Ice Accumulation Detection on Electrical Power Cables Based on Computer Vision

by
Binglin Li

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in Partial Fulfilment of the Requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg

Abstract

Ice storms can cause major power disruptions and detection of ice formation on power cables can help on taking actions for removing the ice before a major problem occurs. A computer vision solution was developed to detect ice on difficult imaging scenarios where illumination and variety of locations can change the image of the cable and background considerably. As the methodology starts with edge detection, support vector regression was applied to predict the upper threshold for the Canny edge detector and it has been proven that this facilitates the later classification accuracy. A total of 44 features based on the gray level co-occurrence matrix, statistical features and Hough transform were extracted and then processed by principal component analysis and sequential forward selection to reduce the feature dimension. Final detection was performed using five different classifiers. It was found that 8 features achieved optimal performance using a support vector machine.

Acknowledgements

With this thesis being completed, I would like to thank people who helped and supported me in various ways. First of all, I would like to express my deepest gratitude to my advisor, Prof. Gabriel Thomas for guiding and advising me over the two years. He is a great advisor who is motivated and knowledgeable. Prof. Thomas spared his much time cultivating me in doing research and helped me with writing this thesis. I greatly appreciate all his support which facilitated my growing in image processing and the related domains.

Besides, I would like to thank the rest of my thesis committees for their invaluable discussion and feedbacks. Thank all the professors in the department of Electrical and Computer Engineering for their good lectures and all the faculty office staff for helping me to deal with various procedures. Also, my sincere thanks go to the technicians Mount First and Guy Jonatschick who helped me with the computer problems and software installation. Furthermore, thank my colleague Huixin Zhang for all our valuable discussions in research and kind friendship during the past two years.

Finally, I would like to thank my family for their love, support, and trust over the years. Without you I could not have done this.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivations

Extreme cold conditions combined with humidity as well as ice storms contribute to ice formation on power cables. Manitoba Hydro removes ice from power lines as quickly as possible to prevent equipment breakage and loss of power. In windy conditions, icy lines with the extra ice weight can swing violently, causing wires to break, wood poles to snap, and even steel towers to crumple.

Ice storms can cause major power disruptions such as the one that occurred on December 2013 that left more than 300,000 customers in Toronto with no electricity [1], and another one hitting eastern United States that in December 2002 left more than 65% of Duke Power's 2.2 million customers without power for days [2]. Power blackouts can be very costly. One can cite the major 2003 event that affected the US and Canada that reduced the gross domestic product in this country by 0.7% [3]. A study by the International Council on Large Electric Systems (CIGRE) showed that ice accretion on power lines, winds or a combination of both, caused 87% of the total damage costs on transmission system failures worldwide in 5 years staring from 1991 [4]. Although catastrophic icing events are not common, an annual average property-related cost of $313,000,000 is incurred in the US along during the year from 1949 to 2000 [5]. In January 1998, an extreme icing storm

caused damage of $1.44 billion in Canada, which gave rise to the largest insured loss in Canadian history [6]. For the event in 2002 [2], the post-storm recovery required 12,500 support personnel.

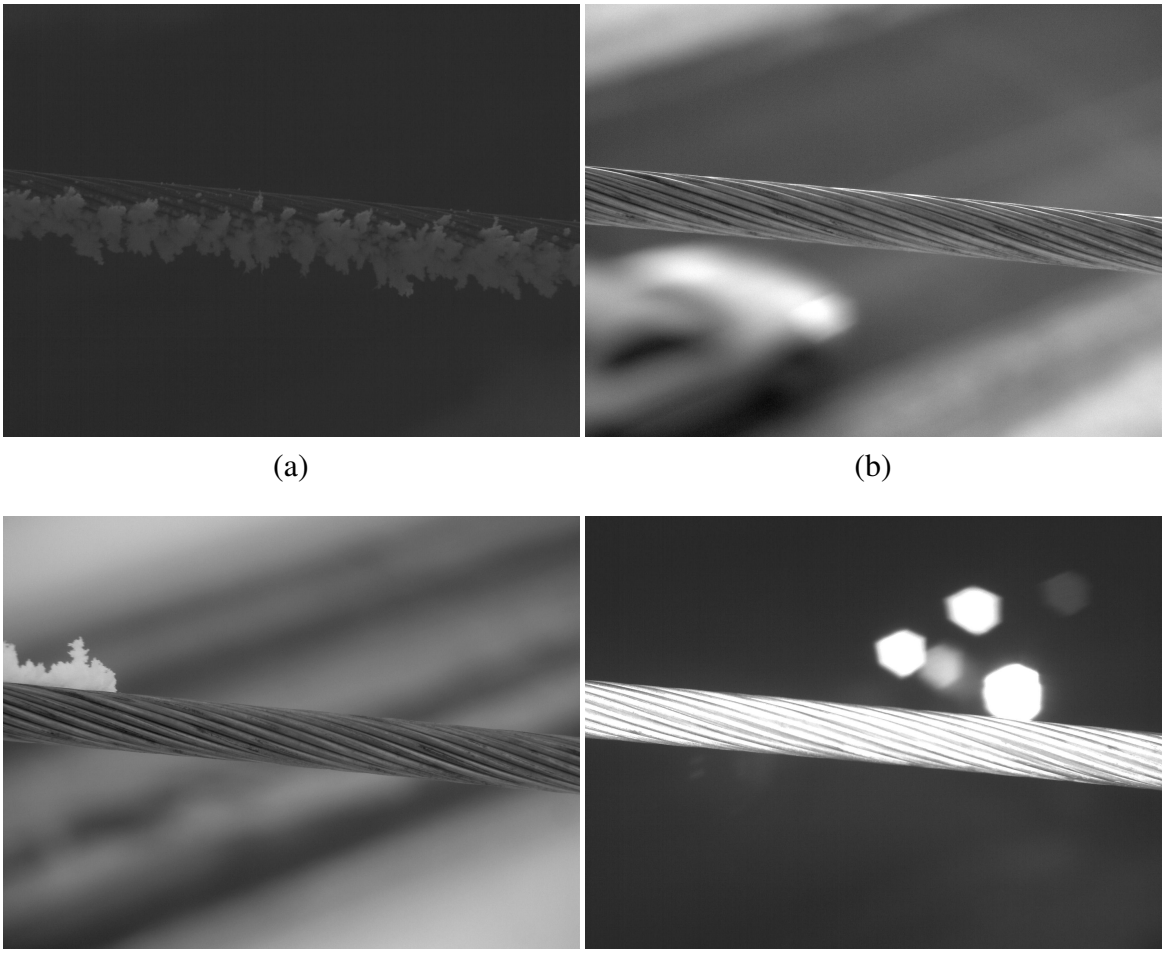
Detection of ice formation and accumulation on power cables can help electric power companies and communities to get prepared and take appropriate actions such as removing the ice before a major problem occurs. It can also facilitate utilities to better plan for recovery by dispatch of repair crews [7]. Hence, the ice accretion detection system can practically decrease the costs associated with icing events.

Many innovative ways have been proposed to address the ice accretion disruption problems. Two major categories of accretion models are developed and used commonly: physical/mathematical models and experimental models [8, 9]. A mathematical model was brought up based on the physical processes of ice accretion [10]. However the parameters involved in these models are difficult to measure [9]. Experimental methods model ice accretion according to the observed weather features and experimental data. In [7] they used meteorological data with a support vector machine to predict the ice accumulation, but as indicated in [7] different learning algorithms can be used to improve on the ice accumulation cases. Some methods based on relative humidity, temperature and dew point sensors tend to be not perfectly reliable and accurate [11]. In [12], 3D shape of ice was recovered to detect the ice accretion on conductors.

Based on a computer vision system, Manitoba Hydro has a frost detection system that uses binary images of the power line as an input as reported in [13] that uses the topological features of the area where the cable is located within the image to assess the amount of ice. The remote field site is located at a desired ice monitoring point of a transmission line, that consists of a camera, a weather station, a wireless antena and an enclosure for additional harware including a PC, router, power supplies and a wireless radio. The Manitoba Hydro system takes images from the overhead line conductors as showed in Figure 1.1 and processes the images only when it is possible for the weather conditions (temperature is between $+5^{\circ}$C and -40°C) to form ice. In this system [13], weather samples are collected

every 5 minutes and ice images are processed every 15 minutes. The systems include 23 ice detection stations installed in a number of ice prone locations throughout the province and these systems provide indications that ice formation was occurring at a certain particular location. It consists of a data collection server that acquires high resolution images and a web based interface. The all-weather camera is attached with an infrared illuminator for night time operation. Therefore, the changing of day and night is not a problem for image collection. Information obtained from the detection system is available to the appropriate ice storm staff using the corporate WAN infrastructure. However this system, as it is currently implemented, yields a number of false positives, an alarm indicating ice formation in a cable that does not present a problem as well as false negatives. As it can be seen in Figure 1.2, challenging cases such as moving objects (cars) and road lights, small part of ice in the corner and low contrast image out of cloudy weather make the detection of ice difficult.

Figure 1.1: Detection system showing camera and weather station



(a) (b)

(c) (d)

Figure 1.2: (a) Low contrast, (b) Passing cars, (c) Small part of ice, (d) Lights behind

As ice is accreted over a period of time, given that the system developed is not necessarily real-time, it is meaningful to alert the technicians in electric power companies to keep eyes on the cables when the snow starts to accumulate.

1.2 Summary of Contributions

This thesis is aimed at solving the local problem faced by Manitoba Hydro that they have to detect ice from power lines as quickly as possible to prevent equipment breakage and loss of power. The solution developed in this thesis can be extended to other locations where people experience similar extreme cold conditions. The contributions can be summarized

as follows:

- In order to improve ice detection, different from the topological features employed in the previous ice detection system in [13], we obtained a total of 44 features based on the Gray Level Co-occurrence Matrix (GLCM), statistical features and features based on the Hough transform, with last four of which are extracted from Canny edge images. Support Vector Regression (SVR) is applied to predict the upper threshold for Canny edge detector. Experiments indicate that Canny parameters predicted by SVR help to achieve better classification performance than the empirical approach.

- Two feature processing techniques from different categories are tested and the results show that Sequential Forward Selection (SFS) is robust among the tested classifiers, while Principal Component Analysis (PCA) is not suited to Bayes classifiers. The final number of features selected is 8, which helps to speed up the processing time.

- Using 25% of the 612 samples for testing in this solution, the best result is 0.0065 which means among the 153 testing samples there is nearly only 1 image that was misclassified, which greatly improved the classification accuracy compared to the previous system used by Manitoba Hydro.

1.3 Block Diagram

Figure 1.3 displays the block diagram for the detection process in a supervised mode. There are training and testing steps. In training stage, images with labels of ice or no ice are preprocessed and then features are extracted from images. If the features have a high dimension, feature processing techniques are applied to reduce the number of features. The processed features are then input to a classifier. The classifier constructs a training model with estimated parameters based on the features from a set of images and their true labels. In testing stage, features are extracted and processed in a similar way to that in the training step from a new set of images without labels. Their labels are predicted based on

the processed new features and the trained model. Here the label is 0 if there is no ice on the cable and 1 otherwise.

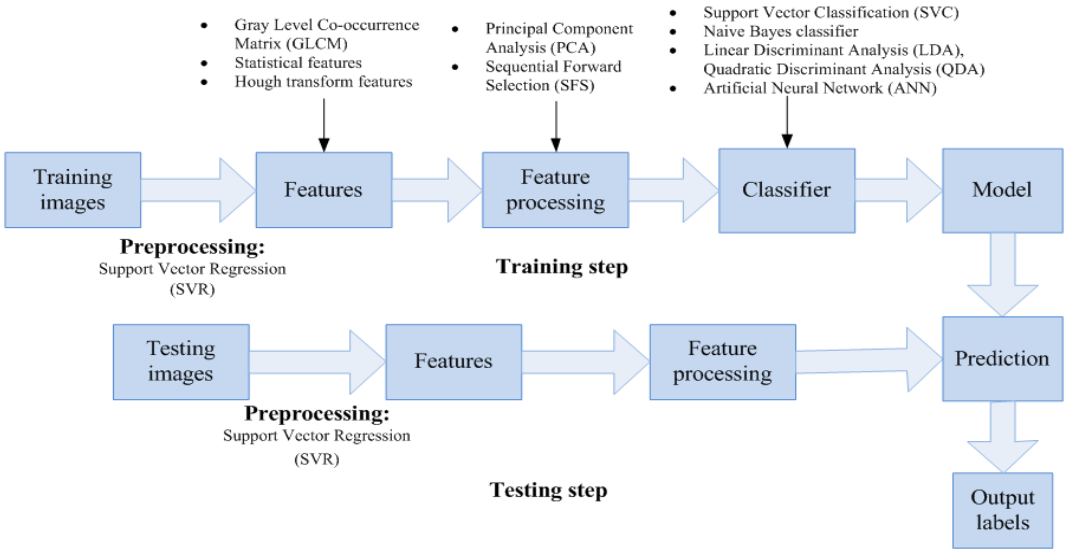


Figure 1.3: Block diagram for the detection process

1.4 Outline of the Thesis

The rest of the thesis is organized as follows. To overcome the issues that are led by some complex backgrounds and challenging scenarios, in Chapter 2 we introduce a preprocessing step that uses Canny edge detection. In this chapter, Support Vector Regression (SVR) is applied to predict the upper threshold parameter for Canny edge detection. In Chapter 3, a total of 44 features based on Gray Level Co-occurrence Matrix (GLCM), statistical features and Hough transform-based features are described and explained how they work in this ice detection system. Based on the features obtained, two feature processing techniques are discussed to reduce the number of features in the way of feature transformation and feature selection respectively in the next chapter. With that being introduced, in Chapter 5 we first turn to two classifiers: Support Vector Classification (SVC) and Naive Bayes classifier. Then Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) are briefly compared with Naive Bayes classifier. Along with that, we present the basic

theory of Neural Network in classification application. Chapter 6 presents the classification results and analysis in detail. The last chapter gives a brief conclusion of this thesis and summarizes some future work.

Chapter 2

Preprocessing with Canny Edge Detector

The first issue to address is the very different background scenarios as shown in Figure 1.2. When a camera is installed, cables are usually stably located in one place on the images under the circumstances of no heavy wind. As it is seen in Figure 1.2, the background area takes up the majority of the whole image. Only the central part of the image where the cable is located provides us the most significant information. To extract strong-separability features from the images, a preprocessing step which focuses on the edge information facilitates to eliminate the influence of the background objects.

