MULTICLASS SUPPORT VECTOR MACHINES FOR DIABETIC RETINOPATHY DIAGNOSIS

JIGME NAMGYAL

A Thesis Submitted in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Applied Mathematics

Suranaree University of Technology

Academic Year 2021

ซัพพอร์ตเวกเตอร์แมชชีนเพื่อการวินิจฉัยโรคจอตาเหตุเบาหวาน

นายจิกมี นัมง์ยาล

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาคณิตศาสตร์ประยุกต์

มหาวิทยาลัยเทคโนโลยีสุรนารี

ปีการศึกษา 2564

MULTICLASS SUPPORT VECTOR MACHINES FOR DIABETIC RETINOPATHY DIAGNOSIS

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree.

Thesis Examining Committee

J. Tanthanuch

(Asst. Prof. Dr. Jessada Tanthannuch)

Chairperson

[illegible]

(Assoc. Prof. Dr. Eckart Schulz)

Member (Thesis Advisor)

[illegible]

(Asst. Prof. Dr. Pisamai Kitipoom

Member

Benjawan Rodjanadid

(Asst. Prof. Dr. Benjawan Rodjanadid)

Member

Panu Yimmuang

(Asst. Prof. Dr. Panu Yimmuang)

Member

Chatchai Jothity

(Assoc. Prof. Dr. Chatchai Jothityangkoon)

Vice Rector for Academic Affairs and Quality Assurance

[illegible]

(Prof. Dr. Santi Maensiri)

Dean of Institute of Science

จิกมี นัมง์ยาล : ซัพพอร์ตเวกเตอร์แมชชีนเพื่อการวินิจฉัยโรคจอตาเหตุเบาหวาน (MULTI-CLASS SUPPORT VECTOR MACHINES FOR DIABETIC RETINOPATHY DIAGNOSIS) อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร.เอ็กการ์ท ชูลซ์, 130 หน้า.

คำสำคัญ: โรคจอตาเหตุเบาหวาน/ซัพพอร์ตเวคเตอร์แมชชีน/กลยุทธ์การตัดสินใจประเภทหลายคลาส

วิทยานิพนธ์นี้ประยุกต์ใช้ซัพพอร์ตเวคเตอร์แมชชีน 5 รูปแบบที่แตกต่างกันสำหรับการจำแนก 4 ระยะของโรคจอตาเหตุเบาหวานชนิดเป็นน้อย ข้อมูลที่ใช้ในการวิเคราะห์คือภาพถ่ายอวัยวะดวงตา 400 ภาพ ซึ่งเป็นข้อมูล ที่ได้รับจากที่เก็บ Messidor หลังจากนั้นเตรียมข้อมูลเบื้องต้นโดยใช้การแยกคุณลักษณะ 13 คุณลักษณะ และใช้คุณลักษณะที่เหมาะสมที่สุดในการนำเข้าข้อมูลสำหรับการจำแนกซัพพอร์ตเวคเตอร์แมชชีน พบว่า การจำแนกโรคจอตาเหตุเบาหวานแบบรุนแรงมีค่าความแม่นยำเท่ากับ 97.44% เมื่อใช้ซัพพอร์ตเวคเตอร์แมชชีนที่มีเคอร์เนลแบบเกาส์เซียน สำหรับการใช้ซัพพอร์ตเวคเตอร์แมชชีนแบบคู่มีความไวสูงสุดเท่ากับ 99.06% การตัดสินใจแบบหลายคลาสใน 4 ระยะของโรคจอตาเหตุเบาหวานทำให้เกิดความแม่นยำ ความไว ความจำเพาะ และความเที่ยงตรงสูงสุดด้วยซัพพอร์ตเวคเตอร์แมชชีนที่มีขอบเขตแบบคู่ในการกำหนดค่าการตัดสินใจแบบหนึ่งต่อหนึ่ง โดยใช้กลยุทธ์การตัดสินใจประเภทหลายคลาสที่รวมระยะทางจากไฮเปอร์เพลนในอัลกอริธึมการเลือกคลาส ผลการเปรียบเทียบอยู่ในเกณฑ์ดีเมื่อเปรียบเทียบกับข้อมูลที่ตีพิมพ์ในงานวิจัยอื่น ๆ

สาขาวิชาคณิตศาสตร์ ลายมือชื่อนักศึกษา [illegible]

ปีการศึกษา 2564 ลายมือชื่ออาจารย์ที่ปรึกษา [illegible]

JIGME NAMGYAL : MULTICLASS LEAST SQUARE SUPPORT VECTOR MACHINES FOR DIABETIC RETINOPATHY DIAGNOSIS. THESIS ADVISOR : ASSOC. PROF. ECKART SCHULZ, Ph.D. 130 PP.

Keywords: DIABETIC RETINOPATHY/SUPPORT VECTOR MACHINE/MULTICLASS DECISION STRATEGY

This work applies five variants of the support vector machine for the classification of the four stages of nonproliferative diabetic retinopathy. Four hundred eye fundus images from the Messidor repository are preprocessed and thirteen features extracted. The features best suited as inputs for support vector machine classification are identified. The binary classification of severe diabetic retinopathy alone achieves an accuracy of 97.44% when optimized for accuracy, using the standard support vector machine with Gaussian kernel. When optimized for sensitivity, the improved version of twin support vector machine achieved the highest sensitivity of 99.06%. Multiclass decision into all 4 stages of diabetic retinopathy achieves highest accuracy, sensititivity, specificity and precision with the twin bounded support vector machine classifier in one-versus-one decision configuration, by using a novel decision strategy that includes distances from the decision hyperplanes in the class selection algorithm. The results compare favorably with data published in the literature.

School of Mathematics	Student's Signature [illegible]
Academic Year 2021	Advisor's Signature [illegible]

ACKNOWLEDGEMENTS

This thesis would not have seen the light if not for my kind and humble advisor Assoc. Prof. Dr. Eckart Schulz. He has groomed me with his well-experienced supervision, guidance, knowledge and moral support in the courses that he have taught and in particular, in this research project. I would always cherish all those long and fruitful discussions on this research. Moreover, I have always appreciated professor's kind insights, wise and careful decisions on the project. I have learned from him on how to analyze and think critically when it comes to comprehending and learning new course materials, and learning mathematics through details and preciseness. Besides, I would also like to offer my sincere gratitude for rendering personal help when I had several health issues during my study. For all this, I am always grateful to my advisor and I could not have imagined having better advisor and mentor during my study.

I would also like to express my appreciation to Asst. Prof. Dr. Jessada Tanthanuch for his generous support, guidance and knowledge that he shared in the class. I am also thankful to Prof. Dr. Sergey Meleshko and Assoc. Prof. Dr. Sayan Kaennakham for their teachings and valuable comments on the courses that I have undertaken with them. And a big thank you to all the committee members and the lecturers at the School of Mathematics, Suranaree University of Technology.

I would also like to offer my profound gratitude to His Majesty's Secretariat, the office of Gyalpoi Zimpon, Youth Welfare and Education, Bhutan, for granting a King's scholarship to pursue this study. The opportunity has been fulfilling and I pledge to serve my nation in my full capacity.

I also thank the management of my workplace, College of Science and Technology, Royal University of Bhutan, for the financial aid for my quarantine expenses in Thailand.

Special acknowledgement also goes to the Vithebundit scholarship offered by Suranaree University of Technology without which this study would not have been possible.

I also personally thank my classmates especially Mr. Titayawat Khumwong and Ms. Warapron Srisup for their invaluable help during my difficult times when I was ill. To my Vietnamese friends, Mr. Nhieu, Mr. Dông, Mr. Behn and Ms. Quynh, thank you for your friendship during my stay at SUT.

Finally, I take this opportunity to wholeheartedly thank my parents, family members and friends for their encouragement, motivation and inspiration.

Jigme Namgyal

CONTENTS

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

CHAPTER I

INTRODUCTION

The eye is a very delicate organ, yet of great importance in a person's daily life. While defects of the eye do not affect ones overall physical health directly, they may impair the interaction with the environment to an extreme degree and make it very difficult to manage even simple daily chores. With increasingly sedentary lifestyles and longevity being observed, the share of the population that suffers from health issues affecting the eye such as high blood pressure or diabetes, has risen worldwide to previously unknown levels.

The World Health Organization (WHO) reported that around 422 million people in the world suffer from Diabetes Mellitus (2014 estimates). This figure is expected to rocket to an even higher number by 2030. Diabetes is a chronic health condition due to insufficient insulin production in the body by the pancreas, or the body cannot utilize the produced insulin. Diabetes are categorized into type 1 diabetes and type 2 diabetes. Type 1 diabetes is due to insulin deficiency and some of the symptoms include excessive urine excretion, vision changes, constant hunger, thirst, and fatigue. Type 2 diabetes is a consequence when the body is not able to effectively utilize the insulin. The symptoms are similar to type 1 diabetes, but appear often only late when the disease is quite advanced. Type 2 diabetes is most common among the patients. Diabetes gradually affects eyes, heart, kidneys, blood vessels and nerves.

One of the most common microvascular complications in people living with diabetes is a condition called diabetic retinopathy (DR). It affects nearly 25-44% of the patients at some point in their life. This is caused when there is increased glucose level in the blood vessels of the retina. Other risk factors include high blood pressure and high serum lipids which are common in diabetic patients. As a consequence, it damages blood vessels in the retina making it to become leaky or blocked. In the advanced stage of the disease, abnormal blood vessels appear from the retina which can bleed and cause retina scarring which results in permanent vision loss or blindness. It is reported that DR, in some patients, can occur as early as within three years on average after diagnosis of diabetes. Moreover, after 20 years of diagnosis of diabetes, every type 1 diabetic patient suffers from DR, whereas 50-80% of type 2 diabetic patients will suffer DR. It is important to detect early signs of diabetic retinopathy in a patient, in order to take measures which can delay or mitigate the deterioration of the eye.

The rapid progress in computing technology achieved in recent years has made possible the development of a great variety of computing tools to assist physicians in diagnosis and treatment. An important component of diagnosis is decision making; to decide whether a patient suffers from some particular ailment and to determine to what degree.

