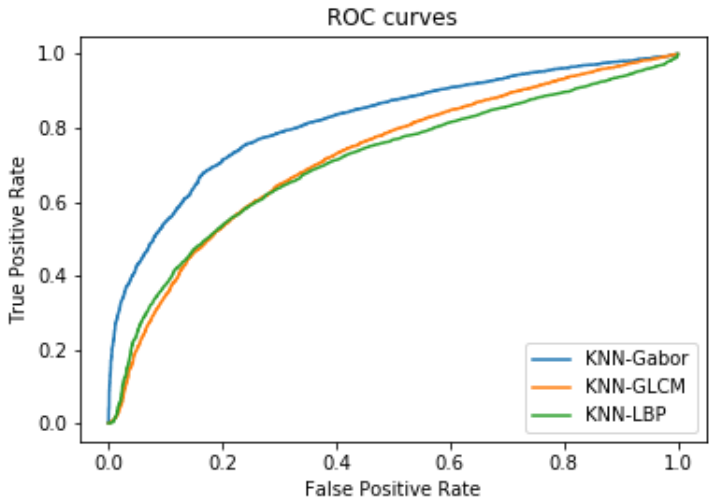



**Figure A.2.** ROC curves on validation dataset using KNN as a classifier and different feature extraction methods.

## A.3 Other Tests on Validation Dataset

The following are different tests on validation dataset using mixture of features.

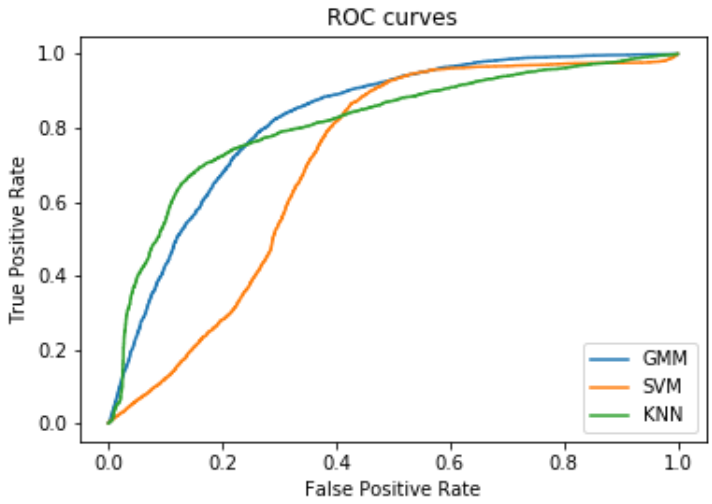



**Figure A.3.** ROC curves for GMM, OC-SVM and KNN classifiers on validation dataset using a mixture of features from Gabor filter and LBP.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Gaussian Mixture Model | 0.76 | 0.82 | 0.52 |
| OC-SVM | 0.71 | 0.7 | 0.44 |
| KNN | 0.76 | 0.81 | 0.54 |

**Table A.3.** Comparison of results on validation dataset using a mixture of features from Gabor filter and LBP.

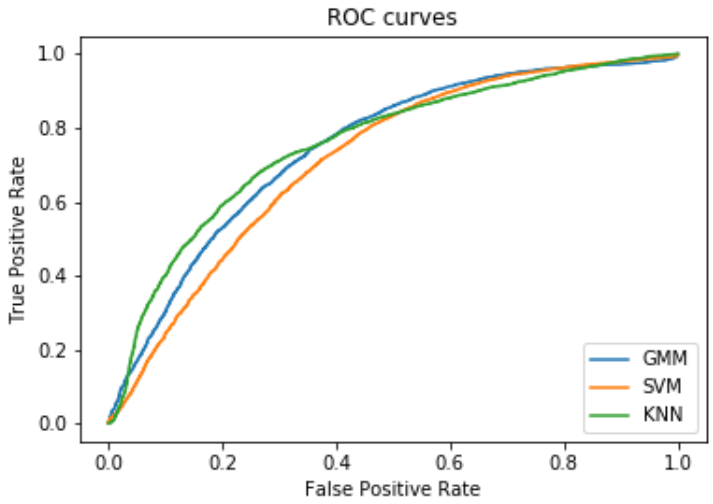



**Figure A.4.** ROC curves for GMM, OC-SVM and KNN classifiers on validation dataset using a mixture of features from GLCM and LBP.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| Gaussian Mixture Model | 0.69 | 0.74 | 0.38 |
| OC-SVM | 0.66 | 0.71 | 0.32 |
| KNN | 0.61 | 0.75 | 0.32 |