2.1 Canny Edge Detector

Canny edge detection [14] assigns pixels as edges at local maxima of the gradient of a Gaussian-smoothed image. Comparisons have been made with various edge detection techniques [15, 16], and it has been proven that Canny edge detection achieves a good performance in many image processing applications [17, 18].

The Canny edge detection scheme follows these steps:

- Smooth the image by convolving a Gaussian filter to reduce the noise. It is inevitable

that images will contain some amount of noise when taken from a camera and edge detection results are easily affected by image noise. Thus it is essential to filter out the noise to prevent false detection caused by noise. The kernel of a Gaussian filter is parameterized by a standard deviation of σ. And the size of the Gaussian kernel will affect the performance of the detector. The larger the size is, the lower the detector's sensitivity to noise. A 5×5 is a good size for most cases, although it also depends on specific situations.

- Compute the gradient magnitude to estimate the edge strength and also calculate the edge directions. Basically the Canny algorithm finds edges where the intensity of the image changes the most, and the gradients of the images determines these areas. The computed edge gradient and direction are : $|G| = \sqrt{G_x^2 + G_y^2}$ and $\theta = arc\tan(\frac{|G_y|}{|G_x|})$, where G_x and G_y are the gradients in the x and y directions respectively.

- Apply the non-maxima suppression to the gradient magnitude to avoid spurious maxima. This step is to trace along the edge in the edge direction and thin the edge by suppressing the non-maxima in the neighbourhood. The process is as follows: compare the edge strength of the current pixel with that in the positive and negative gradient direction. If the edge strength of the current pixel is largest, preserve the value of the current edge strength. If not, remove the value. This algorithm is applied for each pixel in the gradient image.

- Utilize double thresholds to the gradient in order to reduce the false edge points and link the gaps in the detected edges as some edge-pixels remaining after the non-maximum suppression may be caused by noise or color variations for instance due to rough surfaces. Any pixel that has a higher value than the upper threshold is presumed to be an edge point. Any pixel value that is less than the lower threshold is not considered as edge pixel. The value between the two thresholds can be an edge point if it's connected to other edge points.

For the images in Figure 1.2, the corresponding edge images are given in Figure 2.1.

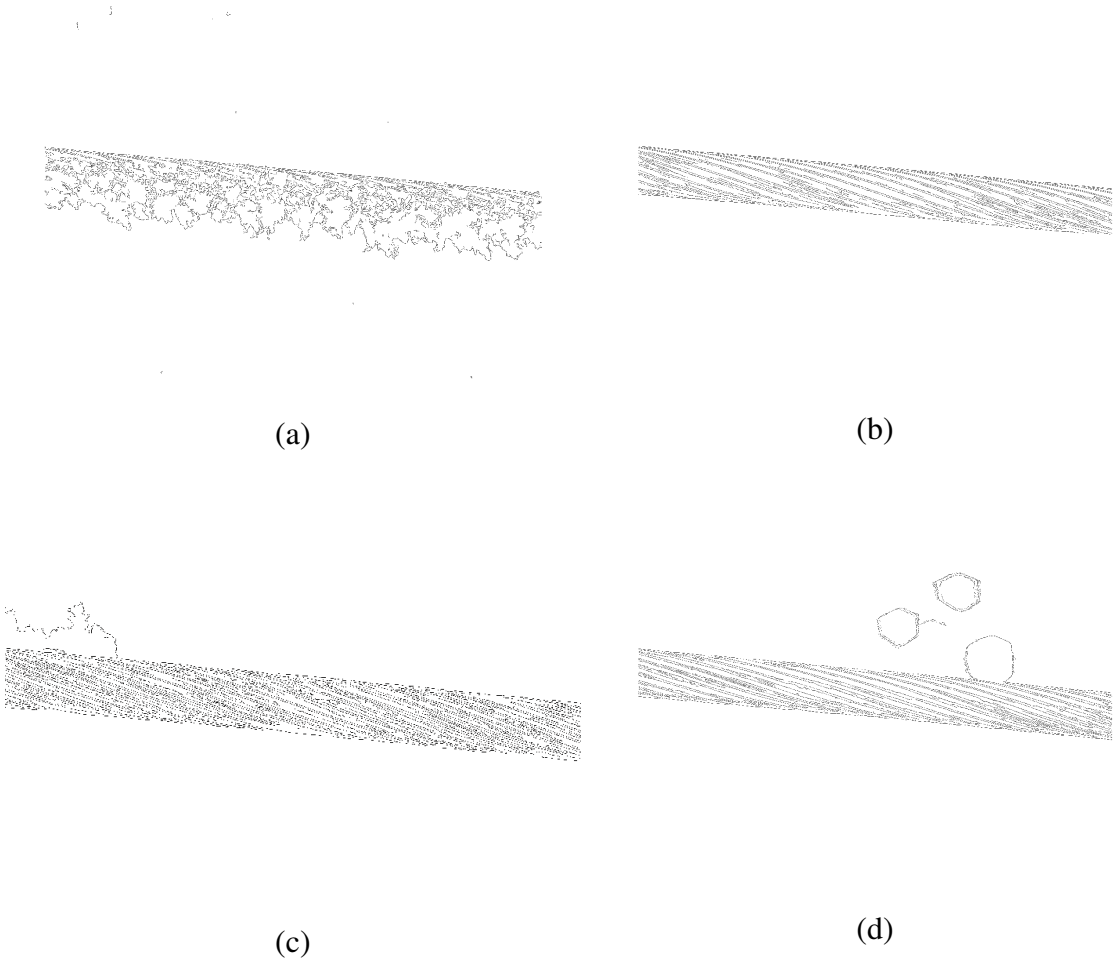


Figure 2.1: Canny results: (a) Low contrast, (b) Passing cars, (c) Small part of ice, (d) Lights behind

This preprocessing step is important because it determines how much edge information will be used to compute some features. The selection of Canny parameters can lead to evident changes of the extracted features and thus affect the detection accuracy. In Matlab, the lower threshold is set to 0.4 times of the upper threshold, while the upper threshold is chosen by users interactively. The value of the standard deviation σ associated to the Gaussian filter was set to $\sqrt{2}$ for all the cases and the size of the filter is chosen automatically based on σ. The main focus is on the upper threshold selection. When the number of images is small, the empirical method by examination of all the images is the easiest way. However, for thousands of images this manual selection method is not practical.

2.2 Empirical Method

The empirical method was conducted on 49 images. Based on the upper threshold values for these images, we established an approximate relationship between the global standard deviation of the original image σ_G and the upper thresholds denoted as T_c, since the global standard deviation reflects the contrast of the image.

Table 2.1: Assignment of threshold based on the global standard deviation

Global standard deviation	σ_G <10	10 $\leq \sigma_G$ <15	$\sigma_G \geq 15$
Canny threshold	$T_c = 0.18$	$T_c = 0.46$	$T_c = 0.13$

2.3 Support Vector Regression Method

The Support Vector (SV) method has been proven to be an effective tool for coping with multidimensional classification problems. The core idea in the classification work with Support Vector Classification (SVC) is to maximize the minimum geometric margin for all the training dataset. The boundary is determined by the data points which are located on and between the margin tube, and also the misclassified points in soft-margin problems. These key points are known as support vectors. Those points far away contribute zeroes to the coefficients, bringing in sparseness in the support vectors. By introduction of alternative loss functions, such as a least square error function, Laplacian loss function and Huber loss function, support vector method can also be applied to regression problems [19]. However, the three loss functions above produce no sparseness. To overcome this issue, ε insensitive loss function [Vapnik, 1995] was proposed to obtain a sparse set of support vectors showed in Figure 2.2. Support Vector Regression (SVR) gained a growing popularity over the years. In the regression problems, the support vectors are the points out of the regression tube. kernel methods also work well for nonlinear problems by mapping onto higher dimensions without bringing in much computational complexity.

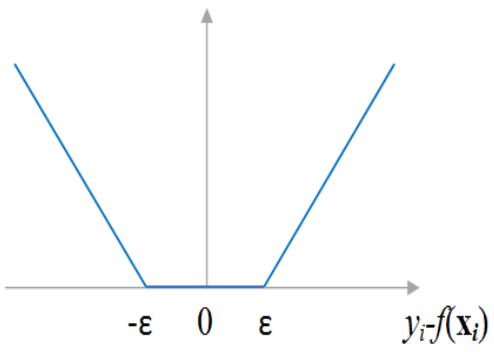


Figure 2.2: $\varepsilon-$insensitive loss function

2.3.1 Basic Theory of Support Vector Regression

In regression formulation, the goal is to estimate a continuous function based on a finite number set of training examples, denoted by $(\mathbf{x}_i, y_i)$, where $i = 1, \cdots, n$, d-dimensional input $\mathbf{x} \in R^d$ and $y \in R$. In the ε insensitive loss function, the SVR finds a function that at most ε deviation from the true label y_i for all the training data and keeps as flat as possible as well [20]. To be specific, first the input $\mathbf{x}$ is mapped to a higher dimensional feature space, denoted as m dimensional space. In that high feature space, a linear model $f(\mathbf{x}, \mathbf{w})$ is constructed as

$$f(\mathbf{x}, \mathbf{w}) = \sum_{j=1}^{m} w_j g_j(\mathbf{x}) + b \tag{2.1}$$

where $g_j(\mathbf{x})$, $j = 1, \cdots, m$, represents a set of nonlinear transformations, and b is the bias.

The ε insensitive loss function $L(y, f(\mathbf{x}, \mathbf{w}))$ is given by

$$L_\varepsilon(y, f(\mathbf{x}, \mathbf{w})) = \begin{cases} 0, & \text{if } |y - f(\mathbf{x}, \mathbf{w})| \leq \varepsilon \\ |y - f(\mathbf{x}, \mathbf{w})| - \varepsilon, & \text{otherwise} \end{cases} \tag{2.2}$$

The risk is:

$$R(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} L_\varepsilon(y, f(\mathbf{x}, \mathbf{w})) \tag{2.3}$$

To be flat, a small $\mathbf{w}$ is preferred which requires to minimize the $||\mathbf{w}||^2$. Non-negative variables ξ_i, ξ_i^*, $i = 1, \cdots, n$ are introduced to measure the deviation of training samples

outside ε insensitive zone. Figure 2.3 shows the geometric meaning of the ε value, slack variables ξ and ξ^*. So the minimization problem is given:

$$\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\Sigma_{i=1}^{n}(\xi_i + \xi_i^*) \tag{2.4}$$

$$s.t. \begin{cases} y_i - f(\mathbf{x}_i, \mathbf{w}) \leq \varepsilon + \xi_i^* \\ f(\mathbf{x}_i, \mathbf{w}) - y_i \leq \varepsilon + \xi_i \\ \xi_i, \xi_i^* \geq 0, i = 1, \cdots, n \end{cases}$$

Parameter $C > 0$ determines the trade off between the model complexity (flatness) and the degree to which deviations larger than ε are tolerated in the optimization formulation. If C is too large (infinity), then the objective emphasizes much on the risk, without regard to model complexity part in the optimization formulation, which may lead to be overfitting. On the other hand, if C is too small (close to 0), the objective will do in the opposite way, which can yield to underfit.

Parameter ε controls the width of the ε insensitive zone. The value of ε will influence the number of support vectors. The bigger the ε, the fewer support vectors. Also the bigger ε values result in more flat estimates, thereby under a risk of being underfitting.

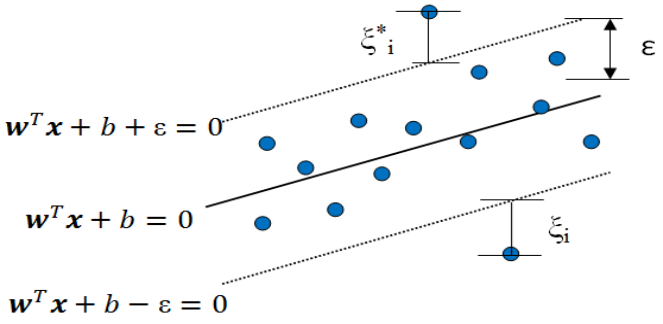


Figure 2.3: Geometric meaning of the ε, slack variables ξ and ξ^*

Similar to the SVC, this optimization problem can be written to its dual form. From Karush-Kuhn-Tucker (KKT) conditions [Karush, 1939, Kuhn and Tucker, 1951], the

solution is

$$f(\mathbf{x}) = \Sigma_{i=1}^{n_{SV}}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}), \qquad \text{s.t} \quad 0 \leq \alpha_i^* \leq C, \quad 0 \leq \alpha_i \leq C \tag{2.5}$$

where n_{SV} is the number of support vectors and the kernel function is

$$K(\mathbf{x}, \mathbf{x}_i) = \Sigma_{j=1}^{m} g_j(\mathbf{x}) g_j(\mathbf{x}_i) \tag{2.6}$$

To find the coefficients α^* and α, we need to maximize the function

$$W(\alpha^*, \alpha) = -\varepsilon\Sigma_{i=1}^{n}(\alpha_i^* + \alpha_i) + \Sigma_{i=1}^{n} y(\alpha_i^* + \alpha_i) - \frac{1}{2}\Sigma_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i^* - \alpha_i)(\alpha_j^* + \alpha_j)K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.7}$$

subject to constraints

$$\Sigma_{i=1}^{n}(\alpha_i^* - \alpha_i) = 0, \qquad 0 \leq \alpha_i, \quad \alpha_i^* \leq C, \qquad i = 1, \cdots, n \tag{2.8}$$

Note that the kernel algorithm can be computed in the terms of dot products between the set of examples [21], which contributes to less computational complexity.

From the KKT conditions, it is stated that at the optimal solution the product between dual variables and constraints has to vanish. So in the SV case,

$$\begin{aligned} \alpha_i(\varepsilon + \xi_i - y_i + < \mathbf{w}, \mathbf{x}_i > + b) = 0 \\ \alpha_i^*(\varepsilon + \xi_i^* + y_i - < \mathbf{w}, \mathbf{x}_i > - b) = 0 \end{aligned} \tag{2.9}$$

and

$$\begin{aligned} (C - \alpha_i)\xi_i = 0 \\ (C - \alpha_i^*)\xi_i^* = 0 \end{aligned} \tag{2.10}$$

Firstly samples $(\mathbf{x}_i, y_i)$ with corresponding $\alpha_i^{(*)} = C$ lie out of the εinsensitive tube. Secondly $\alpha_i \alpha_i^* = 0$, which means at least one of the variables α_i, α_i^* is zero as we cannot have nonzero slacks in both directions. Finally for $\alpha_i^* \in (0, C)$ we have $\xi_i^* = 0$ and the

second factor in Equation (2.9) has to vanish. Hence b can be computed as follows:

$$\begin{aligned} b &= y_i - < \mathbf{w}, \mathbf{x}_i > -\varepsilon \quad \text{for} \quad \alpha_i \in (0, C) \\ b &= y_i - < \mathbf{w}, \mathbf{x}_i > +\varepsilon \quad \text{for} \quad \alpha_i^* \in (0, C) \end{aligned} \tag{2.11}$$

Another method to compute b is to combine together with $\mathbf{w}$ to solve in the context of optimization which needs to add an extra feature valued as 1 for all observations. But actually the bias term does not make much significance in practice [22].