One of the common tools for decision making in machine learning is the Support Vector Machine (SVM). Support vector machines have already been employed with success for diabetic retinopathy classification (Carrera, González, and Carrera, 2017) and (Kandhasamy et al., 2020). Soon after the development of the first support vector machines, a modified support vector machine called *Least Squares Support Vector Machine* was introduced by Suykens and Vandewalle in 1999. One of its advantages is that the training process is simpler, since there are only equality constraints in the mathematical optimization problems of the model. Later-on, further modifications appeared, among them the *Twin Support Vector Machine* and the *Least Squares Twin Support Vector Machine*. It is thus worthwhile to test how well these modified support vector machines perform for diabetic retinopathy analysis and classification.

1.1 Literature review

The machine-aided detection of diseases of the eye is an ongoing and very active field of research. The techniques consist of two essential steps: Image preprocessing, to identify, highlight and extract the relevant features from an image, followed by the model selection and training. Neural networks and support vector machines are popular tools employed for this latter task.

Acharya et al. (2008) proposed an automated grading of diabetic retinopathy stages by a multiclass directed acyclic graph support vector machine algorithm. A total of 300 retinal fundus images with 60 subjects each from normal, mild, moderate and severe non-proliferative diabetic retinopathy, as well as proliferative diabetic retinopathy were used. Histogram equalization was applied as a preprocessing technique. The pre-processed grayscale images were then converted to 1 dimensional data using a Radon transform. Then 18 features were extracted using higher order spectra from which four features were selected using statistical p-test. On classification, an accuracy and sensitivity of 82% each, and specificity of 88% were obtained.

As a typical sign of advanced diabetic retinopathy is the growth of new blood vessels, Welikala et al. (2015) focus on the detection of new abnormal blood vessels. They use two support vector machines, one to distinguish new blood vessels from normal blood vessels, and a second one to distinguish new blood vessels from exudates. A genetic algorithm is used for feature selection. Image regions are identified as new blood vessels only when both machines agree.

Imani et al. (2015) use the variance of shearlet coefficients to first identify whether the blood vessels in an image are of sufficient quality for automatic processing. This is followed by morphological component analysis using shearlet and curvelet transform to separate out retinal vessels and exudates. Then they extract local features and compare them to a database of these features that has been created from a pool of training images, achieving an accuracy of 92.82% for binary classification.

Carrera et al. (2017) have proposed computer assisted diagnosis of non-proliferative diabetic retinopathy stages using digital processing of retinal images and multiclass support vector machine. In total 400 optic fundus, 152 normal, 30 mild, 69 moderate and 149 severe non-proliferative diabetic retinopathy cases were considered. For feature extraction, blood vessels were segmented and their densities obtained, microaneurysms and hard exudates were isolated. Eight features were extracted and fed to the support vector machine classifier. On performance evaluation, a sensitivity of 94.6% and a predictive capacity up to 94% were reported.

James et al. (2018) have introduced a method to detect exudates by morphological operations, blood vessels and haemorrhages by complement function, and microaneurysms by edge detection techniques. Relevant features were obtained and multilayered feed forward neural network was used for classification.

Hui et al. (2019) use two deep convolutional neural networks to construct features from an image. Combined with other features directly extracted directly from the image, they train a multiclass support vector machine for severity grading, including non-proliferative and proliferative diabetic retinopathy and report an overall accuracy of 86.17%.

Sarki et al. (2020) give a comprehensive review of recent publications on diabetic eye and other eye disease detection through deep learning. Practically all publications reviewed make use of convolutional neural networks. Of the referenced papers, Ghosh et al. (2017) obtain an accuracy of about 95% for binary classification, and of 85% for multiclass detection on a data set which includes proliferative retinopathy. Hemanth et al. (2020) obtain a binary classification accuracy of 97%.

Kandhasamy et al. (2020) have presented a multi level set segmentation algorithm and support vector machine with selective features along with genetic algorithm for diabetic retinopathy severity grading. Mathematical morphological operations and principal component analysis were used for clustering before the feature extraction. They reported area under receiver operating curve of 98.01% on average for all severity levels.

Hierarchical severity grading system for detection and classification of non-proliferative diabetic retinopathy was studied by Bhardwaj et al. (2020), where they employed support vector machine and k-nearest neighbor classifiers. From statistical analysis 12 optimal features were obtained and fed to respective classifiers. It was concluded that the k-nearest neighbor classifier achieved best performance accuracy and computation time both in classifying diabetic and non-diabetic retinopathy images and in grading severity level of diabetic retinopathy cases.

An interracial DR screening artificial intelligence (AI) created model was developed by Katada et al. (2020). Convolutional neural network and support vector machine were used to train the model on American fundus dataset. They showed that the trained AI model exhibited higher sensitivity and specificity even when tested on 200 another racial

Japanese subjects in detection of referable diabetic retinopathy.

Dandapat et al. (2021) compare support vector machine, K-nearest neighbor and convolutional neural networks for binary diabetic retinopathy detection and obtain accuracies of 96.62%, 94.38% and 94.74%, respectively.

Overall, most publications using support vector machine focus on binary classification, and use the regular support vector machine. There are only few reports on the multiclass detection of diabetic retinopathy by support vector machines. It should be noted that the results in the literature are not directly comparable as they may be using different datasets.

1.2 Research objectives

The purpose of the proposed research is to develop and test models for diabetic retinopathy detection and classification by support vector machine methods, and in particular,

1. to classify different stages of diabetic retinopathy using multiclass support vector machines,
2. to compare the performance of the various support vector machine models for this task,
3. to evaluate the best multiclass decision strategy, with the aim to
4. improve on the performance of current support vector machine models.

1.3 Thesis outline

This thesis is organized as follows. The following chapter, Chapter II, gives a comprehensive review of the support vector machine and some of its variations. Chapter III discusses multiclass classification by support vector machines and presents a new type of decision strategy for multiclass twin support vector machines. Chapter IV presents the structure of the human eye and the problem of diabetic retinopathy. Chapter V discusses the image pre-processing methods which were used to enhance the eye fundus images. Chapter VI explains the methodologies used in the classification of the eye fundus images. Chapter VII presents the results together with a discussion, and Chapter VIII constitutes a summary of this work.

CHAPTER II

SUPPORT VECTOR MACHINES

This chapter presents a review of the concepts and detailed explanations of the theory of the support vector machines encountered in this thesis. These include the standard support vector machine, the least squares support vector machine, the twin support vector machine and some of their variants.

Troughout we work mainly in $\mathbb{R}^n$. The inner product in any Hilbert space is denoted by $\langle \cdot, \cdot \rangle$. When working in $\mathbb{R}^n$ this is mostly the usual dot product, which can also be expressed in matrix product form,

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{y}^T \boldsymbol{x}, \qquad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

All norms are those induced from the inner product, $\|\boldsymbol{x}\| = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle}$.

2.1 Binary classification

The mathematical principle of *classification* is as follows: Working in Euclidean space, vectors $\boldsymbol{x} \in \mathbb{R}^n$ represent data to be classified. In *binary classification*, the space $\mathbb{R}^n$ is split into two disjoint sets,

$$\mathbb{R}^n = A^+ \cup A^-, \qquad (A^+ \cap A^- = \emptyset),$$

and one seeks a function $f : \mathbb{R}^n \to \{\pm 1\}$ called *decision function* which separates these two sets:

$$f(\boldsymbol{x}) = \begin{cases} +1, & \boldsymbol{x} \in A^+ \\ -1, & \boldsymbol{x} \in A^-. \end{cases}$$

In *multiclass classification*, the space $\mathbb{R}^n$ is split into a finite number of disjoint sets,

$$\mathbb{R}^n = \bigcup_{i=1}^{N} A_i, \qquad (A_i \cap A_j = \emptyset \text{ for } i \neq j),$$

and the decision function sought should yield

$$f(\boldsymbol{x}) = i \qquad \text{if } \boldsymbol{x} \in A_i \quad (i \in \{1, \dots N\}).$$

The sets A^+, A^-, respectively $A_1, \dots A_N$ are called *classes*, and a value $y = f(\boldsymbol{x})$ is called the *label* (or sometimes simply *class*) of a vector $\boldsymbol{x}$.

2.2 Support vector machines

2.2.1 Hard margin support vector machines

Support vector machines (SVM), introduced by Vapnik and co-workers (1998), are machine learning techniques popularly used for classification and regression analysis. Although SVMs are classically designed for binary classification problems, the method can also be extended to multiclass classification problems.

The basic, *hard margin support vector machine* is a simple tool for binary classification. The assumption is that the sets A^+ and A^- are open half spaces separated by a hyperplane. A *hyperplane* H in $\mathbb{R}^n$ is an affine subspace of codimension 1, i.e.

$$H = \boldsymbol{x}_o + V,$$

where V is a linear subspace of $\mathbb{R}^n$ of dimension $n-1$, so that $\dim(V^\perp) = 1$. Up to a scalar multiple, there is thus a unique nonzero vector $\boldsymbol{w} \in \mathbb{R}^n$ with $\boldsymbol{w} \perp V$. For any $\boldsymbol{x} \in \mathbb{R}^n$,

$$\boldsymbol{x} \in H \iff \boldsymbol{x} = \boldsymbol{x}_o + \boldsymbol{v} \ \ (\exists \boldsymbol{v} \in V) \iff \boldsymbol{x} - \boldsymbol{x}_o \in V \iff \boldsymbol{w} \perp (\boldsymbol{x} - \boldsymbol{x}_o)$$

so that

$$H = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0\} \qquad \text{where} \quad b = -\langle \boldsymbol{w}, \boldsymbol{x}_o \rangle.$$

Thus, the pair $(\boldsymbol{w}, b)$ uniquely determines the hyperplane, which we may denote as $H(\boldsymbol{w}, b)$. By a hyperplane H separating the sets A^+ and A^- one means that after choosing the appropriate direction for $\boldsymbol{w}$, then

$$A^+ = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b > 0\} \quad \text{and} \quad A^- = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b < 0\}.$$

Then the decision function will be

$$f(\boldsymbol{x}) = \text{sgn}\big(\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b\big).$$

The length of the vector $\boldsymbol{w}$ may be normalized as explained further below.