**Table A.4.** Comparison of results on validation dataset using a mixture of features from GLCM and LBP.

## A.4 KNN Density-based Method

To compare with the implemented distance-based KNN on the same feature vector used as here, refer to table (6.6). Here the number of neighbours $K$ is set to 100 and the threshold is set to 1.5. See [Cabral et al., 2007] for implementation details.

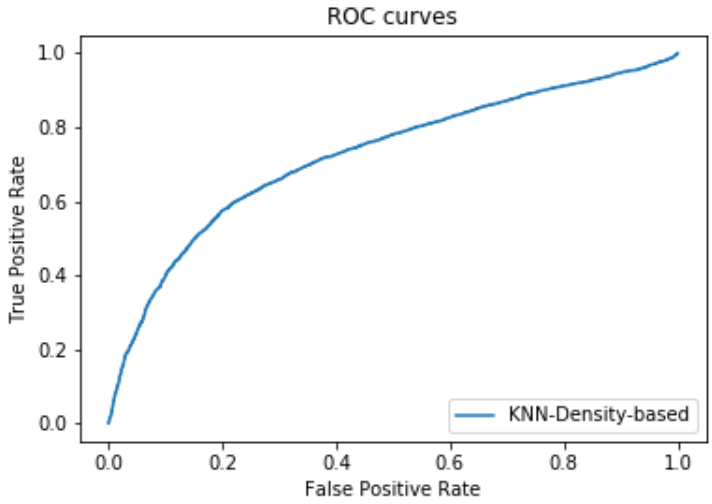



**Figure A.5.** ROC curve for KNN density-based classifier on validation dataset using a mixture of features from GLCM and Gabor filter.

| Classifier | F1-score | AUC | Matthews-Coef |
|---|---|---|---|
| KNN Density-based | 0.67 | 0.72 | 0.37 |

**Table A.5.** Results of KNN density-based method on validation dataset using a mixture of features from GLCM and Gabor filter.

# Bibliography

[Adrian, 2015] Adrian, R. (2015). Local binary patterns with python opencv. `https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/`.

[Ashen, Weerathunga, 2016] Ashen, Weerathunga (2016). Anomaly detection using k-means clustering. `https://wso2.com/library/articles/2016/01/article-anomaly-detection-using-k-means-clustering/`. [Online; accessed October 1, 2017].

[Augereau et al., 2001] Augereau, B., Tremblais, B., Khoudeir, M., and Legeay, V. (2001). A differential approach for fissures detection on road surface images. In *International Conference on Quality Control by Artificial Vision*.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

[Cabral et al., 2007] Cabral, G. G., Oliveira, A. L., and Cahu, C. B. (2007). A novel method for one-class classification based on the nearest neighbor data description and structural risk minimization. In *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pages 1976–1981. IEEE.

[Chambon and Moliard, 2011] Chambon, S. and Moliard, J. M. (2011). Automatic Road Pavement Assessment with Image Processing : Review and Comparison. *International Journal of Geophysics*, page sp.

[Cheung and Deng, 2014] Cheung, Y.-m. and Deng, J. (2014). Ultra local binary pattern for image texture analysis. In *Security, Pattern Analysis, and Cybernetics (SPAC), 2014 International Conference on*, pages 290–293. IEEE.

[Chou et al., 1995] Chou, J., O'NEILL, W. A., and Cheng, H. (1995). Pavement distress evaluation using fuzzy logic and moment invariants. *Transportation research record*, (1505):39–46.

[Clifton et al., 2006] Clifton, D. A., Bannister, P. R., and Tarassenko, L. (2006). *Learning Shape for Jet Engine Novelty Detection*, pages 828–835. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Clifton et al., 2011] Clifton, L., Clifton, D. A., Watkinson, P. J., and Tarassenko, L. (2011). Identification of patient deterioration in vital-sign data using one-class support vector machines. In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pages 125–131. IEEE.