2.3.2 Support Vector Regression with Radial Basis Function kernel

We utilized LIBSVM [23] to carry out the SVR. The Radial Basis Function (RBF) kernel was applied to map the input features to certain higher feature space. As the number of features is not high, usually we don't fit the features with a linear kernel. Fitting with a non-linear kernel may be more effective. Another advantage for RBF kernel is the less number of heperparameters which influences the complexity of the model selection [24]. The RBF kernel is given by

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma ||\mathbf{x}_i - \mathbf{x}_j||^2} \tag{2.12}$$

where γ is the RBF kernel parameter. In ε insensitive SVR, there are three parameters (C, ε, γ) that are interactively selected by users.

2.3.3 Three Global Features for Support Vector Regression

It is not common to input the whole image as features to learn the regression model because of the huge amount number of pixel features. In this work, all images are high resolution with size of 1536 by 2048 pixels. A total of 49 images are chosen from different background scenarios and the upper threshold values were carefully examined by obtaining the best edge images.

In this regression work we prefer that the features extracted from the whole images are easy to compute and at the same time can represent the global information regardless of icy

images or no-icy images, since separable features are more appropriate for the classification part. As in the previous work the global standard deviation is an important element to determine the Canny parameter, we add the mean intensity and the entropy of the image as the other two features. The three features reflect the contrast, average intensity and statistical uncertainty of grey levels.

Before using SVR, we also need to scale the features. One advantage of scaling the features is to avoid the larger numeric values dominating the smaller numeric values. The other advantage is to facilitate the numerical computation of the inner products aroused in kernel methods [24].

There are many ways to scale features. One way is to normalize the features to the range of 0 to 1 or -1 to 1. The difference is that the former only has positive values while the latter has negative values. Both the training and testing data should be scaled in the similar way. Here to guarantee the uniformity, we processed the data by mapping each feature with zero mean and standard deviation of 1. Note that support vector method requires that each data instance is represented by a vector of features. If the feature is categorical, we need to convert it to numeric data.

2.3.4 5-fold Cross Validation

Cross validation is a model evaluation method to indicate how good the learner is in the prediction process. K-fold cross validation is to divide the data set into k subsets. At each time, one of the k subsets is used for training and the other $k-1$ subsets are out together to make a prediction. After k trials, the average squared error is computed. The advantage of this method is that it matters less how the data are divided.

Without loss of accuracy, 5-fold cross validation were applied to find the optimal parameters. The preferable C and γ range from 10^{-3} to 100 and 10^{-6} to 10 respectively, spaced uniformly in log scale. For ε, usually the default set is 1.0, 0.1 or 0.01. To compare the different parameter schemes, we fix ε at certain value and set the selection range from 2^{-5} to 2^{15} for C and 2^{-10} to 2^{15} for γ. The average squared error is calculated after k trials

for each parameter scheme and the minimum error among all the schemes for the fixed ε is recorded. The ε which has the smallest recorded error and the corresponding the C and γ are the optimal parameters for ε insensitive SVR.

2.4 Summary

In this chapter, we introduce the preprocessing step with Canny edge detection to eliminate the background influence. Two methods are presented: empirical method and parameter prediction based on SVR. The latter method is a machine learning based technique. It is applied with RBF kernel to make nonlinear regression. The input are three global features which reflect the contrast, average intensity and statistical uncertainty of grey levels. 5-fold cross validation is employed to find the appropriate SVR parameters.

Chapter 3

Image Features for Classification

An important aspect to consider was the selection of the features to be used for classification. Effective features from image data is crucial because it is meaningless to compare performance on different classifiers if the features are not representative and meaningful. A reduced number of features was required as the system was meant to analyze images at many different locations and albeit a real time signal processing system was not mandatory, we wanted it to be fast enough so that not to depend on an expensive computing set up.

3.1 Features Based on Gray Level Co-occurrence Matrix

As indicated in [25], texture reflects the innate property of many surfaces. It contains important information regarding the structural arrangement of surfaces and their relationship to the surrounding environment and can be used for discrimination.

The Gray Level Co-occurrence Matrix (GLCM) is a popular feature extraction technique that has been widely used in image processing [26–30]. GLCM calculates how often the gray-level value a of a pixel occurs conformed to certain spatial relationships with a pixel that has a value b. It can reveal that the images have fine or coarse textures, or how they are texturally uniform. The final features based on GLCM are calculated as second order statistical textures [27]. GLCM are extracted in two main steps. First the

pairwise spatial co-occurrence of pixels separated by a particular angle and distance are tabulated using a GLCM. Second, a set of scalar quantities also known as features that reveal different aspects of the underlying texture. A total of fourteen textural features were proposed in [25] based on GLCM properties. In [31, 32], some more properties based on GLCM were proposed.

For example, the correlation feature is a measure of linear dependency between two pixels over the whole image. It indicates how a pixel is correlated to another pixel in the particular direction defined in the predicate that specifies the direction between pixels a and b mentioned above. Let $p(i,j)$ be the $(i,j)^{th}$ entry of the GLCM, and $p_x(i)$ is the i^{th} entry in the marginal probability matrix defined as $p_x(i) = \Sigma_{j=0}^{L} p(i,j)$, where $L+1$ is the total number of gray level values. Similarly, $p_y(j) = \Sigma_{i=0}^{L} p(i,j)$. Then the correlation feature is calculated as

$$\text{Correlation} = \frac{\Sigma_{i=0}^{L}\Sigma_{j=0}^{L}(i,j)p(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y} \tag{3.1}$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviation of p_x and p_y.

When there are repeated patterns, the GLCM correlation values tend to be high. Usually the ice is located on the top or at the bottom of the cables. Therefore, the correlation for images with ice are lower than those without ice in the particular direction same as the top and the bottom lines of the cable. For the orientation that the cables extend, the GLCM correlation values for the images in Figure 1.2 are 0.8623, 0.9690, 0.9716 and 0.9872 respectively, which indicates that the GLCM correlation feature can be an important feature to discriminate the icy and non-ice images.

Another feature, contrast, measures the amount of local variations in an image. The contrast is quantified as

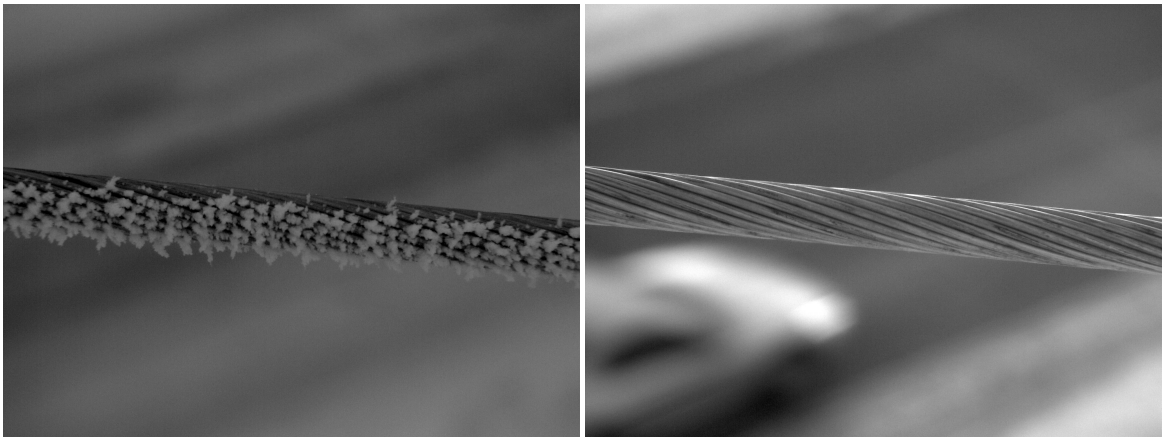
$$\text{Contrast} = \Sigma_{i=0}^{L}\Sigma_{j=0}^{L}(i-j)^2 p(i,j) \tag{3.2}$$

For a constant image, the GLCM contrast is 0. The random ice on the cables results in different local variations from that for non-ice images at different places. For the direction along the twisted lines on the cables, the images with ice have more local

variations than those without ice and therefore have higher contrast values. For example, the GLCM contrast values are 0.1163 for Figure 3.1(a) and 0.0806 for Figure 3.1(b). However for images in Figure 1.2, the contrast values are 0.0197, 0.0806, 0.0440 and 0.0667 respectively. Therefore, this feature is not discriminative in appearance, but may be important when the feature dimension is increased and a non-linear classifier is applied.

In general, to reduce the computational complexity, only a number of relevant features are preferred. For example in [26] only four features (energy, entropy, contrast and inverse difference moment) were applied and achieved better performance than Gabor wavelet features. The work presented in [29] applied only seven features. In addition, some properties of the GLCM are proportional or inversely proportional to other features [26] and not all the features contribute equally to the classification performance.

From the images, it is seen that two particular directions as shown in Figure 3.2 can be used to set the offset for the GLCM properties. One was along the angle that the cables extend (offset = [1 7]). The other was as the same as the twisted lines on the cables (offset = [1 3]). For icy images, the accumulated ice will disturb the repeated patterns on the cable images, thus leading to icy images discriminating from the non-ice images. Based on [25, 31, 32], 20 different GLCM properties were obtained for each direction. Therefore, we have as many as 40 GLCM features altogether. To extract more original information, these 40 features were calculated on the grey images. The number of gray-levels was scaled to 8, which benefited to eliminate the background influence. The other 4 features were computed from the edge images. Basically, we concatenated the features derived from the original images and those obtained from edge images.



(a) (b)

Figure 3.1: (a) Icy image, contrast = 0.1163, (b) Non-ice image, contrast = 0.0806

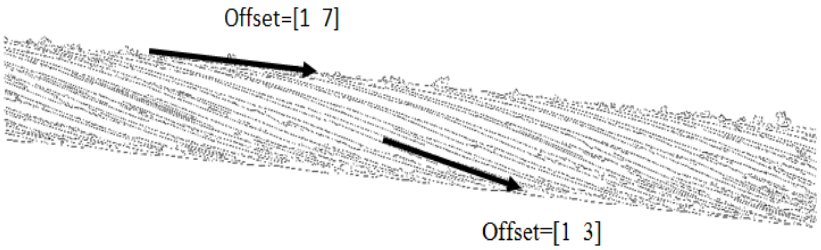


Figure 3.2: Two particular directions used

3.2 Statistical Features

3.2.1 Central Standard Deviation Feature

The standard deviation can be used as a measure of texture. It was found in [33] that in general, low order statistics (mean, variance, etc.) perform about as well as high order statistics. Quite different areas of application such as forest classification and dermoscopy

analysis have successfully used these statistical features [34, 35].

As accumulated ice on the cable yields high variance values compared to cables with no ice, we calculated the standard deviations. To avoid the influence of the background points, edge detections were performed first. Let $I(x, y)$ denotes the gray-level value in image I. The standard deviation σ_C can be obtained by

$$\sigma_C = \sqrt{\frac{1}{N * M - 1} \Sigma_{x=1}^{N} \Sigma_{y=1}^{M} |(I(x, y) - \mu_C)|^2} \tag{3.3}$$

where $\mu_C = \frac{1}{N*M} \Sigma_{x=1}^{N} \Sigma_{y=1}^{M} I(x, y)$ and N, M are the size of the matrix used to compute the standard deviation.

3.2.2 Width Feature of Cables

The width changes of the cable were calculated on the binary images on each image column where the measurement of width is that as shown in Figure 3.3. The maximum and minimum vertical coordinates for each column pixel was found and the difference of these two values was considered to be the width of the cable. The width should be computed based on edge images.

Two decisions were made to eliminate errors brought by the missed edge points on the cable and extra background points.

- The width of the cable in no-ice conditions does not change much. It keeps a value around 200 pixels, while in images with ice is larger than 200 pixels. So when the computed width for a certain column is less than 200, we set the width equal to that from the previous column. This step can reduce the width error which results from the detected discontinuous edges present on some patterns on the surface of the cables.

- Isolated points in the background can result in a sharp increase of width change between two successive columns. As the ice accumulating between two adjacent columns is fairly continuous, the difference of width change between two neighbor-

ing columns which is larger than 50 pixels can be regarded as isolated points, thereby we also set the width as that found in the previous column.

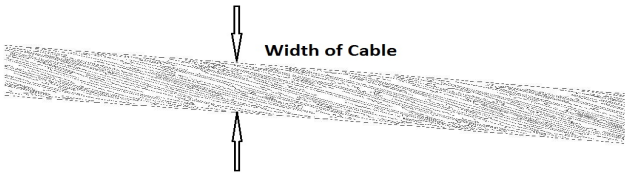


Figure 3.3: Defined width of the cable

The standard deviation of the width changes was the 42th feature in this application. Assume $d(i)$ represents the i^{th} value of cable width along the horizontal axis. The standard deviation σ_W is calculated as

$$\sigma_W = \sqrt{\frac{1}{N-1}\Sigma_{i=1}^{N}(d(i)-\mu_W)^2} \tag{3.4}$$

where $\mu_W = \frac{1}{N}\Sigma_{i=1}^{N}d(i)$ and N is the horizontal dimension of image I.