In general, all machine learning tools depend on internal parameters. In the basic support vector machine these are the vector $\boldsymbol{w}$ (which determines the direction of the hyperplane) and the *bias* b (which determines its location in space). The appropriate values of the internal parameters are obtained by a process called *training*.

In training one starts from a collection of data (i.e. vectors $\boldsymbol{x}$) whose labels are known: Given is a *training data* set

$$\Gamma = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_l, y_l)\},$$

where $\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \{+1, -1\}$, $i = 1, 2, \dots, l$. We now describe how to find the parameter values that best separate the training data Γ according to their labels.

Training a SVM means that one searches for a hyperplane $H = H(\boldsymbol{w}, b)$ which separates and has largest distance from the *positive and negative training sets*

$$D^+ = \{\boldsymbol{x}_i : (\boldsymbol{x}_i, y_i) \in \Gamma,\ y_i = 1\} \quad \text{and} \quad D^- = \{\boldsymbol{x}_i : (\boldsymbol{x}_i, y_i) \in \Gamma,\ y_i = -1\}.$$

First of all, these two sets must be *linearly separable*, that is, there must exist a hyperplane $H = H(\boldsymbol{w}, b)$ so that

$$\text{sgn}\left(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b\right) = y_i \qquad \forall\, \boldsymbol{x}_i \in D = D^+ \cup D^-.$$

Second, the hyperplane should be equidistant from the two sets. It is not difficult to show that the distance from a vector $\boldsymbol{x}$ to a hyperplane $H = H(\boldsymbol{x}, b)$ is

$$d(H, x) = \frac{|\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b|}{\|\boldsymbol{w}\|}.$$

The distance from H to the sets D^+ and D^- is

$$d(H, D^+) = \min_{\boldsymbol{x}_i \in D^+} \frac{|\langle w, \boldsymbol{x}_i\rangle + b|}{\|\boldsymbol{w}\|} \quad \text{and} \quad d(H, D^-) = \min_{\boldsymbol{x}_i \in D^-} \frac{|\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b|}{\|\boldsymbol{w}\|}.$$

Let $\boldsymbol{x}^+ \in D^+$ and $\boldsymbol{x}^- \in D^-$ denote the data points in D^+ and D^-, respectively, that are closest to H, i.e.

$$\begin{aligned} d(H, D^+) = d(H, \boldsymbol{x}^+) &= \frac{|\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle + b|}{\|\boldsymbol{w}\|} = \frac{\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle + b}{\|\boldsymbol{w}\|}, \\ d(H, D^-) = d(H, \boldsymbol{x}^-) &= \frac{|\langle \boldsymbol{w}, \boldsymbol{x}^-\rangle + b|}{\|\boldsymbol{w}\|} = -\frac{\langle \boldsymbol{w}, \boldsymbol{x}^-\rangle + b}{\|\boldsymbol{w}\|}. \end{aligned} \tag{2.1}$$

The two hyperplanes parallel to the desired $H(\boldsymbol{w}, b)$ passing through $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ are

$$\begin{aligned} &H^+ : \langle \boldsymbol{w}, \boldsymbol{x}\rangle + b^+ = 0 \qquad \text{where } b^+ = -\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle \quad \text{and} \\ &H^- : \langle \boldsymbol{w}, \boldsymbol{x}\rangle + b^- = 0 \qquad \text{where } b^- = -\langle \boldsymbol{w}, \boldsymbol{x}^-\rangle. \end{aligned}$$

Then $b^+ < 0$ while $b^- > 0$. From here on, one agrees to normalize the vector $\boldsymbol{w}$ so that

$$b^- - b^+ = \langle \boldsymbol{w}, \boldsymbol{x}^+ - \boldsymbol{x}^-\rangle = 2. \tag{2.2}$$

The vectors $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ are called *support vectors* as they lie in the *supporting hyperplanes* H^+, respectively H^-. The requirement $d(H, D^+) = d(H, D^-)$ now gives that

$$b = \frac{-\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle - \langle \boldsymbol{w}, \boldsymbol{x}^-\rangle}{2} = \frac{b^+ + b^-}{2}.$$

Then for $\boldsymbol{x} \in D^+$ one has by (2.1) and since $\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b > 0$ that

$$\frac{\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b}{\|\boldsymbol{w}\|} = d(H, \boldsymbol{x}) \geq d(H, D^+) = \frac{\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle + b}{\|\boldsymbol{w}\|}$$

so that

$$\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b \geq \langle \boldsymbol{w}, \boldsymbol{x}^+\rangle + b = -b^+ + \frac{b^+ + b^-}{2} = \frac{b^- - b^+}{2} = 1 \qquad (x \in D^+). \tag{2.3}$$

Similarly, for $\boldsymbol{x} \in D^-$ one has, since $\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b < 0$, that

$$\frac{\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b}{\|\boldsymbol{w}\|} = -d(H, \boldsymbol{x}) \leq -d(H, D^-) = \frac{\langle \boldsymbol{w}, \boldsymbol{x}^-\rangle + b}{\|\boldsymbol{w}\|}$$

and thus

$$\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b \leq \langle \boldsymbol{w}, \boldsymbol{x}^-\rangle + b = -b^- + \frac{b^+ + b^-}{2} = \frac{-b^- + b^+}{2} = -1 \qquad (x \in D^-), \tag{2.4}$$

with equality in the case of support vectors. That is,

$$y_i\big(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b\big) \geq 1 \qquad (i = 1, \dots, l)$$

as required. Furthermore, (2.3) and (2.4) show that the supporting hyperplanes have equations

$$H^+ = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 1\} \quad \text{and} \quad H^- = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = -1\}.$$

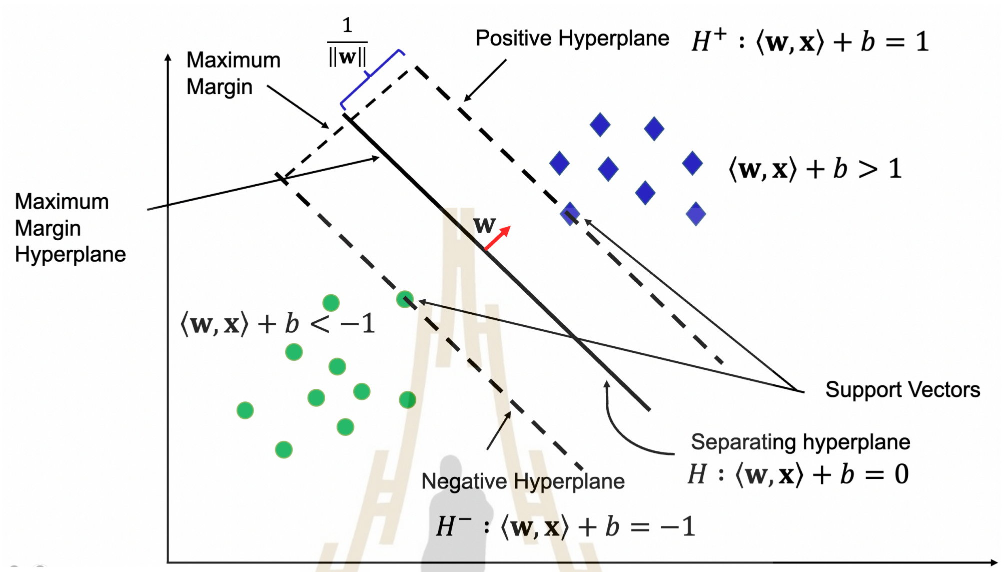

$\frac{1}{\|\mathbf{w}\|}$
Maximum Margin
Positive Hyperplane $H^+ : \langle \mathbf{w}, \mathbf{x} \rangle + b = 1$
$\langle \mathbf{w}, \mathbf{x} \rangle + b > 1$
Maximum Margin Hyperplane
$\mathbf{w}$
$\langle \mathbf{w}, \mathbf{x} \rangle + b < -1$
Support Vectors
Separating hyperplane
$H : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$
Negative Hyperplane
$H^- : \langle \mathbf{w}, \mathbf{x} \rangle + b = -1$

Figure 2.1 A hard margin SVM.

In general, there are many hyperplanes $H(\boldsymbol{w}, b)$ with corresponding supporting hyperplanes $H^+(\boldsymbol{w}, b)$ and $H^-(\boldsymbol{w}, b)$ separating the training data. One chooses the hyperplane H which maximizes the distance between H^+ and H^- (This distance is called the *margin*). Since this distance is

$$d(H^+, H^-) = d(H, \boldsymbol{x}^+) + d(H, \boldsymbol{x}^-) = \frac{\langle \boldsymbol{w}, \boldsymbol{x}^+ \rangle + b}{\|\boldsymbol{w}\|} - \frac{\langle \boldsymbol{w}, \boldsymbol{x}^- \rangle + b}{\|\boldsymbol{w}\|} = \frac{b^- - b^+}{\|\boldsymbol{w}\|} = \frac{2}{\|\boldsymbol{w}\|}$$

then the margin will be largest, if the normalized vector $\boldsymbol{w}$ has smallest norm. To obtain a differentiable function one minimizes $\|\boldsymbol{w}\|^2$ instead of $\|\boldsymbol{w}\|$, which is equivalent, and then solves the convex quadratic programming problem with affine inequality constraints,

$$\begin{cases} \min\limits_{\boldsymbol{w}, b} \dfrac{1}{2}\|\boldsymbol{w}\|^2 \\ y_i\big(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b\big) \geq 1 \qquad (i = 1, \dots, l). \end{cases} \tag{2.5}$$

Observe that the normalization condition (2.2) is implicitly contained in the inequality constraint. There are standard software routines to numerically solve such an optimization problem.

In practice, the training data is often not properly linearly separable. There are two tools to overcome this problem: Adding *slack variables* and the *kernel trick*.