[Daniel and V, 2014] Daniel, A. and V, P. (2014). A novel technique for automatic road distress detection and analysis. *International Journal of Computer Applications*, 101:18–23.

[Eisenbach et al., 2017] Eisenbach, M., Stricker, R., Seichter, D., Amende, K., Debes, K., Sesselmann, M., Ebersbach, D., Stoeckert, U., and Gross, H.-M. (2017). How to get pavement distress detection ready for deep learning? a systematic approach. In *Int. Joint Conf. on Neural Networks (IJCNN)*, pages 2039–2047. IEEE.

[Hadjidemetriou et al., 2016] Hadjidemetriou, G. M., Christodoulou, S. E., and Vela, P. A. (2016). Automated detection of pavement patches utilizing support vector machine classification. In *Electrotechnical Conference (MELECON), 2016 18th Mediterranean*, pages 1–5. IEEE.

[Hall-Beyer, 2000] Hall-Beyer, M. (2000). Glcm texture: a tutorial. *National Council on Geographic Information and Analysis Remote Sensing Core Curriculum.*

[Haralick et al., 1973] Haralick, R. M., Shanmugam, K., et al. (1973). Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621.

[Hofmann, 2006] Hofmann, M. (2006). Support vector machines-kernels and the kernel trick. *An elaboration for the Hauptseminar Reading Club SVM*.

[Hu et al., 2010] Hu, Y., Zhao, C.-x., and Wang, H.-n. (2010). Automatic pavement crack detection using texture and shape descriptors. *IETE Technical Review*, 27(5):398–405.

[Jing and Aiqin, 2010] Jing, L. and Aiqin, Z. (2010). Pavement crack distress detection based on image analysis. In *Machine Vision and Human-Machine Interface (MVHI), 2010 International Conference on*, pages 576–579. IEEE.

[Koch and Brilakis, 2011] Koch, C. and Brilakis, I. (2011). Pothole detection in asphalt pavement images. *Advanced Engineering Informatics*, 25(3):507–515.

[Li et al., 2009] Li, N., Hou, X., Yang, X., and Dong, Y. (2009). Automation recognition of pavement surface distress based on support vector machine. In *Intelligent Networks and Intelligent Systems, 2009. ICINIS'09. Second International Conference on*, pages 346–349. IEEE.

[Lin and Liu, 2010] Lin, J. and Liu, Y. (2010). Potholes detection based on svm in the pavement distress image. In *Distributed Computing and Applications to Business Engineering and Science (DCABES), 2010 Ninth International Symposium on*, pages 544–547. IEEE.

[Manevitz and Yousef, 2001] Manevitz, L. M. and Yousef, M. (2001). One-class svms for document classification. *Journal of Machine Learning Research*, 2(Dec):139–154.

[Markou and Singh, 2003a] Markou, M. and Singh, S. (2003a). Novelty detection: A review—part 1: Statistical approaches. *Signal Process.*, 83(12):2481–2497.

[Markou and Singh, 2003b] Markou, M. and Singh, S. (2003b). Novelty detection: A review—part 2: Neural network based approaches. *Signal Process.*, 83(12):2499–2521.

[Marques, 2012] Marques, A. (2012). Automatic Road Pavement Crack Detection using SVM. Master's thesis, Universidade Tecnica de Lisboa.

[Matlab, ] Matlab, I.-P.-T. Using a gray-level co-occurrence matrix (glcm). `http://matlab.izmiran.ru/help/toolbox/images/enhanc15.html`.

[Naidoo et al., 2015] Naidoo, T., Chiwewe, T. M., Luvhengo, T., Motloutsi, T., and Tyatyantsi, A. (2015). A hybrid human machine system for the detection and management of potholes on asphalt road surfaces. Southern African Transport Conference.

[Nguyen et al., 2009] Nguyen, T. S., Avila, M., and Begot, S. (2009). Automatic detection and classification of defect on road pavement using anisotropy measure. In *Signal Processing Conference, 2009 17th European*, pages 617–621. IEEE.