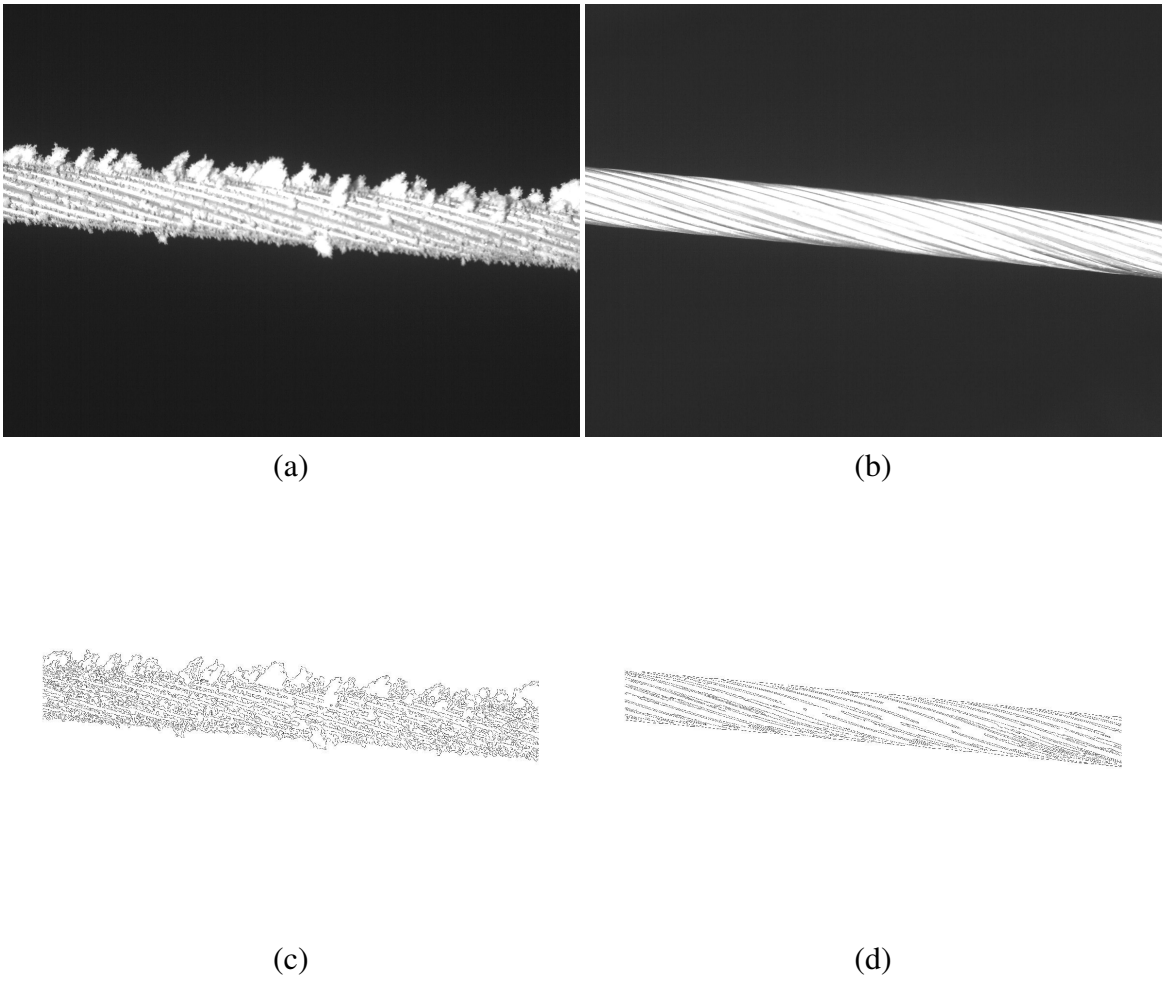


Figure 3.4: (a) Icy image, (b) Non-ice image, (c) and (d) Canny edge images for image(a) and (b)

Figure 3.5 shows the width changes for the images shown in Figure 3.4(a) and (b). The standard deviations of the width changes calculated are 542.9248 for the ice case and 3.3192 for the no ice scenario, from which we can see the standard deviation of the width changes derived from the edge images is quite discriminative. This point can be confirmed by the classification performance when processed by the feature selection technique in Chapter 6. This feature ranked first in the order of the importance in terms of the classification performance with the classifiers discussed later.

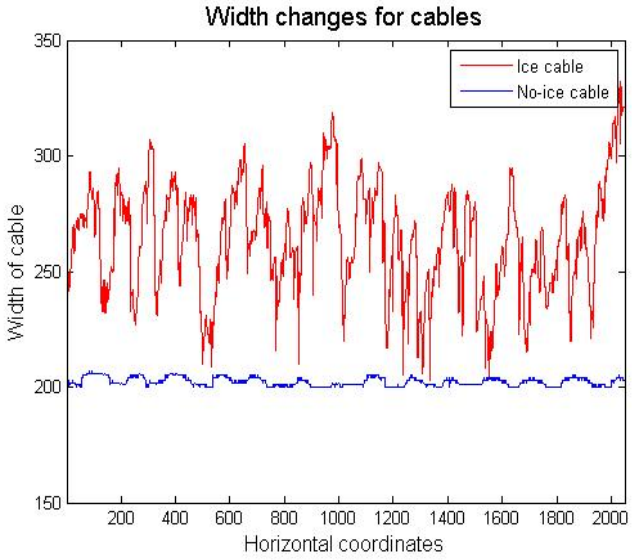


Figure 3.5: Width changes for icy and non-ice images

3.3 Features Based on Hough Transform

The Hough transform is a popular technique that detects lines in pictures [36]. In fact, this technique can be used to detect lines, circles or other structures if their parametric equation is known, and it can given robust detection under noise and partial occlusion. For every point x and y on a binary image, a line crossed it can be expressed by

$$y = mx + b \tag{3.5}$$

where m is the slope and b is the intersection in the Cartesian coordinate system. If we think backwards, we can instead express Equation (3.5) as

$$b = -x \cdot m + y \tag{3.6}$$

Now let m and b be the variables. This is a transformation from (x, y)space to (m, b)space.

For every given point (x, y), it can be represented in (m, b)space with a line. When a set of points in (x, y)space are on the same line, in (m, b)space there will be an intersection point of the lines corresponding to the set of points. The (m, b)space is not commonly used because vertical lines cannot be represented (when $m \to \infty$). Instead polar coordinates

are adopted, meaning that we represent each line with an angle θ and the perpendicular distance ρ to the origin point. The principle is similar and the equation is

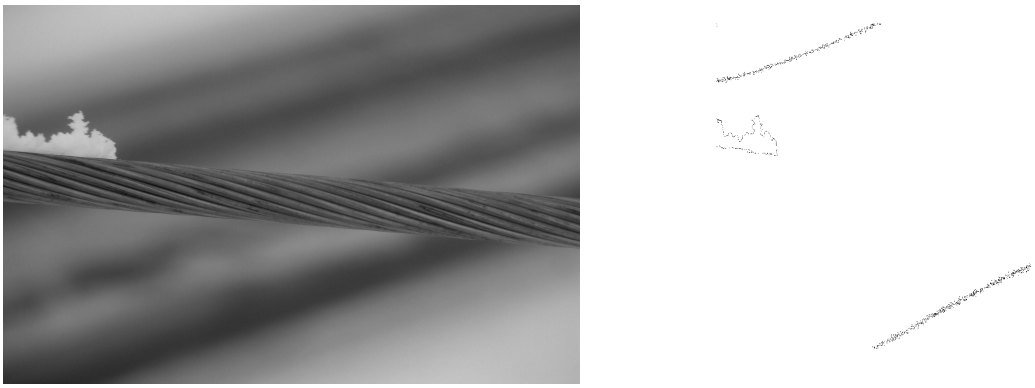
$$\rho = xcos\theta + ysin\theta \tag{3.7}$$

For each point in an image, all the lines going through this point are mapped to a sinusoid in (θ, ρ) space. All the points on the top or bottom line that the cable extends will produce a bunch of sinusoids in crossing a point with certain θ and ρ corresponding to the line that the cable extends. In practice, an accumulator matrix is created, where columns corresponds to a finite set of angles and the rows corresponds to the distance values specified by bins or intervals. The maximum value in this accumulator matrix indicates the total number of pixels associated with the line that has the same angle as the cable extends in the image.

These maximum values will be higher on images such as the one shown in Figure 3.4(b) than the one shown in (a). Another benefit is that if the only lines to be computed have the angles of the lines corresponding to the cable with no ice, then these high Hough values would indicate no ice and all the possible edges corresponding to the car shown in Figure 1.2(b) would be omitted. So the Canny parameter is not that rigid when extracting the Hough-based features because we focus on the θ in one particular direction without much influence of the background objects. These maximum values in the accumulator matrix form the 43th feature extracted from the images.

As some of the ice accumulates on for example, the bottom of the cable, please refer to Figure 1.2(a), edge detection would capture the edges on the top of the cable and the Hough transform would tend to yield higher values. A very difficult case is shown in Figure 3.6(a) where the surface of the cable would yield a high value in the accumulator matrix indicating no ice where in fact there is considerable ice on the upper left part. In order to alleviate these problems, the pixels in the original image with intensity values above 80% of the maximum value were segmented with a simple threshold and then edge detection was performed to eliminate the edges from the cable. The Hough transform would yield a

feature value that corresponds to the ice scenario as the edges after segmentation present almost no lines in the right angle as shown in Figure 3.6(b). This modification then forms the 44th and last feature extracted from the images.



(a) (b)

Figure 3.6: (a) Original image with a high Hough feature value indicating no ice. (b) Edge detection after segmentation. Hough features in this image would reduce the value of pixels with lines corresponding to the cable indicating no ice

3.4 Correlation Coefficients of Features

The correlation coefficient is also known as the product-moment coefficient of correlation or Pearson's correlation. it is a measure of their linear dependence between two random variables. If each variable (also refer to features here) has N scalar observations, then the correlation coefficient is defined as

$$\rho(A,B) = \frac{1}{N-1}\Sigma_{i=1}^{N}(\frac{A_i - \mu_A}{\sigma_A})(\frac{B_i - \mu_B}{\sigma_B}) \tag{3.8}$$

where μ_A and σ_A are the mean and standard deviation of variable A respectively, and μ_B and σ_B are the mean and standard deviation of variable B. Alternatively, the correlation coefficient in terms of the covariance of A and B can be defined as

$$\rho(A,B) = \frac{cov(A,B)}{\sigma_A \sigma_B} \tag{3.9}$$

Here there are 44 features with each feature taken as a random variable, so we got a matrix of correlated coefficient with 44×44 dimension as shown in Figure 3.7. From the Figure, it is seen that the diagonal values are 1s, because variables have the highest correlation with themselves. Besides, some of the GLCM features have high mutual correlation with each other. The dark red points represent high positive correlation between two features whilst the dark blue points indicate high negative correlation. From the matrix of correlated coefficients, redundancy exists among all the 44 features, which suggests that feature processing is essential to achieve the more efficient solution and as well more accurate detection results with some classifiers.

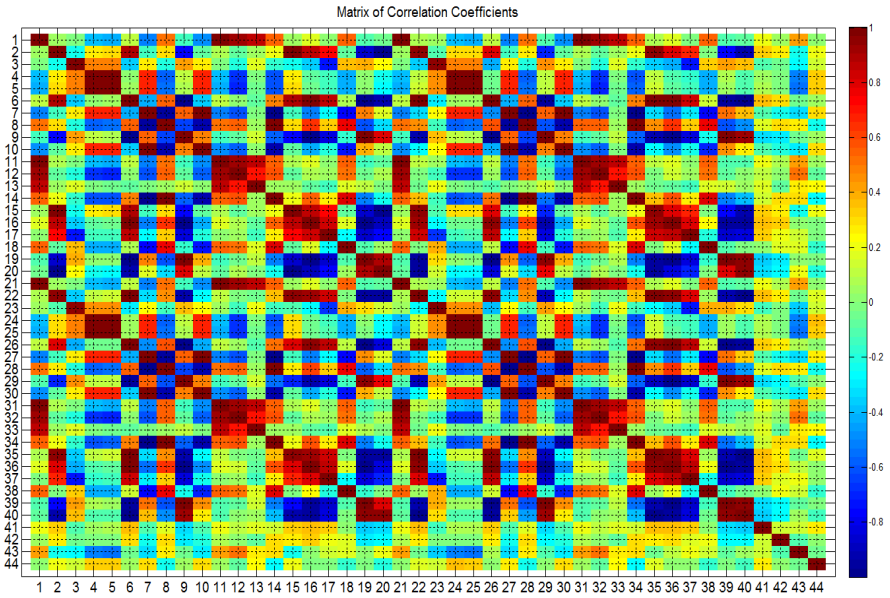


Figure 3.7: Matrix of correlation coefficients

3.5 Summary

A total of 44 features are given in this chapter. The first 40 features are based on the GLCM with 20 different features for each direction. These features are extracted from the gray images in order to obtain more original information. Followed are the two statistical features by computing the central standard deviation and standard deviation of the width changes of cables. The last two features are based on the Hough transform to detect the top

and bottom lines of the cables. Then we analyzed the correlation coefficients matrix of the feature and found that some of the features are highly correlated.

Chapter 4

Feature Processing

Feature selection [37–39] is a popular technique in feature processing. It is known that feature selection includes filter-based and wrapper based methods.

Filters method involves a non-iterative computation on the dataset, which can execute much faster. Besides, since it evaluates the intrinsic properties of the features without interactions with a particular classifier, their results exhibit more generality. However this method tends to select the full feature set as the optimal solution, thus forcing the user to decide a cutoff on the number of features to be selected. The paper in [39] brought up a correlation-based feature selection (CFS). CFS is a filter-based method that seeks features that are highly correlated with the class and uncorrelated with each other. In [40], it was proposed to select features corresponding to the maximal statistical dependency criterion based on mutual information.

Wrappers generally achieve better recognition performance than filter methods as they are tuned to the specific interactions between the classifier and the dataset. With cross-validation technique to measure predictive accuracy, wrapper methods can avoid overfitting. The problem is that this method can become unfeasible for computationally intensive methods and it lacks generality since it biases to the classifier used in the evaluation function. The work in [38] studied the strengths and weaknesses of the wrapper method and indicated that wrapper methods outperformed filter methods. One

wrapper method, Sequential Forward Selection (SFS) is widely used due to its simplicity and efficiency [41, 42]. Reference [41] presents a feature selection method based on SFS and Feed Forward Neural Network (FFNN) to estimate the prediction error as a selection criterion and they achieved superior accuracy with Artificial Metaplasticity on Perceptron Multilayer (AMMLP) than that obtained by conventional BP algorithm and other recent feature selection algorithms. In [42], a comparison was made to evaluate the Sequential Forward Floating Selection (SFFS) and SFS in on-line handwritten whiteboard note recognition.

Another feature reduction method, principal component analysis (PCA) [43] is an alternative feature extraction method to transform the original variables into other variables based on the linearly uncorrelated principal components (orthogonal). The number of principal components is less than or equal to the number of original variables. Different from the feature selection technique mentioned above, feature extraction is to process features via a linear transformation from the original feature space to another feature space. This transformation is defined in the way that the first principal component has the largest possible variance (accounts for as much of the variability in the data as possible), and succeeding components are ranked in turn on the variance. The extracted features are not members from the original features any more. The kernel technique can be applied to perform the nonlinear form of PCA in [44].