2.2.2 Slack variables

If the training data is nearly linearly separable, except for some outliers, then *slack variables* $\xi_1, \dots, \xi_l$ are introduced. Problem (2.5) is modified to the convex quadratic programming problem

$$\begin{cases} \min\limits_{\boldsymbol{w}, b, \boldsymbol{\xi}} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum\limits_{i=1}^{l}\xi_i \\ y_i\big(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b\big) \geq 1 - \xi_i \quad (i = 1, \dots, l) \\ \xi_i \geq 0, \qquad\qquad\qquad\qquad (i = 1, \dots, l) \end{cases} \tag{2.6}$$

where $\boldsymbol{\xi} = (\xi_1, \dots, \xi_l)^T \in \mathbb{R}^l$ is the vector of slack variables, and $C > 0$ is a penalty parameter which controls the trade-off between maximizing the margin (i.e. minimizing $\|\boldsymbol{w}\|$) and minimizing the misclassification. A value $0 < \xi_i < 1$ means that the corresponding training point $\boldsymbol{x}_i$ is still correctly classified, but lies on the wrong side of the corresponding supporting hyperplane H^+, respectively H^-. A value of $\xi_i \geq 1$ means that $\boldsymbol{x}_i$ is misclassified, as it lies on the wrong side of the separating hyperplane H, or on H itself. This type of SVM is called a *soft margin support vector machine*.

In Figure 2.2, the data points $\boldsymbol{x}_3, \boldsymbol{x}_4$ and $\boldsymbol{x}_5$ have non-zero slack variables. $\xi_3 < 1$, while $\xi_4 > 2$ and $\xi_5 > 1$. Thus, the point $\boldsymbol{x}_3$ is still correctly classified, while $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$ are not. $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are support vectors.

2.2.3 The kernel trick

When the training data is far from being linearly separable, one may employ a mapping

$$\Phi : \mathbb{R}^n \to \mathcal{H}$$

of $\mathbb{R}^n$ into a real Hilbert space $\mathcal{H}$ of higher finite or infinite dimension. The training sets D^+ and D^- will be mapped to

$$\widetilde{D}^+ = \big\{\Phi(\boldsymbol{x}) : \boldsymbol{x} \in D^+\big\} \qquad \text{resp.} \qquad \widetilde{D}^- = \big\{\Phi(\boldsymbol{x}) : \boldsymbol{x} \in D^-\big\}$$

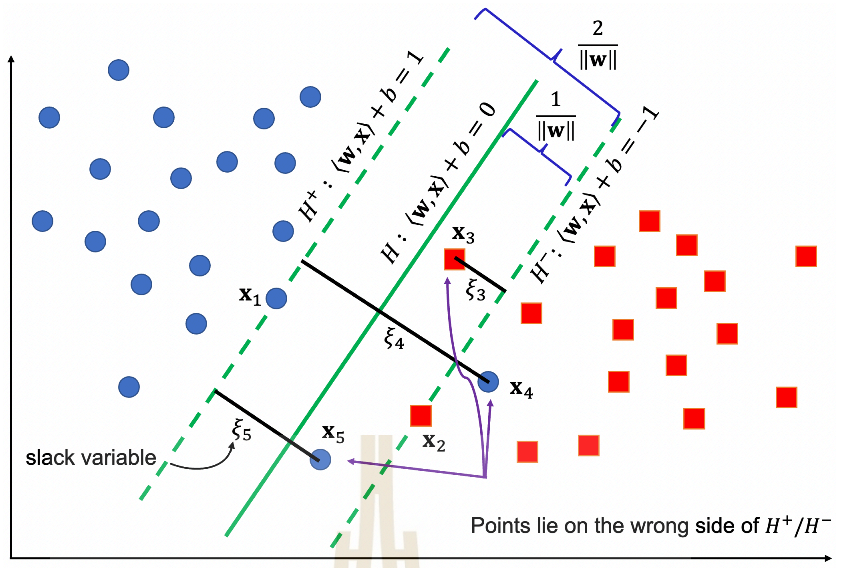

$\frac{2}{\|\mathbf{w}\|}$
$\frac{1}{\|\mathbf{w}\|}$
$H^+ : \langle \mathbf{w}, \mathbf{x} \rangle + b = 1$
$H : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$
$H^- : \langle \mathbf{w}, \mathbf{x} \rangle + b = -1$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
$\mathbf{x}_5$
ξ_3
ξ_4
ξ_5
slack variable
Points lie on the wrong side of H^+/H^-

Figure 2.2 A soft margin SVM.

and the separating hyperplane will now be located in $\mathcal{H}$. Modifying (2.6), one thus has to solve a problem, called the *primal problem*,

$$\begin{cases} \min\limits_{\boldsymbol{w}\in\mathcal{H},b,\boldsymbol{\xi}} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ y_i\big(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i)\rangle + b\big) \geq 1 - \xi_i \quad (i = 1,\ldots,l) \\ \xi_i \geq 0 \quad (i = 1,\ldots,l) \end{cases} \tag{2.7}$$

and decision function will thus be of the form

$$f(\boldsymbol{x}) = \operatorname{sgn}\big(\langle \boldsymbol{w}, \Phi(\boldsymbol{x})\rangle + b\big). \tag{2.8}$$

Observe that in the above, the inner product and norm are those in $\mathcal{H}$.

Figure 2.3 shows training data which is not linearly separable, but can easily be mapped into a higher dimensional space so that it becomes linearly separable.

2.2.4 The dual problem

Because $\mathcal{H}$ may have very high or even infinite dimension, it may not feasible to solve the minimization problem (2.7) directly. Instead, one solves another simpler problem, called the *dual* problem, using the theory of convex optimization. We briefly

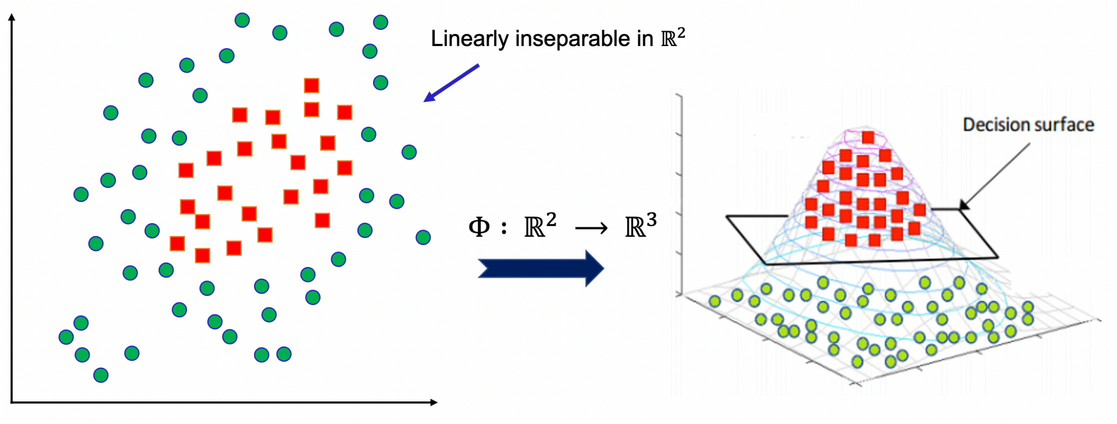

Linearly inseparable in $\mathbb{R}^2$
$\Phi : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$
Decision surface

Figure 2.3 Making data linearly separable.

review some of its concepts, simplified as they apply here. Details and proofs can be found in the books by Boyd and Venderberge (2004) and Deng, Tian, and Zhang (2013).

Consider a convex optimization problem with affine constraints[*],

$$\begin{cases} \min\limits_{\boldsymbol{x}\in\mathbb{R}^n} \; f_0(\boldsymbol{x}) \\ \quad \text{subject to} \\ f_i(\boldsymbol{x}) = a_i^T\boldsymbol{x} + p_i \leq 0, \quad i = 1\ldots m \\ h_j(\boldsymbol{x}) = c_j^T\boldsymbol{x} + q_j = 0, \quad j = 1\ldots s \end{cases}$$

where the function $f_0 : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and convex, $a_i, c_j \in \mathbb{R}^n$ and $p_i, q_j \in \mathbb{R}$. This problem can be restated in vector form,

$$\begin{cases} \min\limits_{\boldsymbol{x}\in\mathbb{R}^n} \; f_0(\boldsymbol{x}) \\ \quad \text{subject to} \\ \boldsymbol{f}(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{p} \leq 0 \\ \boldsymbol{h}(\boldsymbol{x}) = C\boldsymbol{x} + \boldsymbol{q} = 0 \end{cases} \tag{2.9}$$

where $\boldsymbol{f}$ and $\boldsymbol{h}$ are vector valued functions, A and C are matrices and $\boldsymbol{p}$ and $\boldsymbol{q}$ are

[*]In SVMs the constraints are of affine type; this restriction simplifies the exposition somewhat.

vectors:

$$\boldsymbol{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix}, \quad \boldsymbol{h} = \begin{pmatrix} h_1 \\ \vdots \\ h_s \end{pmatrix}, \quad A = \begin{pmatrix} a_1^T \\ \vdots \\ a_m^\top \end{pmatrix}, \quad C = \begin{pmatrix} c_1^T \\ \vdots \\ c_s^\top \end{pmatrix}, \quad \boldsymbol{p} = \begin{pmatrix} p_1 \\ \vdots \\ p_m \end{pmatrix}, \quad \boldsymbol{q} = \begin{pmatrix} q_1 \\ \vdots \\ q_s \end{pmatrix}.$$

The feasible region $\mathcal{S}$ is the subset of $\mathbb{R}^n$ satisfying the constraints, and is convex as the constraints are affine. Next construct a function $L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) : \mathcal{S} \times R^m \times \mathbb{R}^s \to \mathbb{R}$ called the *Lagrangian* by

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f_0(\boldsymbol{x}) + \langle \boldsymbol{\lambda}, \boldsymbol{f}(\boldsymbol{x}) \rangle + \langle \boldsymbol{\nu}, \boldsymbol{h}(\boldsymbol{x}) \rangle$$

and define the *Lagrange dual function* $g : R^m \times \mathbb{R}^s \to [-\infty, \infty)$ by

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\boldsymbol{x} \in \mathcal{S}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}).$$

The newly introduced variables $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are called *Lagrange multipliers*. Then $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is a concave function. The *dual problem* of (2.9) is

$$\begin{cases} \max\limits_{\boldsymbol{\lambda}, \boldsymbol{\nu}} \ g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \qquad \text{subject to} \\ \boldsymbol{\lambda} \geq 0 \end{cases} \tag{2.10}$$

which is again a convex optimization problem. There is a connection between the optimal solutions of problem (2.9) and its dual problem (2.10) through Theorem 2.1 below. But first we give a definition.