[NICR, 2016] NICR, P. (2016). Asinvos. `https://www.tu-ilmenau.de/neurob/projects/asinvos/`.

[Nitsche, 2017] Nitsche, F. (2017). Schadstellendetektion mit autoencodern. *Bachelor Thesis*.

[Ojala et al., 2002] Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987.

[Oliveira and Correia, 2009] Oliveira, H. and Correia, P. L. (2009). Supervised crack detection and classification in images of road pavement flexible surfaces. In *Recent advances in signal processing*. InTech.

[Oliveira and Correia, 2013] Oliveira, H. and Correia, P. L. (2013). Automatic road crack detection and characterization. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):155–168.

[Ouyang et al., 2010] Ouyang, A., Luo, C., and Zhou, C. (2010). Surface distresses detection of pavement based on digital image processing. In *International Conference on Computer and Computing Technologies in Agriculture*, pages 368–375. Springer.

[Paalanen et al., 2006] Paalanen, P., Kamarainen, J.-K., Ilonen, J., and Kälviäinen, H. (2006). Feature representation and discrimination based on gaussian mixture model probability densitiesâpractices and algorithms. *Pattern Recognition*, 39(7):1346–1358.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pimentel et al., 2014] Pimentel, M. A. F., Clifton, D. A., Clifton, L., and Tarassenko, L. (2014). Review: A review of novelty detection. *Signal Process.*, 99:215–249.

[Powell and Satheeshkumar, 2016] Powell, L. and Satheeshkumar, K. (2016). Automated road distress detection. In *Emerging Technological Trends (ICETT), International Conference on*, pages 1–6. IEEE.

[Radopoulou and Brilakis, 2015] Radopoulou, S. C. and Brilakis, I. (2015). Patch detection for pavement assessment. *Automation in Construction*, 53:95–104.

[Rampun et al., 2013] Rampun, A., Strange, H., and Zwiggelaar, R. (2013). Texture segmentation using different orientations of glcm features. In *Proceedings of the 6th International Conference on Computer Vision/Computer Graphics Collaboration Techniques and Applications*, page 17. ACM.

[Salman et al., 2013] Salman, M., Mathavan, S., Kamal, K., and Rahman, M. (2013). Pavement crack detection using the gabor filter. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 2039–2044.

[Schölkopf et al., 2000] Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., and Platt, J. C. (2000). Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588.

[sckikit, learn, 2011] sckikit, learn (2011). Simple 1d kernel density estimation. `http://scikit-learn.org/stable/_images/sphx_glr_plot_kde_1d_003.png`. [Online; accessed October 1, 2017].

[Tarassenko et al., 2009] Tarassenko, L., Clifton, D. A., Bannister, P. R., King, S., and King, D. (2009). Chapter 35 novelty detection.

[Tzikas et al., 2008] Tzikas, D. G., Likas, A. C., and Galatsanos, N. P. (2008). The variational approximation for bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146.

[Upadhyaya and Singh, 2012] Upadhyaya, S. and Singh, K. (2012). Nearest neighbour based outlier detection techniques. *International Journal of Computer Trends and Technology*, 3(2):299–303.

[van der Maaten and Hinton, 2008] van der Maaten, L. and Hinton, G. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

[Vlasveld, 2013] Vlasveld, R. (2013). Introduction to one-class support vector machines. `http://rvlasveld.github.io/blog/2013/07/12/introduction-to-one-class-support-vector-machines/`.

[Wang et al., 2017] Wang, P., Hu, Y., Dai, Y., and Tian, M. (2017). Asphalt pavement pothole detection and segmentation based on wavelet energy field. *Mathematical Problems in Engineering*, 2017.

[Wikipedia, the free encyclopedia, 2017] Wikipedia, the free encyclopedia (2017). Gabor filter. `https://en.wikipedia.org/wiki/Gabor_filter#/media/File:Gabor-ocr.png`. [Online; accessed October 2, 2017].

[Zorriassatine et al., 2005] Zorriassatine, F., Al-Habaibeh, A., Parkin, R., Jackson, M., and Coy, J. (2005). Novelty detection for practical pattern recognition in condition monitoring of multivariate processes: a case study. *The International Journal of Advanced Manufacturing Technology*, 25(9):954–963.