4.1 Principal Component Analysis

The PCA technique as a preprocessing step has been used to improve the efficiency for classification of hyperspectral images [45]. One benefit using PCA in [45] was to transform the original data that consisted on highly correlated neighboring bands of hyperspectral images to data based with orthogonal principal components. Although the separability of features may not be maintained perfectly as original features after the PCA transformation, benefits on efficiency and memory cost made it worthwhile to compare the performance.

PCA is based on the eigenvalue decomposition of the covariance matrix, which is symmetric. To measure the variance properly, firstly the data needs to be normalized. A linear combination of the samples X can be written as $t = Xw$, where w is the vector with a dimension of m. Then the problem is to maximize the variance of t as below

$$arg\max_{w}(t^T t) = arg\max_{w}(w^T X^T X w) \qquad \text{s.t} \quad ||w|| = 1 \tag{4.1}$$

The constraint $||w|| = 1$ is equivalent to $w^T w = 1$. And the optimization can be written as

$$arg\min_{w} -\frac{1}{2}(w^T X^T X w) \qquad \text{s.t} \quad w^T w = 1 \tag{4.2}$$

Using Lagrange multipliers to form the Lagrangian formation,

$$L(w) = -\frac{1}{2}(w^T X^T X w) + \frac{1}{2}\lambda(w^T w - 1) \tag{4.3}$$

In linear algebra, we have$\frac{\partial w^T A w}{w} = (A + A^T)w$. So the derivative of Equation (4.3) is

$$\begin{aligned} \frac{\partial L(w)}{w} &= -\frac{1}{2}(w^T X^T X w) + \frac{1}{2}\lambda(w^T w - 1) \\ &= -X^T X w + \lambda w \end{aligned} \tag{4.4}$$

Let Equation (4.4) equal to 0, we obtain

$$-X^T X w + \lambda w = 0 \quad \Rightarrow \quad X^T X w = \lambda w \tag{4.5}$$

Assuming the $X^T X$ is positive definite which means the inverse $(X^T X)^{-1}$ exists, w^* is the eigenvector of the matrix $X^T X$. Substituting Equation (4.5) to (4.1),

$$w^{*T} X^T X w^* = \lambda w^{*T} w^* = \lambda \tag{4.6}$$

Therefore the optimal w for the optimization problem is the eigenvector corresponding to the maximum eigenvalue of the covariance matrix $X^T X$.

Suppose that W is the matrix composed of the m eigenvectors $w_k (k = 1, 2, \cdots, m)$

$$W = (w_1, \cdots, w_k, \cdots, w_m) \tag{4.7}$$

The linear transform defined by

$$t_i = W^T x_i,\ (i = 1, 2, \cdots, n) \tag{4.8}$$

is the PCA vector where n is the number of observations. The set $T = t_1, t_2, \cdots, t_n$ is the transformed data examples.

As mentioned before, 44 features are used in this work with 40 of them obtained from GLCM properties. It was seen that some of the GLCM features are highly correlated. Therefore, when using PCA on the total feature set, some of the eigenvalues can be 0, which means even for the total variation there is no need to use all of the 44 features. Moreover, as indicated in the results section, the first 10 principal components explained 99.09% of the total variability. Therefore, employing PCA can reduce the number of features to less than 10 features without loss of much variability.

4.2 Sequential Forward Selection

The SFS algorithm is a bottom-up search method which starts from an empty feature set χ_0 and adds another feature one by one selected by a certain criterion. Usually the criterion is the Mean Square Error (MSE) for regression models and Misclassified Error (MCE) for classification cases. This process continues until no improvement on the criterion is achieved by adding more features.

Basically, one feature $x^{(1)}$ which yields the lowest classification error $J_0(x^{(i)} = f_i)$ out of the set of whole features $F = f_1, \cdots, f_m$ is added to the empty feature set χ_0 and formed the χ_1.

$$x^{(1)} = arg \min_{f_i \in F} J(f_i) \tag{4.9}$$

Then the feature set χ_1 is augmented recursively by

$$x^{(k+1)} = arg \min_{f_i \in F \setminus \chi_k} J(f_i, \chi_k) \tag{4.10}$$

$$\chi_{k+1} = \chi_k \cup x^{(k+1)} \tag{4.11}$$

where the feature $f_i = x^{(k)}$ is added at each iteration until the error does not decrease or the required number of features are selected.

There is another similar selection method Sequential Backward Selection (SBS). Different from SFS, SBS goes in the opposite way starting from the full set, and sequentially remove the feature that least reduces the value of the objective function (criterion). SBS works best when the optimal feature subset is large, since SBS spends most of time visiting large subsets, which makes it obvious that SBS is slower compared to the SFS. Here we applied SFS because it is more efficient.

4.3 Summary

In this chapter, we mainly present how the two feature processing techniques work. One is the transform based PCA which belongs to the feature extraction category. The other is the SFS which is one of the feature selection algorithms. The two methods have their own advantages and disadvantages depending on the specific application.

Chapter 5

Classification Work

In this chapter we describe the classifiers used. One of the most popular classifiers is the SVC mentioned in Chapter 2. The Naive Bayes classifier assumes that the features are independent. Although this assumption often is difficult to satisfy with real-world data, the Naive Bayes classifier has been proven in practice to be competitive to other more sophisticated classifiers [46].

5.1 Support Vector Classification

Similar to the SVR, we have n training examples denoted by $(\mathbf{x}_i, y_i)$, where $i = 1, \cdots, n$ and d-dimensional input $\mathbf{x} \in R^d$. The class label is one of two values $y \in (-1, 1)$ for a two-category case. In classification formulation, the algorithm tries to find a hyperplane $(\mathbf{w}, b)$ (here $\mathbf{w}$ is a row vector) to maximize the minimum geometric margin for all the training examples. The hard-margin model is given below

$$f(\mathbf{x}, \mathbf{w}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b) \tag{5.1}$$

where $\mathbf{w}$ is a vector orthogonal to the hyperplane, b is the bias and $\cdot$ denotes the dot product. In practice, we usually use the soft-margin model where $f(\mathbf{x}, \mathbf{w}) = \mathbf{w} \cdot \mathbf{x} + b$ and it has a

hinge loss function $L(y_i, f(\mathbf{x}_i, \mathbf{w}))$ as shown in Figure 5.1.

$$L(y_i, f(\mathbf{x}_i, \mathbf{w})) = \max(0, (1 - (\mathbf{x}_i \cdot \mathbf{w} + b)y_i)) \tag{5.2}$$

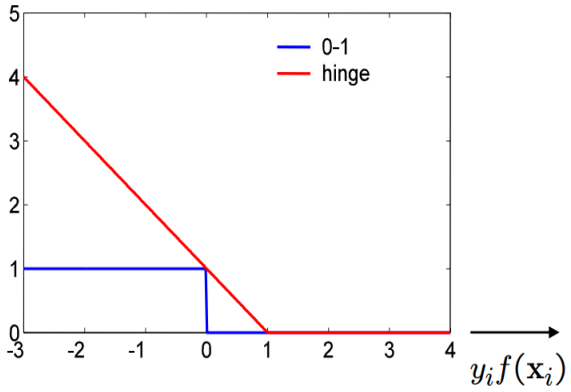


Figure 5.1: Hinge loss function

The goal that maximizes the geometric distance to the closest data points is accomplished by minimizing $||\mathbf{w}||$. So the objective function is

$$\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i, \mathbf{w})) \tag{5.3}$$

which can also be written as

$$\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i \tag{5.4}$$

$$s.t. \begin{cases} (\mathbf{x}_i \cdot \mathbf{w} + b)y_i \geq 1 - \xi_i & \forall i \\ \xi_i \geq 0 & \forall i \end{cases}$$

The constant $C > 0$. When C is large, it penalizes much on the error. So the algorithm will try to separate the classes, which may result in overfitting. When C is small, it penalize less on the error. The algorithm will try best to maximize the margin which may result in underfitting. Here the parameter C is chosen by applying the cross validation technique.

The dual form of the object function with Lagrange multipliers is

$$\max_{\alpha_i \geq 0, \forall i} \min_{\mathbf{w}} L(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \Sigma_{i=1}^{n}\alpha_i[(\mathbf{x}_i \cdot \mathbf{w} + b)y_i - 1] \tag{5.5}$$

where α is the vector of n non-negative Lagrange multipliers. Compute the derivative with respect to **w** and b and then set them to 0. We have

$$\begin{cases} \mathbf{w} = \Sigma_{i=1}^{n_{SV}} \alpha_i y_i \mathbf{x}_i \\ \Sigma_{i=1}^{n_{SV}} \alpha_i y_i = 0 \end{cases} \tag{5.6}$$

where n_{SV} is the number of support vectors.

The vector **w** is a linear combination of the training examples. Substituting Equation (5.6) into (5.5), the final formulation is

$$\max_{\alpha} \Sigma_{i=1}^{n} \alpha_i - \frac{1}{2} \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{5.7}$$

$$s.t. \begin{cases} \Sigma_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C \end{cases}$$

The negative of this maximization form is a Quadratic Programming (QP) minimization problem as below.

$$\min_{\alpha} -\Sigma_{i=1}^{n} \alpha_i + \frac{1}{2} \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} \alpha_i \alpha_j y_i y_j < \mathbf{x}_i, \mathbf{x}_j > \tag{5.8}$$

$$s.t. \begin{cases} \Sigma_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C \end{cases}$$

Fortunately, many techniques have been developed to solve QP problems, such as the Sequential Minimal Optimization (SMO) [47]. The reason why it is not suggestive to solve the problem straightforward is that first the memory consumption for matrix H($H_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$) is $O(n^2)$, increasing quadratically with the number of training examples n, and second the numerical instability resulting in sub-optimal solutions or sometimes no solution is a potential hazard [48].

It can also be shown that

$$\alpha_i (y_i (w \cdot x_i + b) - 1) = 0 \quad \forall i \tag{5.9}$$

which indicates that when the functional distance of a training example is greater than 1, then $\alpha_i = 0$. These training examples that are close to the boundary (functional distance is less than or equals to 1) contribute to **w** and are termed support vectors (please see circled points in Figure 5.2). Those support vectors (SV) which are located on the side of positive examples (blue points) are positive SVs. Likewise the SVs on the side of negative examples (red points) are negative SVs.

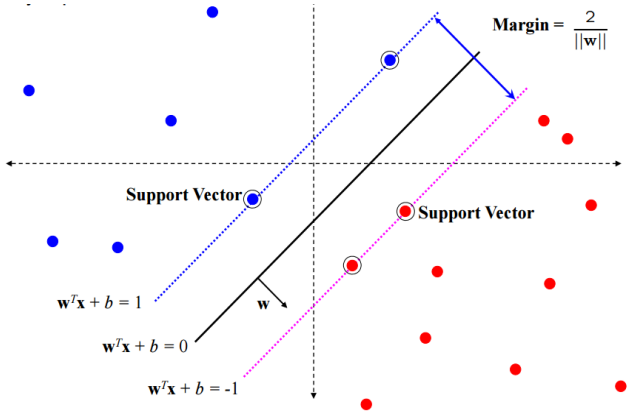


Figure 5.2: Examples of support vectors

Suppose we have the optimal α^*, then we can construct w. For the positive and negative support vectors, x^+ and x^-, we have

$$(w \cdot x^+ + b) = +1$$

$$(w \cdot x^- + b) = -1$$

Solving the equations, we get

$$b = -\frac{1}{2}(w \cdot x^+ + w \cdot x^-) \tag{5.10}$$

For non-linear classification problems, a mapping function $\phi(\mathbf{x})$ is defined that transforms the d dimensional input **x** into a m (usually higher) dimensional vector **z**.

$$\mathbf{z} = \phi(\mathbf{x}) \tag{5.11}$$

The Equation (5.6) would be

$$w = \Sigma_{i=1}^{n_{SV}} \alpha_i y_i \phi(\mathbf{x}_i) \tag{5.12}$$

And Equation (5.1) would be

$$f(\mathbf{x}, \mathbf{w}) = \text{sign}(\mathbf{w} \cdot \phi(\mathbf{x}) + b) \tag{5.13}$$

The final formation for Equation (5.8) is

$$\min_{\alpha} -\Sigma_{i=1}^{n} \alpha_i + \frac{1}{2} \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} \alpha_i \alpha_j y_i y_j (K(\mathbf{x}_i, \mathbf{x}_j)) \tag{5.14}$$

$$s.t. \begin{cases} K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)\phi(\mathbf{x}_j) \\ \Sigma_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C \end{cases}$$

5.2 Naive Bayes Classifier

Naive Bayes classifier has been proven to be effective in many applications such as text classification, medical diagnosis [46]. By assuming that features are independent given class, then $P(X|C) = \prod_{i=1}^{n} P(X_i|C)$, where $X = (X_1, \cdots, X_n)$ is a feature vector and C is a class.

A function $g : \Omega \rightarrow 0, \cdots, K-1$, where $g(\mathbf{x}) = C$ is deterministic without noise, which is the ground truth class. Suppose there are K class in total.

The Bayes classifier uses the class posterior probabilities as discriminant function given the training examples, i.e. $f_k(\mathbf{x}) = P(C = k|\mathbf{X} = \mathbf{x})$, where $k = 1, \cdots, K$. Bayes rule gives

$$P(C = k|\mathbf{X} = \mathbf{x}) = \frac{P(\mathbf{X} = \mathbf{x}|C = k)P(C = k)}{P(\mathbf{X} = \mathbf{x})} \tag{5.15}$$

where $P(\mathbf{X} = \mathbf{x})$ is identical for all classes. So the denominator term can be ignored, which gives

$$f_k(x) = P(\mathbf{X} = \mathbf{x}|C = k)P(C = k) \tag{5.16}$$

where $P(\mathbf{X} = \mathbf{x}|C = k)$ is the class-conditional probability. Then the Bayes classifier finds the maximum a posteriori probability (MAP) hypothesis given examples $\mathbf{x}$ as below

$$h(\mathbf{x}) = \arg\max_k \ P(\mathbf{X} = \mathbf{x}|C = k)P(C = k) \tag{5.17}$$

The direct estimation of $P(\mathbf{X} = \mathbf{x}|C = k)$ is difficult when it is applied to a high dimensional space. By assuming the features are independent given the class, the simplified form yields the naive Bayes classifier defined by

$$f_k^{NB}(\mathbf{x}) = \prod_{j=1}^{n} P(X_j = x_j|C = k)P(C = k) \tag{5.18}$$

where n is the number of training data.

The risk of Naive Bayes classifier is defined as

$$\begin{aligned} R &= P(h(\mathbf{X}) \neq g(\mathbf{X})) \\ &= \sum_{x \in \Omega} P(h(\mathbf{X}) \neq g(\mathbf{X}))P(\mathbf{X} = \mathbf{x}) \\ &= E_{\mathbf{x}} P(h(\mathbf{x}) \neq g(\mathbf{x})) \end{aligned} \tag{5.19}$$

where $E_{\mathbf{x}}$ is the expectation over $\mathbf{x}$.