Definition 2.1. A triple $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) \in \mathcal{S} \times \mathbb{R}^m \times \mathbb{R}^s$ is said to satisfy the *Karush–Kuhn–Tucker (KKT) conditions* if

(KKT1) $\boldsymbol{f}(\boldsymbol{x}^*) \leq 0$

(KKT2) $\boldsymbol{h}(\boldsymbol{x}^*) = 0$

(KKT3) $\boldsymbol{\lambda}^* \geq 0$

(KKT4) $\boldsymbol{\lambda}_i^* f_i(\boldsymbol{x}^*) = 0 \quad (i = 1, \dots l)$

(KKT5) $\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) = \nabla_{\boldsymbol{x}} f_0(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i^* \nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}^*) + \sum_{j=1}^{s} \nu_j^* \nabla_{\boldsymbol{x}} h_j(\boldsymbol{x}^*) = 0.$

Here, $\nabla_{\boldsymbol{x}} L$ denotes the gradient of $L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$ with respect to $\boldsymbol{x}$, $\boldsymbol{\lambda}^* = (\lambda_1^*, \dots \lambda_m^*)$ and $\boldsymbol{\nu}^* = (\nu_1^*, \dots \nu_s^*)$. In case there is no inequality constraint, i.e. $\boldsymbol{f} = 0$, then only the two conditions (KKT2) and (KKT5) are non-vacuous, and they constitute a reformulation of the usual method of Lagrange multipliers given equality constraints.

Theorem 2.1. *1. Problems (2.9) and (2.10) both have solutions $\boldsymbol{x}^*$ and $(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$, respectively.*

2. For any such solution pair, $f_0(\boldsymbol{x}^*) = g(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$.

3. The pair $\boldsymbol{x}^*$ and $(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ forms a solution to (2.9) and (2.10) if and only the KKT conditions hold.

Now apply the above the KKT conditions to the SVM optimization problem (2.7). Setting $\boldsymbol{x} = (\boldsymbol{w}, b, \boldsymbol{\xi})^T$, $m = 2l$ and $s = 0$ then $\boldsymbol{\lambda} = (\boldsymbol{\alpha}, \boldsymbol{\beta})^T$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}^l$, and the Lagrangian becomes

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i \Big[y_i \big(\langle w, \Phi(x_i) \rangle + b \big) + \xi_i - 1 \Big] - \sum_{i=1}^{l} \beta_i \xi_i.$$

Let $\mathbf{1} = (1, 1, \dots, 1)^T$ be the constant vector and $\boldsymbol{y} = (y_1, \dots, y_l)^T \in \mathbb{R}^l$ the vector of labels. Then the Langrangian can be rewritten as

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \langle C\mathbf{1} - \boldsymbol{\alpha} - \boldsymbol{\beta}, \boldsymbol{\xi} \rangle - \sum_{i=1}^{l} \alpha_i y_i \langle w, \Phi(x_i) \rangle - b\langle \boldsymbol{y}, \boldsymbol{\alpha} \rangle + \langle \mathbf{1}, \boldsymbol{\alpha} \rangle$$

and hence

$$\nabla_{(\boldsymbol{w}, b, \boldsymbol{\xi})} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{bmatrix} \boldsymbol{w} - \sum_{i=1}^{l} \alpha_i y_i \Phi(x_i) \\ -\langle \boldsymbol{\alpha}, \boldsymbol{y} \rangle \\ C\mathbf{1} - \boldsymbol{\alpha} - \boldsymbol{\beta} \end{bmatrix}.$$

Now if $\boldsymbol{x} = (\boldsymbol{w}, b, \boldsymbol{\xi})^T$ minimizes L for some fixed $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ (with a finite minimum), then necessarily, $\nabla_{(\boldsymbol{w}, b, \boldsymbol{\xi})} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = 0$ at this point, and in particular,

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \Phi(x_i), \qquad \langle \boldsymbol{\alpha}, \boldsymbol{y} \rangle = 0 \qquad \text{and} \qquad C\mathbf{1} - \boldsymbol{\alpha} - \boldsymbol{\beta} = 0. \tag{2.11}$$

Substituting these into L we obtain the dual function

$$\begin{aligned}
g(\boldsymbol{\alpha},\boldsymbol{\beta}) &= \min_{\boldsymbol{w},b,\boldsymbol{\xi}} L(\boldsymbol{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) \\
&= \frac{1}{2}\left\|\sum_{i=1}^{l}\alpha_i y_i \Phi(x_i)\right\|^2 - \sum_{i=1}^{l}\alpha_i y_i \left\langle \sum_{j=1}^{l}\alpha_j y_j \Phi(x_j), \Phi(x_i)\right\rangle + \sum_{i=1}^{l}\alpha_i \\
&= -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j \left\langle \Phi(x_j), \Phi(x_i)\right\rangle + \sum_{i=1}^{l}\alpha_i
\end{aligned}$$

which is quadratic. In fact, it is of the form

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) = g(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} + \langle \mathbf{1},\boldsymbol{\alpha}\rangle$$

where Ω is the $l \times l$ symmetric and positive semidefinite matrix

$$\Omega = [q_{ij}], \qquad q_{ij} = y_i y_j \langle\, \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)\,\rangle.$$

The dual problem is then of the form

$$\begin{cases}
\max\limits_{\boldsymbol{\alpha},\boldsymbol{\beta}}\; g(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} + \langle \mathbf{1},\boldsymbol{\alpha}\rangle \\
\qquad \text{subject to} \\
\langle \boldsymbol{\alpha},\boldsymbol{y}\rangle = 0, \qquad C\mathbf{1} - \boldsymbol{\alpha} - \boldsymbol{\beta} = 0 \\
\boldsymbol{\alpha} \geq 0, \qquad \boldsymbol{\beta} \geq 0
\end{cases} \tag{2.12}$$

where the inequalities in the last row are just (KKT3). This problem can be simplified: Since the objective function is independent of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha},\boldsymbol{\beta} > 0$ it suffices to solve

$$\begin{cases}
\max\limits_{\boldsymbol{\alpha}}\; g(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} + \langle \mathbf{1},\boldsymbol{\alpha}\rangle \\
\qquad \text{subject to} \\
\langle \boldsymbol{\alpha},\boldsymbol{y}\rangle = 0 \\
0 \leq \boldsymbol{\alpha} \leq C\mathbf{1}
\end{cases}$$

This is usually formulated as a minimization problem,

$$\begin{cases}
\min\limits_{\boldsymbol{\alpha}}\; \frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} - \langle \mathbf{1},\boldsymbol{\alpha}\rangle \\
\qquad \text{subject to} \\
\langle \boldsymbol{\alpha},\boldsymbol{y}\rangle = 0 \\
0 \leq \boldsymbol{\alpha} \leq C\mathbf{1}
\end{cases} \tag{2.13}$$

Note that the size of this dual problem depends on the number of training data points, but is independent of the dimension of the Hilbert space $\mathcal{H}$. After solving this problem for a maximizer $\boldsymbol{\alpha}$, then the $\boldsymbol{w}$-part of the minimizer of (2.7) can be computed by (2.11). Indeed, one has:

Theorem 2.2. *Let $\boldsymbol{\alpha}$ be the solution to the optimization problem (2.13). Suppose, there exists j with $0 < \alpha_j < C$. Then the components $\boldsymbol{w}$ and b in the solution of the primal problem (2.7) are*

$$\begin{aligned} &1) \qquad \boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \Phi(\boldsymbol{x}_i) \qquad \textit{and} \\ &2) \qquad b = y_j - \sum_{i=1}^{l} \alpha_i y_i \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle \end{aligned} \tag{2.14}$$

This method of solving the dual problem has a fundamental advantage in that one need not know the map Φ at all ! In fact, define a map $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ called the *kernel* by

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle \qquad (\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^n).$$

Then the entries of the matrix Ω in the dual problem (2.13) are of the form

$$q_{ij} = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

and the decision function (2.8) becomes

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad f(\boldsymbol{x}) &= \text{sgn}\big(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}) \rangle + b\big) = \text{sgn}\left(\left\langle \sum_{i=1}^{l} \alpha_i y_i \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}) \right\rangle + b \right) \\ &= \text{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b \right) \end{aligned} \tag{2.15}$$

where b is as in (2.14),

$$b = y_j - \sum_{i=1}^{l} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

for some j with $0 < \alpha_j < C$. This shows that once the kernel is known, the dual problem can be solved and the decision function obtained without knowledge of the map Φ. There are various ways to construct kernels without knowledge of the corresponding

Hilbert space $\mathcal{H}$ or the map Φ, see (Deng, Tian, and Zhang, 2013). A popular kernel is the *Gaussian kernel*, also called RBF kernel

$$K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\|\boldsymbol{x}-\boldsymbol{x}'\|^2/2\sigma^2}.$$

When Φ is the identity mapping, then $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{x}, \boldsymbol{x} \rangle$, and we recover the support vector machine (2.6). This is called the *linear kernel*.

2.3 Least squares support vector machines

Least squares support vector machine (LSSVM) is a variant of the SVM and was proposed by Suykens and Vandevalle (2003). Different from a standard SVM, a least squares SVM deals with solving a system of linear equations rather than solving the convex quadratic programming problem. In this view, the problem is simpler and faster to train as compared to the standard SVM.

Consider again training data

$$\Gamma = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$$

where $\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \{+1, -1\}$, $i = 1, 2, \ldots, l$, and the corresponding *positive and negative* training sets:

$$\widetilde{D}+ = \{\Phi(\boldsymbol{x}_i) : (\boldsymbol{x}_i, y_i) \in \Gamma,\ y_i = 1\} \qquad \text{and}$$
$$\widetilde{D}^- = \{\Phi(\boldsymbol{x}_i) : (\boldsymbol{x}_i, y_i) \in \Gamma,\ y_i = -1\}.$$

The intuition of LSSVM is that the parallel supporting hyperplanes H^+ and H^- should be placed so that the training points in $\tilde{D}^+$ and $\tilde{D}^-$ are as close as possible to their respective supporting hyperplanes ("proximal"), while at the same time maximizing the margin and minimizing the classification error. Figure 2.4 shows that in a LSSVM, the hyperplanes H^+ and H^- pass through the central region of each training data class.