The Naive Bayes classifier is designed for use when features are independent of one another within each class, but it has been proven that it appears to work well in practice even when it does not satisfy the independence assumption. There are two steps in Naive Bayes classification:

- Training step: It estimates the parameters of a probability distribution based on the given training data, assuming the features are conditionally independent within each class.

- Prediction step: For unseen test data, it computes that posterior probability of that test sample belonging to each class. The method then assign the test data based on the largest posterior probability.

Here, we model the distribution within each class using a multivariate Gaussian

distribution which is commonly used with density:

$$P(\mathbf{X}|C=k) = \frac{1}{\sqrt{(2\pi)^m|\Sigma_k|}} exp(-\frac{1}{2}(X-\mu_k)^T\Sigma_k^{-1}(X-\mu_k)) \tag{5.20}$$

where m is the dimension of features. In the training step, we estimate from the training examples the class priors $P(C=k)$ by the proportion of instances of class k, the class mean μ_k by the empirical sample class means and the covariance matrices Σ_k by the empirical sample class covariance matrices.

5.3 Linear Discriminant Analysis and Quadratic Discriminant Analysis

In section 5.2, we introduced the Naive Bayes classifier which finds the MAP hypothesis given examples by assuming the features are independent, which basically means that we assume the covariance matrices are diagonal. In the case of Linear Discriminant Analysis (LDA), the Gaussians for each class are assumed to share the same covariance matrix: $\Sigma_k = \Sigma$ for all k, which leads to linear decision boundaries. In the case of Quadratic Discriminant Analysis (QDA), there are no assumption on the covariance matrices Σ_k of the Gaussians, yielding to quadratic decision boundaries. So the LDA is a special case of QDA.

In nonlinear discriminant problems, QDA performed better than LDA due to the constant covariance assumption of LDA. Nonetheless, QDA is not always better. As Bayes classifier is based on the minimum risk estimation, so if the actual boundary is indeed linear, QDA tends to be overfitting. At this point, LDA is better than QDA. These posterior probability based classifier are popular because they have closed-form solutions that can be easily computed and do not have hyperparameters to tune. They have been proven to work well in many applications [49, 50].

5.4 Artificial Neural Network

Artificial Neural Network (ANN) is a computational model inspired by the biological neural network in human brain as shown in Figure 5.3. Natural neurons receive signals through synapses located on the dendrites or membrane of the neuron. The neuron is activated and emits a signal through the axon if the received signals are strong enough (surpass a certain threshold). This signal might be passed to another synapse and go on.

ANN is a highly abstracted model based on the biological architecture above constructed by layers of artificial neurons (nodes). Basically, it consists of inputs (like synapses), which are multiplied by weights (strength of the signals) and then computed by a activation function before being accepted by next neuron, intermediate layers which are called hidder layers, and an output layer which renders an estimated label in classification case. Figure 5.4 gives an ANN architecture with two hidden layers, where w_{ij}, w_{jk} and w_{kl} are weights for different layers, a_j, a_k and a_l are the inputs for the two hidden layers and the output layer, z_j, z_k and z_l are outputs for the three layers, and $\theta = [w_{ij}, w_{jk}, w_{kl}]$. In the learning stage, the algorithm adjusts the weights of the ANN to obtain the desired output from the network. One of the most common models for learning the appropriate weights is the backpropagation algorithm [51]. The artificial neurons send their signals forward, and then errors (difference between ground-truth and estimated results) are propagated backwards. The idea of the backpropagation algorithm is to reduce the final error by adjusting the weights which are initialized randomly.

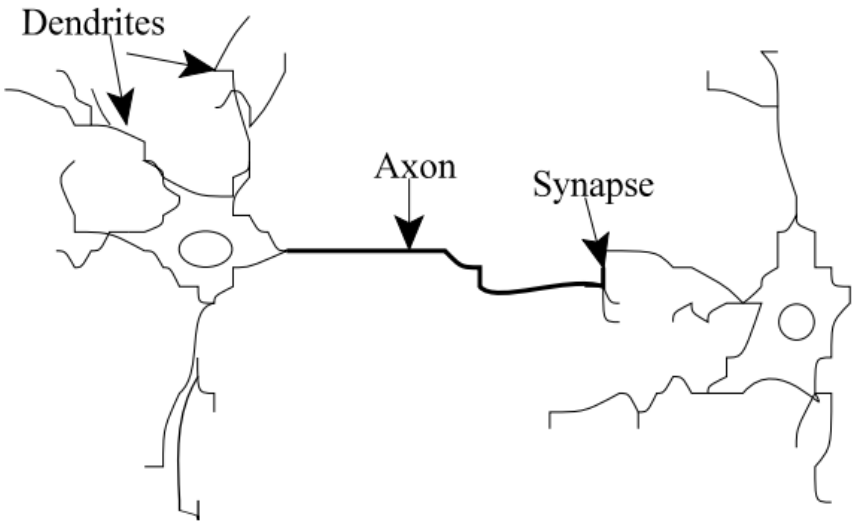


Figure 5.3: Natural neural architecture

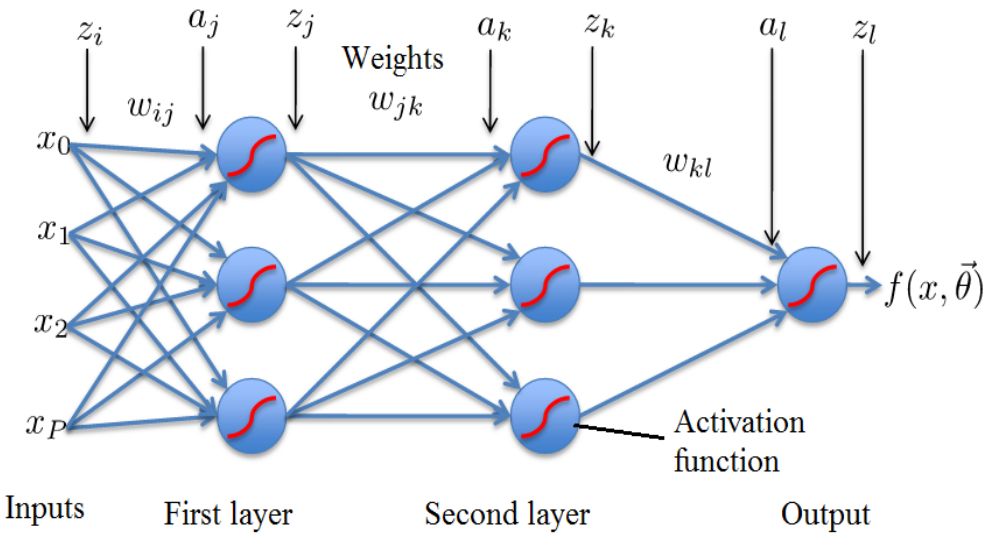


Figure 5.4: Two layer's Neural Network architecture

The most common activation function is the sigmoid function, which is close to 1 for large positive numbers, 0.5 at zero, and close to 0 for large negative numbers given below

$$g(t) = \frac{1}{1 + e^{-t}} \tag{5.21}$$

This function is introduced to conduct a non-linear model as shown in Figure 5.5.

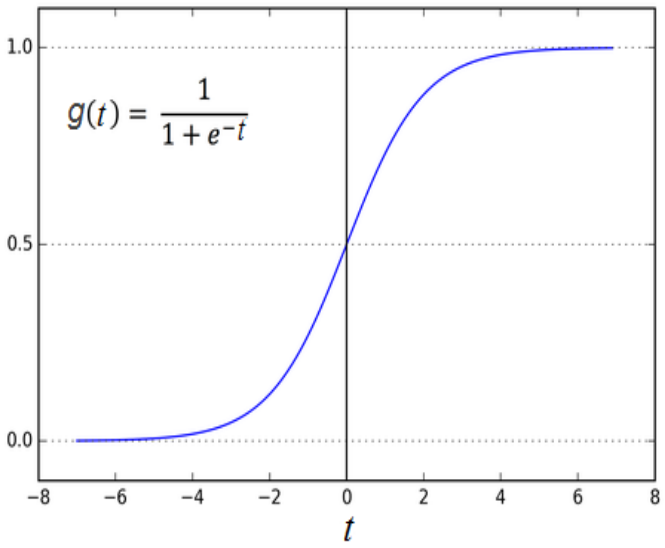


Figure 5.5: Sigmoid function

From Figure 5.4, we have

$$a_j = \sum_i w_{ij} z_i \qquad a_k = \sum_j w_{jk} z_j \qquad a_l = \sum_k w_{kl} z_k \tag{5.22}$$

$$z_j = g(a_j) \qquad z_k = g(a_k) \qquad z_l = g(a_l) \tag{5.23}$$

As mentioned above, the error is the difference between the ground-truth and the estimated output, and we need to adjust the weights in order to achieve the minimum. The output error function can be defined with the least square form and be given as

$$\begin{aligned} R(\theta) &= \frac{1}{N} \sum_{n=0}^{N} L(y_n - f(x_n)) \\ &= \frac{1}{N} \sum_{n=0}^{N} \frac{1}{2} (y_n - f(x_n))^2 \\ &= \frac{1}{N} \sum_{n=0}^{N} \frac{1}{2} (y_n - g(\sum_{k}^{K} w_{kl} g(\sum_{j}^{M} w_{jk} g(\sum_{i}^{P} w_{ij} x_{n,i}))))^2 \end{aligned} \tag{5.24}$$

where N is the number of training examples, P is the number of descriptors, M and K are the number of nodes in the two hidden layers.

For $f(x, y)$, with f differentiable with respect to x and y, and x and y differentiable

with respect to u and v, we have

$$\frac{\partial f}{\partial u} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial u} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial u} \tag{5.25}$$

and

$$\frac{\partial f}{\partial v} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial v} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial v} \tag{5.26}$$

First we initialize the parameter θ (w_{ij}, w_{jk} and w_{kl}) and feed forwards the inputs to the outputs, so that we obtain the current a_j, a_k, a_l, z_j, z_k and z_l. The Backpropagation approach takes the gradient of the last component and iterates backwards. Optimizing the weights w_{kl} in the output layer, we have

$$\begin{aligned}
\frac{\partial R}{\partial w_{kl}} &= \frac{1}{N}\sum_{n}^{N}[\frac{\partial L_n}{\partial a_{l,n}}][\frac{\partial a_{l,n}}{\partial w_{kl}}] \\
&= \frac{1}{N}\sum_{n}^{N}[\frac{\partial \frac{1}{2}(y_n - g(a_{l,n}))^2}{\partial a_{l,n}}][\frac{\partial z_{k,n}w_{kl}}{\partial w_{kl}}] \\
&= \frac{1}{N}\sum_{n=1}^{N}[-(y_n - z_{l,n})g^{'}(a_{l,n})]z_{k,n} \\
&= \frac{1}{N}\sum_{n=1}^{N}\delta_{l,n}z_{k,n}
\end{aligned} \tag{5.27}$$

where $\delta_{l,n} = -(y_n - z_{l,n})g^{'}(a_{l,n})$. Then we optimize the weights w_{jk} in the last hidden layer.

$$\begin{aligned}
\frac{\partial R}{\partial w_{jk}} &= \frac{1}{N}\sum_{n}^{N}[\frac{\partial L_n}{\partial a_{k,n}}][\frac{\partial a_{k,n}}{\partial w_{jk}}] \\
&= \frac{1}{N}\sum_{n}^{N}[\sum_{l}^{M}\frac{\partial L_n}{\partial a_{l,n}}\frac{\partial a_{l,n}}{\partial a_{k,n}}][\frac{\partial a_{k,n}}{\partial w_{jk}}] \\
&= \frac{1}{N}\sum_{n}^{N}[\sum_{l}^{M}\delta_l\frac{\partial a_{l,n}}{\partial a_{k,n}}][z_{j,n}] \\
&= \frac{1}{N}\sum_{n}^{N}[\sum_{l}^{M}\delta_l w_{kl}g^{'}(a_{k,n})][z_{j,n}] \\
&= \frac{1}{N}\sum_{n}^{N}[\delta_{k,n}][z_{j,n}]
\end{aligned} \tag{5.28}$$

where $\delta_{k,n} = \sum_l^M \delta_l w_{kl} g^{'}(a_{k,n})$. Similarly, we have

$$\frac{\partial R}{\partial w_{ij}} = \frac{1}{N} \sum_n^N [\delta_{j,n}][z_{i,n}] \tag{5.29}$$

where $\delta_{j,n} = \sum_k^K \delta_k w_{jk} g^{'}(a_{j,n})$.

Now that we have defined gradients for each parameter, update the weights using gradient descent methodology.

$$\begin{aligned} w_{ij}^{t+1} &= w_{ij}^t - \eta \frac{\partial R}{\partial w_{ij}} \\ w_{jk}^{t+1} &= w_{jk}^t - \eta \frac{\partial R}{\partial w_{jk}} \\ w_{kl}^{t+1} &= w_{kl}^t - \eta \frac{\partial R}{\partial w_{kl}} \end{aligned} \tag{5.30}$$

where η is the learning rate. Other approaches such as scaled conjugate gradient backpropagation [52], Levenberg-Marquardt backprogation [53] are also applied in practice.

With the final θ when the error function converges, it forms a trained model which can be used to make a prediction with new inputs. Note that the gradient descent method can only find the local minima. With random initialization, it can achieve the global minimum and local minima as well.

5.5 Summary

Different classifiers are discussed in this chapter. First is the SVC. The basic theory is similar to the SVR, but the objective function is different and the output is the label which the example should be assigned to. Then we talk about the probability theory of Naive Bayes classifier. The objective is to find the MAP hypothesis. After that we briefly compare how LDA and QDA differ from Naive Bayes classifier. The three classifiers applied here are all based on the Gaussian distribution model. The difference is that the covariance matrix for each classifier is on different assumption. At last some basic knowledge about the gradient descent backpropogation for Neural Network classification was described.

Chapter 6

Results and Analysis

6.1 Support Vector Regression for Canny Threshold Selection

SVR was conducted based on the LIBSVM [23]. 49 images were used for the training of the Canny upper thresholds and the ground-truth threshold values were formed by empirically examination via viewing the edge results by the user. 5-fold cross validation was applied to determine the SVR parameters C, γ and ε. From Table 6.1, the optimal C used was 8 and γ was 0.25 when setting $\varepsilon = 0.01$. The minimum error was 0.00375. When setting ε as 0.001, 0.1 or 1, the resulting parameter values corresponding to minimum errors are displayed in Table 6.1. Based on the results, the optimal values to achieve best SVR performance are $\varepsilon = 0.01$, $C = 8$ and $\gamma = 0.25$.

Once we have the optimal parameters for SVR, the next step is to train the 49 images and make a prediction on other testing images. The testing images are a set of 612 observations with 312 icy and 300 non-ice images. Need to mention that among the 612 images, 83 of them were misclassified by the previous detection algorithm used by Manitoba Hydro. After obtaining the predicted parameters for the 612 images, classifications were done to evaluate the performance on Matlab R2014a.

In this work we call original features to the features used with the empirical method

in Chapter 2.2. With QDA and LDA classifiers, the misclassified errors are 0.0180 and 0.0392 respectively based on the original features to determine the Canny upper threshold. The results are the average errors based on the 10-fold cross validation using the 40 GLCM features [54] from the original images and the 4 features that include the statistical features and Hough transform from the edge images. With the SVR technique, the errors were reduced to 0.0114 and 0.0376, which offers an improvement.

Table 6.1: 5-fold cross validation results with different ε values

ε	0.001	0.01	0.1	1
C	2	8	0.5	0.0313
γ	4	0.25	0.25	0.000977
Error	0.00443	0.00375	0.00721	0.04169

6.2 Principal Component Analysis for Classification Results

Experiments were conducted on the set of 612 images. It shows that the first 10 principal components account for around 99.1% of the total variations. As the first 3 principal components explained 81.42% of the whole variability transformed from original features (81.56% for features obtained by SVR), the scores corresponding to each feature were plotted to explore some latent information as shown in Figure 6.1. Most of the variables have positive and negative coefficients along each principal component. Intuitively, these principal components can distinguish images that have high values for the positive set of variables and low values for the negative set of variables and images that have the opposite case.

Based on the QDA and LDA classifiers, Figure 6.2 shows the classification errors changing from the first one principal component to all components. It is seen that the classification error with QDA outperformed that with LDA and the error nearly does not decrease any more after the number of features reaches 10. The classification error with

the first 10 principal components is 0.01961, which is close to the 0.0180 without PCA mentioned in the paragraph before Table 6.1.

Figure 6.3 shows the results with PCA when applying SVR to predict the Canny thresholds. For the QDA, the classification error keeps almost stable when the number of principal components equals to 8 where the error is 0.02288. The error decreases to 0.01797 with 10 first principal components which is close to the 0.0114 without PCA.

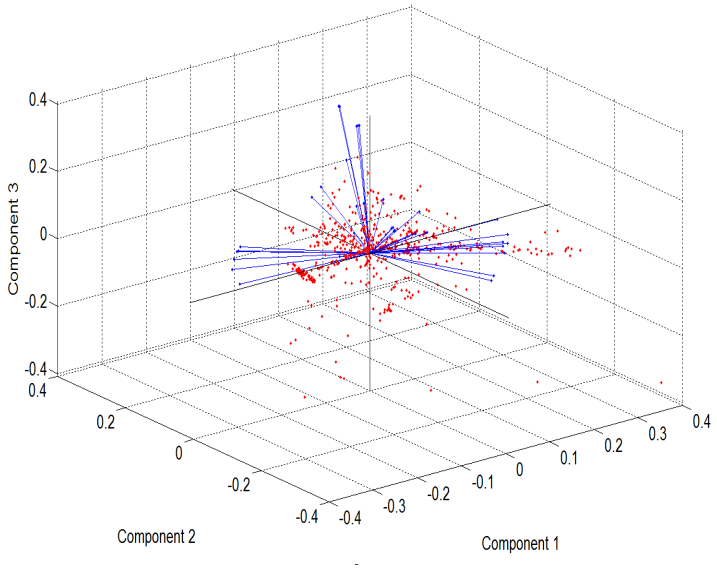


Figure 6.1: Visualization of the first 3 components

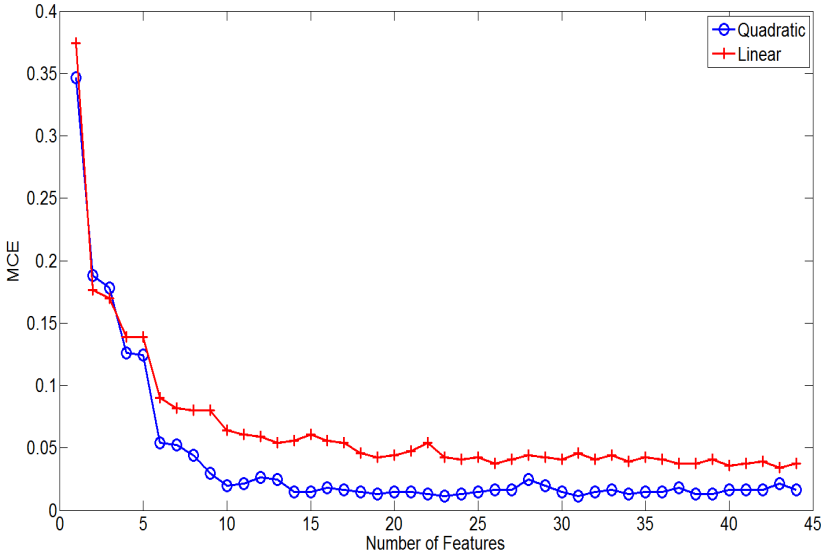


Figure 6.2: Classification errors with PCA (original features)

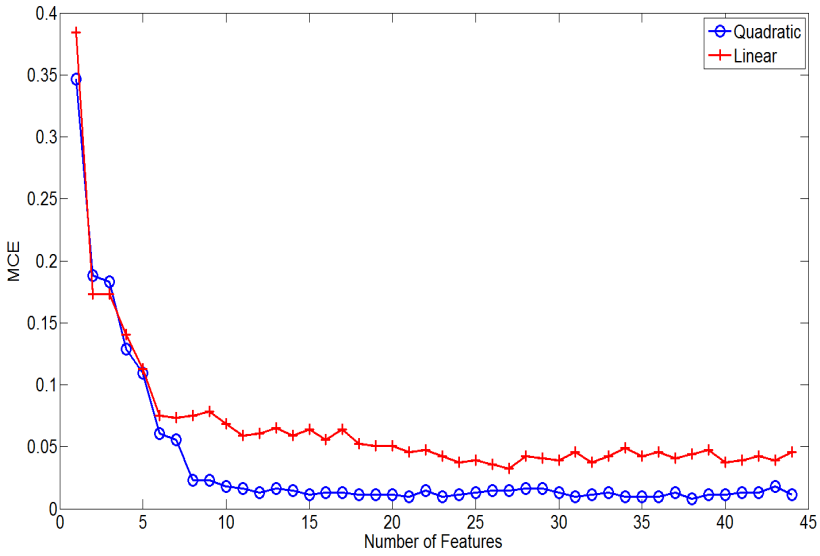


Figure 6.3: Classification errors with PCA (features obtained by SVR)

6.3 Sequential Forward Selection for Classification Results

The SFS for classification results are given in Figure 6.4 which shows the relationship between the classification errors and the number of features selected. The results with QDA achieved better results than LDA and the classification errors do not reduce when 10 features were selected with an error of 0.01144.

Figure 6.5 shows the results with SFS obtained with Canny upper thresholds learned by SVR. With the first 8 features, the classification error is 0.009804, which achieved the best classification accuracy.

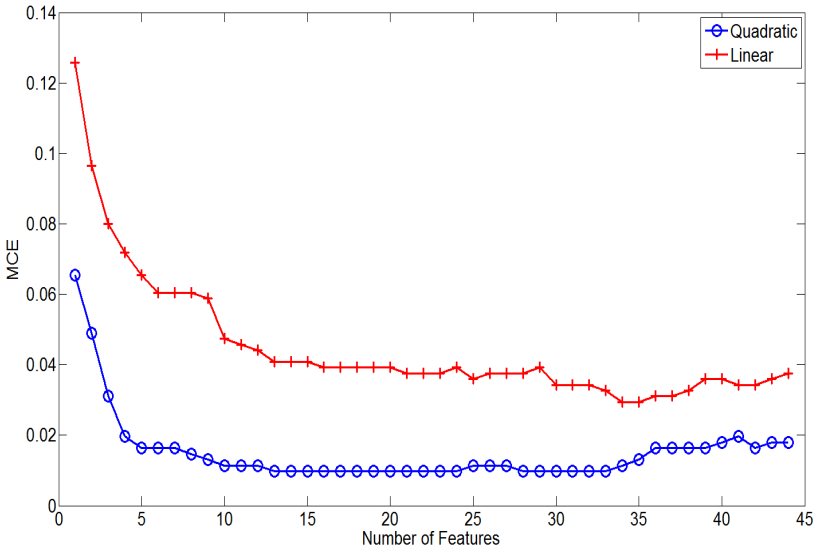


Figure 6.4: Classification errors with SFS (original features)

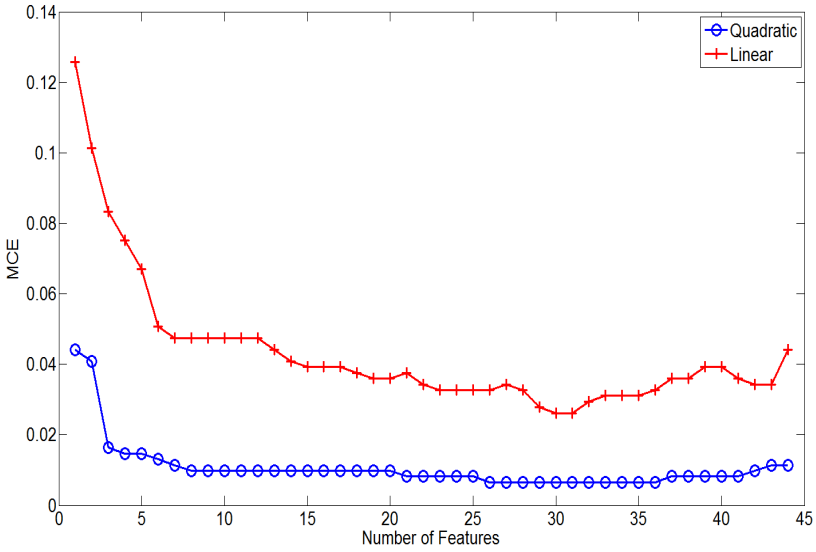


Figure 6.5: Classification errors with SFS (features obtained by SVR)

6.4 Classification Results with Support Vector Classification and Bayes Classifier

From the above results, it is seen that the QDA classification results with the first 10 principal components for original features (0.01961) and features obtained by SVR

(0.01797) are both close to the results without PCA (0.0180 and 0.0114 respectively). With SFS, the classification errors are 0.01144 and 0.009804 with first 10 and 8 features accordingly. We can see that using features selected by SFS leads to lower classification errors than the results using all the 44 features. But as mentioned in Chapter 4, wrapper method bias to the classifier used in the criterion function, which usually achieves over optimistic estimated accuracy. Hence it is essential to test the selected features on other more classifiers. Next, comparisons based on the other two classifiers are discussed.

For SVC, the L1 loss SVC with Gaussian kernel from LIBSVM was applied. Among the 612 samples, 75% were used for training and the last 25% were used for testing. Again 10-fold cross validation was performed on the training set to find the optimal C and γ that yield the highest accuracy for validation. From Table 6.2, it is evident that features obtained by SVR outperformed original features and the corresponding results with PCA and SFS both boost the classification accuracy. Also one thing that can be seen is that SVC is not much affected by the mutual correlation among the features.

The order of selected features with SVR is 42, 2, 22, 17, 1, 21, 37, 23. These 8 features correspond to width change feature, GLCM contrasts [25], autocorrelation [31] and the first information measure of correlation [25] for both directions, and correlation [25] with offset = [1 3]. The 42th feature width change faeture dominates among all the 44 features. Note that features that ordered at 2 and 22, 17 and 37, 1 and 21 are pairwise correlated. Since the 2nd feature is important, the 22th feature will be selected accordingly. The other two pair groups are in the same case. Some algorithms such as SVC can assign the weight for each feature in the learning process, so the correlation does not affect much the results. From the order of the selected features, given that the 2nd and 22th features are correlated, the classification error decreased when adding the 22th feature. Thinking from other way, the 2nd and 22th features are the same GLCM feature but from different directions and can indicate different locations ice accumulated. Therefore, keeping all the 8 selected features for this detection work is reasonable. Among the 612 images, there are as many as 18 false positive and 65 false negative errors from the previous detection system in [13]. With 25%

of the 612 samples for testing in this solution, the best result is 0.0065 which means among the 153 testing samples there is nearly only 1 image that was misclassified.

For the Naive Bayes classifier, as it relies on the assumption of Gaussian distribution and that features are independent, it may give bad results if the assumption is violated. Still the results with SFS under Naive Bayes classifier are not bad because the selected features may not have such high correlation. Although the transformed features are independent after performing PCA since the bases in the new feature space are orthogonal, transformation may change the distribution of the original data, which may lead to violation of the Bayes assumption, so the result based on PCA for dimension reduction deteriorated with the Bayes-based classifier.

Table 6.2: Comparisons on errors with SVC and Naive Bayes

		PCA	SFS	All features
SVC	Original features	0.0131	0.0131	0.0196
SVC	Features by SVR	0.0065	0.0065	0.0065
Naive Bayes	Original features	0.0850	0.0261	0.0654
Naive Bayes	Features by SVR	0.0980	0.0196	0.0588

6.5 Results with Neural Network

In this section, we present the classification results with Neural Network. As mentioned above, for SVC we need to perform the 10-fold cross validation for the training set to choose the optimal parameters C and γ. For Neural Network, it is like a black box and no parameters need to tune. To achieve the global minimum, we initialize the θ randomly to avoid potential local minima. Similar to the data partition in SVC and Naive Bayes classification, 60% of the 612 images are set for training and 15% are used for validation in case of overfitting when the epochs go too high. The last 25% are left for testing which are the same testing samples as in SVC and Naive Bayes classification. The number of hidden layers is 25. Three different inputs with all features, 8 features selected from SFS and first 10 features transformed from PCA are compared. They are all obtained with the

Canny parameter predicted by SVR, and the latter two use SFS and PCA techniques on the 44 features to reduce the number of features respectively. The optimization algorithm is the scaled conjugate gradient backpropagation by default.

The Mean Squared Error (MSE) versus epoch with best results are displayed in Figure 6.6. From the figure, it is seen that the final MSE with features obtained from three different inputs are quite close. The MSE differ significantly at first but will converge as the epoch goes high. The iterations stop at certain point where the validation error fails to decrease.