Thus, one poses the primal problem

$$\begin{cases} \displaystyle\min_{\boldsymbol{w} \in \mathcal{H}, b, \boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{2}\sum_{i=1}^{l} \xi_i^2 \\ y_i(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i) \rangle + b) = 1 - \xi_i \qquad (i = 1, \ldots, l) \end{cases} \tag{2.16}$$

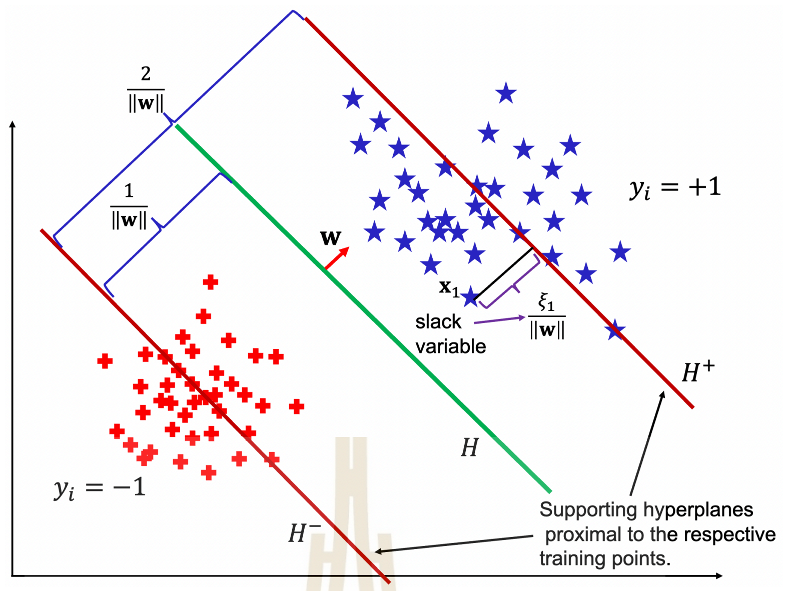

$\frac{2}{\|\mathbf{w}\|}$
$\frac{1}{\|\mathbf{w}\|}$
$\mathbf{w}$
$y_i = +1$
$\mathbf{x}_1$
slack variable
$\frac{\xi_1}{\|\mathbf{w}\|}$
H^+
H
$y_i = -1$
H^-
Supporting hyperplanes proximal to the respective training points.

Figure 2.4 A Least Squares SVM.

and where $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_l)^T$. As with a soft margin SVM, a value of $\xi_i \geq 1$ of a slack variable means that the corresponding data $\boldsymbol{x}_i$ is misclassified. However, unlike in the soft margin SVM, ξ_i can also be negative. The error term $\sum \xi_i^2$ now measures how far the ensemble of data points $\Phi(\boldsymbol{x}_i)$ is located from their respective supporting hyperplanes, and squaring increases the penalty quadratically with distance. For convenience in differentiation, the penalty factor is written in the form $\frac{C}{2}$. We note that this primal problem has equality constraints only.

The Lagrangian function is thus

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{2}\|\boldsymbol{\xi}\|^2 - \sum_{i=1}^{l} \alpha_i \underbrace{[y_i(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i)\rangle + b) + \xi_i - 1]}_{h_i(\boldsymbol{w}, b, \boldsymbol{\xi})}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)^T$ is a Lagrange multiplier, $\boldsymbol{y} = (y_1, \ldots, y_l)^T$, $\mathbf{1} = (1, 1, \ldots, 1)^T$ and $h_i(\boldsymbol{w}, b, \boldsymbol{\xi}) = 0$ are the constraints in (2.16). Because there are no inequality constraints, one can use the standard method of Lagrange multipliers by first solving the equation

$$\nabla_{(\boldsymbol{w}, b, \boldsymbol{\xi})} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = 0$$

which is

$$\begin{pmatrix} \boldsymbol{w} - \sum_{i=1}^{l} \alpha_i y_i \Phi(\boldsymbol{x}_i) \\ \langle \boldsymbol{\alpha}, \boldsymbol{y} \rangle \\ C\boldsymbol{\xi} - \boldsymbol{\alpha} \end{pmatrix} = 0.$$

From the third component of this gradient one obtains that $\boldsymbol{\xi} = \frac{1}{C}\boldsymbol{\alpha}$, from the second that $\boldsymbol{y}^T\boldsymbol{\alpha} = \langle \boldsymbol{\alpha}, \boldsymbol{y} \rangle = 0$ and from the first that

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \Phi(\boldsymbol{x}_i).$$

Exchanging the symbols i and j above and substituting $\boldsymbol{w}$ and $\boldsymbol{\xi}$ into the constraints $h_i(\boldsymbol{w}, b, \boldsymbol{\xi}) = 0$, one must solve the system of equations for $\boldsymbol{\alpha}$,

$$\begin{aligned} y_i \sum_{j=1}^{l} \alpha_j y_j \langle \Phi(\boldsymbol{x}_j), \Phi(\boldsymbol{x}_i) \rangle + y_i b + \frac{1}{C}\alpha_i - 1 &= 0 \qquad (i = 1, \ldots l) \\ \boldsymbol{y}^T \boldsymbol{\alpha} &= 0 \end{aligned}$$

In matrix form,

$$\left(\begin{array}{c|c} \boldsymbol{y} & \Omega + \frac{1}{C} I \\ \hline 0 & \boldsymbol{y}^T \end{array}\right) \left(\begin{array}{c} b \\ \hline \boldsymbol{\alpha} \end{array}\right) = \left(\begin{array}{c} \boldsymbol{1} \\ \hline 0 \end{array}\right)$$

where $\Omega \in M_{l \times l}(\mathbb{R})$ is again the symmetric matrix $\Omega = [q_{ij}]_{i,j}$ where

$$q_{ij} = y_i y_j \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad i, j = 1, \ldots, l,$$

with appropriate kernel function K.

Solving this system of equations yields $\boldsymbol{\alpha}$ and b. The decision function is then

$$f(\boldsymbol{x}) = \text{sgn}\big(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}) \rangle + b\big) = \text{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right)$$

which takes the same form as the decision function (2.15) of the usual support vector machine. Again, knowledge of the kernel is enough; one need neither know the Hilbert space $\mathcal{H}$ nor the map Φ.

2.4 Twin support vector machines

Recall that the purpose of the least squares support vector machine is to cluster the data close to two parallel hyperplanes. In a *twin support vector machine* the two hyperplanes need no longer be parallel. This variant of support vector machine was first introduced in (Jayadeva et al., (2007)).

2.4.1 The linear twin support vector machine

We first discuss the *linear* twin support vector machine. Here, one searches for hyperplanes H_1 and H_2, so that the positive data points in D^+ are closest to H_1, while at the same time the data from D^- lie beyond the -1 margin of H_1, up to slack variables. Similarly, the negative data points in D^- should be closest to H_2, while those from D^+ should be beyond the -1 margin of H_2, up to slack variables.

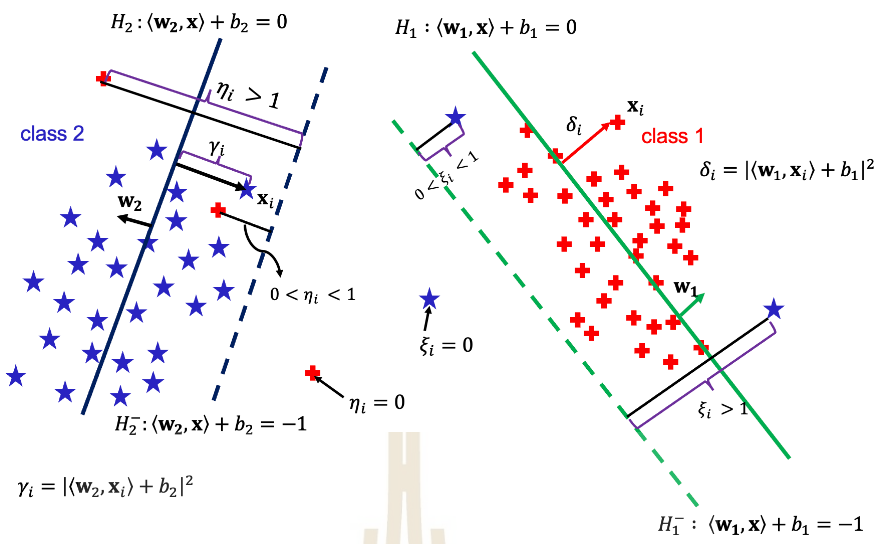
Relabel the training data so that

$$D^+ = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p\} \subseteq \mathbb{R}^n \qquad \text{and} \qquad D^- = \{\boldsymbol{x}_{p+1}, \ldots, \boldsymbol{x}_{p+q}\} \subseteq \mathbb{R}^n$$

so that $p + q = l$. For example, to find $H_1 : \langle \boldsymbol{w}_1, \boldsymbol{x}\rangle + b_1 = 0$ one is to solve the minimization problem

$$\begin{cases} \displaystyle\min_{\boldsymbol{w}_1, b_1, \xi_1, \ldots \xi_q} \frac{1}{2}\sum_{i=1}^{p} \frac{|\langle \boldsymbol{w}_1, \boldsymbol{x}_i\rangle + b_1|^2}{\|\boldsymbol{w}_1\|^2} + C_1 \sum_{j=1}^{q} \xi_j \\ \quad \text{subject to} \\ -(\langle \boldsymbol{w}_1, \boldsymbol{x}_{p+j}\rangle + b_1) \geq 1 - \xi_j & (j = 1, \ldots, q) \\ \xi_j \geq 0 & (j = 1, \ldots, q) \end{cases}$$

(Compare with (2.7)) The first term in the objective function forces the positive data points to be proximal to the hyperplane, while the the second term minimizes the sum of the slack variables belonging to the negative data points. Division by $\|\boldsymbol{w}_1\|^2$ can been removed to make this a convex problem; this can be countered by a different optimal choice of the parameter C_1.



$H_2 : \langle \mathbf{w_2}, \mathbf{x} \rangle + b_2 = 0$
$H_1 : \langle \mathbf{w_1}, \mathbf{x} \rangle + b_1 = 0$
$\eta_i > 1$
class 2
γ_i
$\mathbf{x}_i$
$\mathbf{w_2}$
$0 < \eta_i < 1$
$H_2^- : \langle \mathbf{w_2}, \mathbf{x} \rangle + b_2 = -1$
$\eta_i = 0$
$\gamma_i = |\langle \mathbf{w_2}, \mathbf{x}_i \rangle + b_2|^2$
δ_i
$\mathbf{x}_i$
class 1
$\delta_i = |\langle \mathbf{w_1}, \mathbf{x}_i \rangle + b_1|^2$
$0 < \xi_i < 1$
$\mathbf{w_1}$
$\xi_i = 0$
$\xi_i > 1$
$H_1^- : \langle \mathbf{w_1}, \mathbf{x} \rangle + b_1 = -1$

Figure 2.5 A Twin SVM.

In addition, by reversing direction of the vector $\boldsymbol{w}_1$ which also reverses the sign of b_1, the minus sign on the second line can be removed, to obtain

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_1}, b_1, \xi_1, \ldots \xi_q} \frac{1}{2} \sum_{i=1}^{p} |\langle \boldsymbol{w}_1, \boldsymbol{x}_i \rangle + b_1|^2 + C_1 \sum_{j=1}^{q} \xi_j \\ \quad \text{subject to} \\ \langle \boldsymbol{w}_1, \boldsymbol{x}_{p+j} \rangle + b_1 \geq 1 - \xi_j \qquad (j = 1, \ldots, q) \\ \xi_j \geq 0 \qquad (j = 1, \ldots, q) \end{cases} \tag{2.17}$$

Similarly, the hyperplane $H_2 : \langle \boldsymbol{w}_2, \boldsymbol{x} \rangle + b_2 = 0$ can be found by solving

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_2}, b_2, \eta_1, \ldots \eta_p} \frac{1}{2} \sum_{i=p+1}^{p+q} |\langle \boldsymbol{w}_2, \boldsymbol{x}_i \rangle + b_2|^2 + C_2 \sum_{j=1}^{p} \eta_j \\ \quad \text{subject to} \\ \langle \boldsymbol{w}_2, \boldsymbol{x}_j \rangle + b_2 \geq 1 - \eta_j \qquad (j = 1, \ldots, p) \\ \eta_j \geq 0 \qquad (j = 1, \ldots, p) \end{cases} \tag{2.18}$$

This is simpler expressed in matrix form: Let X_1 be the $p \times n$-matrix whose rows consist of the data in D^+, and X_2 the $q \times n$ matrix whose rows consist of the data in D^-. That is,

$$X_1 = \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_p^T \end{bmatrix} \qquad \text{and} \qquad X_2 = \begin{bmatrix} \boldsymbol{x}_{p+1}^T \\ \vdots \\ \boldsymbol{x}_{p+q}^T \end{bmatrix}.$$

Then (2.17) and (2.18) can be expressed in the Euclidean norm as

$$\begin{cases} \min\limits_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}} \ \frac{1}{2}\|X_1\boldsymbol{w}_1 + b_1\boldsymbol{e}_1\|^2 + C_1\boldsymbol{e}_2^T\boldsymbol{\xi} \\ \quad \text{subject to} \\ X_2\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 \geq \boldsymbol{e}_2 - \boldsymbol{\xi}, \qquad \boldsymbol{\xi} \geq 0 \end{cases} \tag{2.19}$$

and

$$\begin{cases} \min\limits_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}} \ \frac{1}{2}\|X_2\boldsymbol{w}_2 + b_2\boldsymbol{e}_2\|^2 + C_2\boldsymbol{e}_1^T\boldsymbol{\eta} \\ \quad \text{subject to} \\ X_1\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \geq 0. \end{cases} \tag{2.20}$$

where $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}^n$, $b_1, b_2 \in \mathbb{R}$, $\boldsymbol{e}_1 = (1, 1, \ldots, 1)^T \in \mathbb{R}^p$, $\boldsymbol{e}_2 = (1, 1, \ldots, 1)^T \in \mathbb{R}^q$, while $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_q)^T$ and $\boldsymbol{\eta} = (\eta_1, \eta_2, \ldots, \eta_p)^T$ are the vectors of slack variables.

A data point is now classified to belong to the class whose hyperplane is closest. Thus, the decision function is

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \dfrac{|\langle \boldsymbol{w}_1, \boldsymbol{x}\rangle + b_1|}{\|\boldsymbol{w}_1\|} \leq \dfrac{|\langle \boldsymbol{w}_2, \boldsymbol{x}\rangle + b_2|}{\|\boldsymbol{w}_2\|} \\ -1 & \text{else.} \end{cases} \tag{2.21}$$

Often one uses the simplified decision function

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad |\langle \boldsymbol{w}_1, \boldsymbol{x}\rangle + b_1| \leq |\langle \boldsymbol{w}_2, \boldsymbol{x}\rangle + b_2| \\ -1 & \text{else.} \end{cases} \tag{2.22}$$

2.4.2 The nonlinear twin support vector machine

Just as with the regular support vector machine, one may map the given data into a higher dimensional Hilbert space by a map

$$\Phi : \mathbb{R}^n \to \mathcal{H}$$

with corresponding kernel

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle \,.$$

As before, the training sets D^+ and D^- will be mapped to

$$\widetilde{D}^+ = \left\{ \Phi(\boldsymbol{x}) : \boldsymbol{x} \in D^+ \right\} \qquad \text{resp.} \qquad \widetilde{D}^- = \left\{ \Phi(\boldsymbol{x}) : \boldsymbol{x} \in D^- \right\}.$$

Because the vectors in $\widetilde{D} = \widetilde{D}^+ \cup \widetilde{D}^-$ form a finite dimensional subspace of $\mathcal{H}$, one may choose $\mathcal{H}$ to be the span of these vectors, so that $\mathcal{H}$ is of finite dimension. Hence following (2.17) and (2.18) , the twin hyperplanes $H_1 : \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}) \rangle + b_1 = 0$ and $H_2 : \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}) \rangle + b_2 = 0$ in $\mathcal{H}$ can be found by solving the systems

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_1}, b_1, \xi_1, \dots \xi_q} \ \frac{1}{2} \sum_{i=1}^{p} \left| \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_i) \rangle + b_1 \right|^2 + C_1 \sum_{j=1}^{q} \xi_j \\ \quad \text{subject to} \\ \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_{p+j}) \rangle + b_1 \geq 1 - \xi_j & (j = 1, \dots, q) \\ \xi_j \geq 0 & (j = 1, \dots, q) \end{cases} \tag{2.23}$$

and

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_2}, b_2, \eta_1, \dots \eta_p} \ \frac{1}{2} \sum_{i=p+1}^{p+q} \left| \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_i) \rangle + b_2 \right|^2 + C_2 \sum_{j=1}^{p} \eta_j \\ \quad \text{subject to} \\ \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_j) \rangle + b_2 \geq 1 - \eta_j & (j = 1, \dots, p) \\ \eta_j \geq 0 & (j = 1, \dots, p) \end{cases} \tag{2.24}$$

with $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{H}$. The simplified decision function (2.25) will then be of the form

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \left| \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}) \rangle + b_1 \right| \leq \left| \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}) \rangle + b_2 \right| \\ -1 & \text{else.} \end{cases} \tag{2.25}$$

while the normalized decision function (2.26) will be

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \dfrac{\left| \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}) \rangle + b_1 \right|}{\| \boldsymbol{w}_1 \|} \leq \dfrac{\left| \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}) \rangle + b_2 \right|}{\| \boldsymbol{w}_2 \|}. \\ -1 & \text{else.} \end{cases} \tag{2.26}$$

The difficulty here, however, is that the mapping Φ is unknown in general; only its kernel is known. Since $\mathcal{H}$ is spanned by the vectors in $\widetilde{D}$, one expresses $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ as linear combinations of the elements in $\widetilde{D}$, e.g.

$$\boldsymbol{w}_1 = \sum_{k=1}^{p+q} u_k \Phi(\boldsymbol{x}_k), \qquad \boldsymbol{w}_2 = \sum_{k=1}^{p+q} v_k \Phi(\boldsymbol{x}_k). \tag{2.27}$$

This permits to express the vectors $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ as coordinate vectors $\boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathbb{R}^{p+q}$,

$$\boldsymbol{u}_1 = \begin{bmatrix} u_1 \\ \vdots \\ u_{p+q} \end{bmatrix} \in \mathbb{R}^{p+q} \quad \text{and} \quad \boldsymbol{u_2} = \begin{bmatrix} v_1 \\ \vdots \\ v_{p+q} \end{bmatrix} \in \mathbb{R}^{p+q}.$$

Note that the coordinates u_k, v_k are not unique in general, because the collection of vectors $\Phi(\boldsymbol{x}_k)$ need not be linearly independent in $\mathcal{H}$! The inner products in the left-hand sums of (2.23) and (2.24) become

$$\big\langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_i)\big\rangle = \sum_{k=1}^{p+q} u_k \big\langle \Phi(\boldsymbol{x}_k), \Phi(\boldsymbol{x}_i)\big\rangle = \sum_{k=1}^{p+q} K(\boldsymbol{x}_i, \boldsymbol{x}_k) u_k$$

$$\big\langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_i)\big\rangle = \sum_{k=1}^{p+q} v_k \big\langle \Phi(\boldsymbol{x}_k), \Phi(\boldsymbol{x}_i)\big\rangle = \sum_{k=1}^{p+q} K(\boldsymbol{x}_i, \boldsymbol{x}_k) v_k$$

and similarly, the inner products in the inequality conditions become

$$\begin{aligned} \big\langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_{p+j})\big\rangle &= \sum_{k=1}^{p+q} K(\boldsymbol{x}_{p+j}, \boldsymbol{x}_k) u_k \\ \big\langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_j)\big\rangle &= \sum_{k=1}^{p+q} K(\boldsymbol{x}_j, \boldsymbol{x}_k) v_k. \end{aligned} \tag{2.28}$$