We can also compute the classification errors for the testing samples. As the result is different at each run because of random initialization, we just record the best result after many trials. The classification errors on ANN are 0.0131, 0.0131, 0.0196 for all features, features selected by SFS, features transformed by PCA respectively. Therefore, it is seen that the two feature processing techniques are effective and the reduced features can perform as well as all the 44 features in the classification.

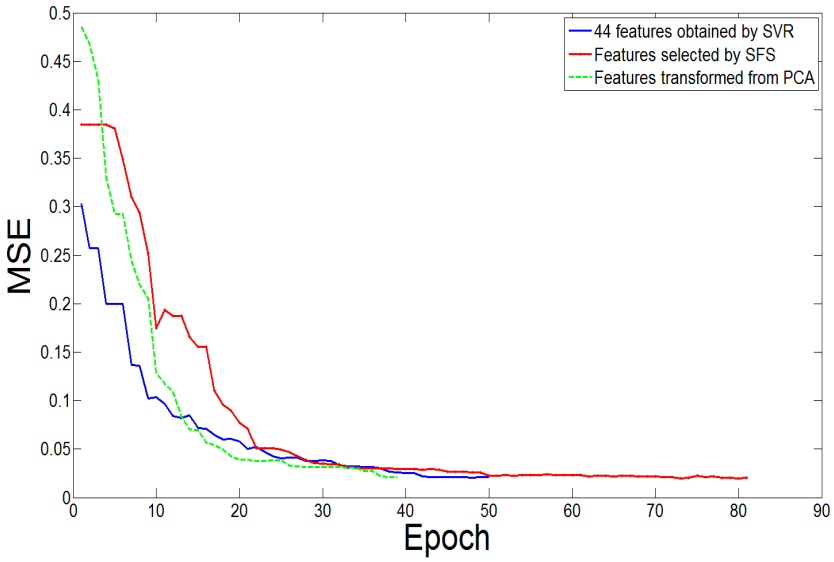


Figure 6.6: Neural Network results using three different inputs

6.6 Summary

In this chapter, we first showed the parameter selection results for SVR using 5-fold cross validation. Then classification performance with PCA and SFS based on LDA and QDA classifiers were presented. Afterwards, two more classifiers SVC and Naive classifier were further tested to evaluate the generalized performance. Finally, we gave the results based on the Neural Network for inputs with all features and reduced features.

Chapter 7

Conclusions and Future Works

7.1 Conclusion and Comments

In this thesis, a solution to detect the accumulated ice on the power lines using computer vision was developed and comparisons were made to evaluate the performance on different classifiers. Compared to the previous system used by Manitoba Hydro which extracted topological features from binary images such as perimeter, area and compactness factor, this new solution combines different aspects of features including textural features based on GLCM, statistical features and features based on Hough transform. The benefits of extracting features from edge images were explained in Chapter 2. Nevertheless, when detecting the edges from the images, the most common problem was to select the parameters which determine a threshold used to extract the edge information. Here we adopted a machine learning method SVR to predict the upper threshold for the Canny edge detector. Experiments showed that the classification accuracy was improved with SVR technique to determine the Canny parameter compared to the results using the empirical method.

As mentioned previously, the first 40 GLCM features were derived from gray images instead of binary images as we wanted to obtain more original information. Especially, we focused on the textural and shape features. The standard deviation of the width changes

of the cables and the features based on Hough transform detected the shape discrepancy between icy and non-ice images. When computing the correlation coefficients of the 44 features, it was obvious that there exists redundancy among the features. At the same time, reducing the dimension of features enhance the computational efficiency. So we considered reducing the feature dimension. The results in Chapter 6 indicated that SFS performed better than PCA, especially with the Bayes classifiers. The reason is that PCA transforms the features from the original space to a new feature space which may alter the data distribution. The Bayes classifier applied here was based on the assumption that the features conform to multivariate Gaussian distribution. Moreover, for SVC, the results obtained from PCA and SFS behave almost the same. From the final 8 selected features by SFS, we could see that some selected GLCM features were still correlated. In Neural Network classification, the final MSE for best results were very close for inputs with all features and features obtained from SFS and PCA, which means that the reduced features work well in the classification compared to all the 44 features.

To conclude, first, the SVR technique was applied to predict the Canny parameters and it has been proven that the classification results based on this methodology performed better than the empirical method. Second, the SFS outperformed PCA with QDA, LDA and Naive Bayes classifier, but they achieved parallel performance with SVC. They achieved similar final MSE in the Neural Network classification. The best performance we achieved was only one error among the 153 testing images with SVC.

7.2 Future Works

As mentioned in Chapter 1, the system is performed on the images which are derived every 15 minutes. Most of the images are in the form of a sequence. The stream information is not taken advantaged of in this thesis. In future work, based on the requirement that how soon it is demanded to deliver a message whether the ice starts to accumulate or not, the classification results obtained from the later images can enhance or confirm the results

judging from the former images.

Furthermore, more other textural and shape-based features can be considered apart from the three kinds of features adopted in this thesis. Color information is not much emphasized here because the background is complicated and the illumination can change in the cable areas.

As indicated in [31] which analysed the texture of synthetic aperture radar (SAR) sea ice imagery with GLCM, the best GLCM implementation in representing sea ice texture is to utilize a range of displacement values so that both microtextures and macrotextures of sea ice can be captured adequately. We may consider applying different displacements of GLCM and combining them together as features, which may further boost the performance.

References

[1] J. Hanna and S. Gallman, "Tens of thousands in U.S., Canada without power days after ice storm," http://www.cnn.com/2013/12/25/us/winter-weather/, 2013, [Online; accessed December-2014].

[2] A. T. DeGaetano, B. N. Belcher, and P. L. Spier, "Short-term ice accretion forecasts for electric utilities using the weather research and forecasting model and a modified precipitation-type algorithm," *Weather and Forecasting*, vol. 23, no. 5, pp. 838–853, 2008.

[3] . U.-C. P. S. O. T. Force, *Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations*, 2004.

[4] F. Kiessling, P. Nefzger, J. F. Nolasco, and U. Kaintzyk, *Overhead power lines: planning, design, construction*. Springer, 2014.

[5] S. A. Changnon and T. G. Creech, "Sources of data on freezing rain and resulting damages," *Journal of Applied Meteorology*, vol. 42, no. 10, pp. 1514–1518, 2003.

[6] E. L. Lecomte, A. W. Pang, and J. W. Russell, *Ice storm'98*. Institute for Catastrophic Loss Reduction Ottawa,, Canada, 1998.

[7] A. Zarnani, P. Musilek, X. Shi, X. Ke, H. He, and R. Greiner, "Learning to predict ice accretion on electric power lines," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 3, pp. 609–617, 2012.

[8] P. McComber, J. Druez, and J. Laflamme, "A comparison of selected models for estimating cable icing," *Atmospheric research*, vol. 36, no. 3, pp. 207–220, 1995.

[9] P. McComber, J. Druez, J. De Lafontaine, A. Paradis, J. Laflamme *et al.*, "Estimation of transmission line icing at different sites using a neural network," in *The Ninth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers, 1999.

[10] T. G. Myers and J. P. Charpin, "A mathematical model for atmospheric ice accretion and water flow on a cold surface," *International Journal of Heat and Mass Transfer*, vol. 47, no. 25, pp. 5483–5500, 2004.

[11] M. Akhloufi and N. Benmesbah, "Outdoor ice accretion estimation of wind turbine blades using computer vision," in *Computer and Robot Vision (CRV), 2014 Canadian Conference on*. IEEE, 2014, pp. 246–253.

[12] C. Yu and Q. Peng, "Icing detection using image-based 3d shape recovery," *Computer-Aided Design and Applications*, vol. 7, no. 3, pp. 335–347, 2010.

[13] R. Wachal, J.-S. Stoezel, M. Peckover, and D. Godkin, "A computer vision early-warning ice detection system for the smart grid," in *Transmission and Distribution Conference and Exposition (T&D), 2012 IEEE PES*. IEEE, 2012, pp. 1–6.

[14] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 679–698, 1986.

[15] M. Sharifi, M. Fathy, and M. T. Mahmoudi, "A classified and comparative study of edge detection algorithms," in *Information Technology: Coding and Computing, 2002. Proceedings. International Conference on*. IEEE, 2002, pp. 117–120.

[16] G. Shrivakshan and C. Chandrasekar, "A comparison of various edge detection techniques used in image processing," *IJCSI International Journal of Computer Science Issues*, vol. 9, no. 5, pp. 272–276, 2012.

[17] B. Ramamurthy and K. Chandran, "Content based image retrieval for medical images using canny edge detection algorithm," *International Journal of Computer Applications*, vol. 17, no. 6, 2011.

[18] H.-Y. Cheng, C.-C. Weng, and Y.-Y. Chen, "Vehicle detection in aerial surveillance using dynamic bayesian networks," *Image Processing, IEEE Transactions on*, vol. 21, no. 4, pp. 2152–2159, 2012.

[19] S. R. Gunn *et al.*, "Support vector machines for classification and regression," *ISIS technical report*, vol. 14, 1998.

[20] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Information Processing-Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

[21] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[22] R. M. Rifkin, "Everything old is new again: a fresh look at historical approaches in machine learning," Ph.D. dissertation, MaSSachuSettS InStitute of Technology, 2002.

[23] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[24] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," 2003.

[25] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *Systems, Man and Cybernetics, IEEE Transactions on*, no. 6, pp. 610–621, 1973.

[26] M. Partio, B. Cramariuc, M. Gabbouj, and A. Visa, "Rock texture retrieval using gray level co-occurrence matrix," in *Proc. of 5th Nordic Signal Processing Symposium*, vol. 75. Citeseer, 2002.

[27] P. Mohanaiah, P. Sathyanarayana, and L. GuruKumar, "Image texture feature extraction using glcm approach," *International Journal of Scientific and Research Publications*, vol. 3, no. 5, p. 1, 2013.

[28] F. R. de Siqueira, W. R. Schwartz, and H. Pedrini, "Multi-scale gray level co-occurrence matrices for texture description," *Neurocomputing*, vol. 120, pp. 336–345, 2013.

[29] N. Tripathi and S. Panda, "A review on textural features based computer aided diagnostic system for mammogram classification using glcm & rbfnn," *Int. J. Eng. Trends Technol*, vol. 17, no. 9, pp. 462–464, 2014.

[30] S. Beura, B. Majhi, and R. Dash, "Mammogram classification using two dimensional discrete wavelet transform and gray-level co-occurrence matrix for detection of breast cancer," *Neurocomputing*, vol. 154, pp. 1–14, 2015.

[31] L.-K. Soh and C. Tsatsoulis, "Texture analysis of sar sea ice imagery using gray level co-occurrence matrices," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 37, no. 2, pp. 780–795, 1999.

[32] D. A. Clausi, "An analysis of co-occurrence texture statistics as a function of grey level quantization," *Canadian Journal of remote sensing*, vol. 28, no. 1, pp. 45–62, 2002.

[33] J. S. Weszka, C. R. Dyer, and A. Rosenfeld, "A comparative study of texture measures for terrain classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 4, no. SMC-6, pp. 269–285, 1976.

[34] C. Coburn and A. C. Roberts, "A multiscale texture analysis procedure for improved forest stand classification," *International journal of remote sensing*, vol. 25, no. 20, pp. 4287–4308, 2004.

[35] C. Barata, M. Ruela, M. Francisco, T. Mendonça, and J. S. Marques, "Two systems for the detection of melanomas in dermoscopy images using texture and color features," *Systems Journal, IEEE*, vol. 8, no. 3, pp. 965–979, 2014.

[36] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[37] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial intelligence*, vol. 97, no. 1, pp. 245–271, 1997.

[38] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1, pp. 273–324, 1997.

[39] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, The University of Waikato, 1999.

[40] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1226–1238, 2005.

[41] A. Marcano-Cedeño, J. Quintanilla-Domínguez, M. Cortina-Januchs, and D. Andina, "Feature selection using sequential forward selection and classification applying artificial metaplasticity neural network," in *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2010, pp. 2845–2850.

[42] J. Schenk, M. Kaiser, and G. Rigoll, "Selecting features in on-line handwritten whiteboard note recognition: Sfs or sffs?" in *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*. IEEE, 2009, pp. 1251–1254.

[43] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[44] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Artificial Neural NetworksICANN'97*. Springer, 1997, pp. 583–588.

[45] C. Rodarmel and J. Shan, "Principal component analysis for hyperspectral image classification," *Surveying and Land Information Science*, vol. 62, no. 2, p. 115, 2002.

[46] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22. IBM New York, 2001, pp. 41–46.

[47] J. Platt *et al.*, "Sequential minimal optimization: A fast algorithm for training support vector machines," 1998.

[48] D. Boswell, "Introduction to support vector machines," 2002.

[49] A. Starzacher and B. Rinner, "Evaluating knn, lda and qda classification for embedded online feature fusion," in *Intelligent Sensors, Sensor Networks and Information Processing, 2008. ISSNIP 2008. International Conference on*. IEEE, 2008, pp. 85–90.

[50] S. Bhattacharyya, A. Khasnobish, S. Chatterjee, A. Konar, and D. Tibarewala, "Performance analysis of lda, qda and knn algorithms in left-right limb movement classification from eeg data," in *Systems in Medicine and Biology (ICSMB), 2010 International Conference on*. IEEE, 2010, pp. 126–131.

[51] D. E. Rumelhart, J. L. McClelland, P. R. Group *et al.*, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1-2," *Cambridge, MA*, 1986.

[52] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural networks*, vol. 6, no. 4, pp. 525–533, 1993.

[53] A. Reynaldi, S. Lukas, and H. Margaretha, "Backpropagation and levenberg-marquardt algorithm for training finite element neural network," in *Computer Modeling and Simulation (EMS), 2012 Sixth UKSim/AMSS European Symposium on*, Nov 2012, pp. 89–94.

[54] A. Uppuluri, "GLCM texture features," http://www.mathworks.com/matlabcentral/fileexchange/22187-glcm-texture-features, 2008, mATLAB Central File Exchange. [Online; accessed 12-January-2016].

Publication List

[1] **B. Li**, G. Thomas and D. Williams, ”Detection of Ice on Power Cables Based on Image Texture Features,” *IEEE International Instrumentation and Measurement Technology Conference*, 2016, industry paper, accept.

[2] **B. Li**, G. Thomas and D. Williams, ”Ice Detection on Electrical Power Cables,” *Advances in Visual Computing*, Springer International Publishing, pp. 355-364, Dec. 2015.