Thus, (2.23) and (2.24) can be expressed in matrix notation as

$$\begin{cases} \min\limits_{\boldsymbol{u}_1, b_1, \boldsymbol{\xi}} \dfrac{1}{2}\|X_1\boldsymbol{u}_1 + b_1\boldsymbol{e}_1\|^2 + C_1\boldsymbol{e}_2^T\boldsymbol{\xi} \\ \quad \text{subject to} \\ X_2\boldsymbol{u}_1 + b_1\boldsymbol{e}_2 \geq \boldsymbol{e}_2 - \boldsymbol{\xi}, \qquad \boldsymbol{\xi} \geq 0 \end{cases} \tag{2.29}$$

and

$$\begin{cases} \min\limits_{\boldsymbol{u}_2, b_2, \boldsymbol{\eta}} \dfrac{1}{2}\|X_2\boldsymbol{u}_2 + b_2\boldsymbol{e}_2\|^2 + C_2\boldsymbol{e}_1^T\boldsymbol{\eta} \\ \quad \text{subject to} \\ X_1\boldsymbol{u}_2 + b_2\boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \geq 0. \end{cases} \tag{2.30}$$

where now X_1 is a $p \times (p+q)$ matrix and X_2 is a $q \times (p+q)$ matrix,

$$X_1 = \left[K(\boldsymbol{x}_i, \boldsymbol{x}_k)\right]_{1 \le i \le p, 1 \le k \le p+q} \qquad X_2 = \left[K(\boldsymbol{x}_{p+i}, \boldsymbol{x}_k)\right]_{p+1 \le i \le p+q, 1 \le k \le p+q}$$

and again, $b_1, b_2 \in \mathbb{R}$, $\boldsymbol{e}_1 = (1, 1, \dots, 1)^T \in \mathbb{R}^p$, $\boldsymbol{e}_2 = (1, 1, \dots, 1)^T \in \mathbb{R}^q$, while $\boldsymbol{\xi} = (\xi_1, \xi_2, \dots, \xi_q)^T$ and $\boldsymbol{\eta} = (\eta_1, \eta_2, \dots, \eta_p)^T$ are the vectors of slack variables. Finally, using (2.28) the decision function (2.25) can be expressed as

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \left|\sum_{k=1}^{p+q} u_k K(\boldsymbol{x}_k, \boldsymbol{x}) + b_1\right| \le \left|\sum_{k=1}^{p+q} v_k K(\boldsymbol{x}_k, \boldsymbol{x}) + b_2\right| \\ -1 & \text{else} \end{cases}$$

which can be simplified to

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \left|\langle \boldsymbol{u}_1, \boldsymbol{K}(\boldsymbol{x})\rangle + b_1\right| \le \left|\langle \boldsymbol{u}_2, \boldsymbol{K}(\boldsymbol{x})\rangle + b_2\right| \\ -1 & \text{else} \end{cases} \tag{2.31}$$

where $\boldsymbol{K}(\boldsymbol{x})$ is the vector $\left(K(\boldsymbol{x_1}, \boldsymbol{x}), K(\boldsymbol{x_2}, \boldsymbol{x}) \dots, K(\boldsymbol{x_{p+q}}, \boldsymbol{x})\right)^T \in \mathbb{R}^{p+q}$. Observe that the two sets of equations (2.19), (2.20), (2.22), respectively (2.29), (2.30), (2.31) have the same form. To obtain the normalized decision function (2.26) we observe that

$$\begin{aligned} \|\boldsymbol{w}_1\|^2 = \langle \boldsymbol{w}_1, \boldsymbol{w}_1 \rangle &= \left\langle \sum_{i=1}^{p+q} u_i \Phi(\boldsymbol{x}_j), \sum_{k=1}^{p+q} u_k \Phi(\boldsymbol{x}_k) \right\rangle \\ &= \sum_{i,k=1}^{p+q} u_i \left\langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_k)\right\rangle u_k = \boldsymbol{u}_1^T X \boldsymbol{u}_1. \end{aligned} \tag{2.32}$$

and similarly,

$$\|\boldsymbol{w}_2\|^2 = \boldsymbol{u}_2^T X \boldsymbol{u}_2$$

where X is the $(p+q) \times (p+q)$ *Gramian matrix*

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \left[K(\boldsymbol{x}_i, \boldsymbol{x}_k)\right]_{1 \le i,k \le p+q}.$$

Thus the normalized decision function (2.26) becomes

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \dfrac{\left|\langle \boldsymbol{u}_1, \boldsymbol{K}(\boldsymbol{x})\rangle + b_1\right|}{\sqrt{\boldsymbol{u}_1^T X \boldsymbol{u}_1}} \le \dfrac{\left|\langle \boldsymbol{u}_2, \boldsymbol{K}(\boldsymbol{x})\rangle + b_2\right|}{\sqrt{\boldsymbol{u}_2^T X \boldsymbol{u}_2}} \\ -1 & \text{else.} \end{cases} \tag{2.33}$$

2.4.3 The dual problem for the twin support vector machine

Next we discuss the dual problem for the nonlinear twin support hyperplanes, applying Theorems 2.1 and 2.2. The linear case will be a special, simplified version and is omitted.

We start with problem (2.29)-(2.30). To simplify notation, we set

$$\begin{aligned} &\tilde{X}_1 = \begin{bmatrix} X_1 & \boldsymbol{e}_1 \end{bmatrix} \in M_{p,p+q+1} && \tilde{X}_2 = \begin{bmatrix} X_2 & \boldsymbol{e}_2 \end{bmatrix} \in M_{q,p+q+1}, \\ &\tilde{X} = \begin{bmatrix} X & 0 \\ 0 & 1 \end{bmatrix} \in M_{p+q+1,p+q+1}, && \\ &\tilde{\boldsymbol{u}}_1 = \begin{bmatrix} \boldsymbol{u}_1 \\ b_1 \end{bmatrix} \in \mathbb{R}^{p+q+1} && \tilde{\boldsymbol{u}}_2 = \begin{bmatrix} \boldsymbol{u}_2 \\ b_2 \end{bmatrix} \in \mathbb{R}^{p+q+1}. \end{aligned} \tag{2.34}$$

Then (2.29) can be expressed as

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi}} \dfrac{1}{2}\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\|^2 + C_1\langle \boldsymbol{e}_2, \boldsymbol{\xi}\rangle \\ \qquad \text{subject to} \\ \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1 \le 0, \qquad -\boldsymbol{\xi} \le 0. \end{cases} \tag{2.35}$$

while (2.30) can be expressed as

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_2,\boldsymbol{\eta}} \dfrac{1}{2}\|\tilde{X}_2\tilde{\boldsymbol{u}}_2\|^2 + C_2\langle \boldsymbol{e}_1, \boldsymbol{\eta}\rangle \\ \qquad \text{subject to} \\ \boldsymbol{e}_1 - \boldsymbol{\eta} - \tilde{X}_1\tilde{\boldsymbol{u}}_2 \le 0, \qquad -\boldsymbol{\eta} \le 0 \end{cases} \tag{2.36}$$

Proceeding in a similar way to the discussion after Theorem 2.1, the Lagrangian of (2.35) now is

$$\begin{aligned} L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{2}\left\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\right\|^2 + C_1\langle \boldsymbol{e}_2, \boldsymbol{\xi}\rangle + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1\rangle + \langle \boldsymbol{\beta}, -\boldsymbol{\xi}\rangle \\ &= \frac{1}{2}\left\langle \tilde{X}_1^T\tilde{X}_1\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_1\right\rangle - \left\langle \tilde{X}_2^T\boldsymbol{\alpha}, \tilde{\boldsymbol{u}}_1\right\rangle + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2\rangle + \langle C_1\boldsymbol{e}_2 - \boldsymbol{\alpha} - \boldsymbol{\beta}, \boldsymbol{\xi}\rangle \end{aligned}$$

for $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}^q$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \ge 0$. Now the matrix $\tilde{X}_1^T\tilde{X}_1$ is only positive semidefinite. In order to be able to invert it lateron, one adds a small multiple of the identity matrix,

$$\begin{aligned} L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\left\langle \left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_1\right\rangle - \left\langle \tilde{X}_2^T\boldsymbol{\alpha}, \tilde{\boldsymbol{u}}_1\right\rangle + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2\rangle \\ + \langle C_1\boldsymbol{e}_2 - \boldsymbol{\alpha} - \boldsymbol{\beta}, \boldsymbol{\xi}\rangle \end{aligned} \tag{2.37}$$

for some small positive δ. By (KKT5), its gradient must vanish,

$$0 = \nabla_{(\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi})} L(\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) = \begin{bmatrix} \left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 - \tilde{X}_2^T\boldsymbol{\alpha} \\ C_1\boldsymbol{e}_2 - \boldsymbol{\alpha} - \boldsymbol{\beta} \end{bmatrix}.$$

This gives

$$\left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 - \tilde{X}_2^T\boldsymbol{\alpha} = 0 \qquad \text{and} \qquad C_1\boldsymbol{e}_2 - \boldsymbol{\alpha} - \boldsymbol{\beta} = 0. \tag{2.38}$$

The first equation can be solved for $\tilde{\boldsymbol{u}}_1$:

$$\tilde{\boldsymbol{u}}_1 = \left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{\alpha} \qquad (\delta \text{ small}). \tag{2.39}$$

Eliminating $\boldsymbol{\beta} = C_1\boldsymbol{e}_2 - \boldsymbol{\alpha}$ from the second equation, using the fact that $\boldsymbol{\alpha},\boldsymbol{\beta} \geq 0$, one obtains

$$0 \leq \boldsymbol{\alpha} \leq C_1\boldsymbol{e}_2. \tag{2.40}$$

Now substituting (2.39) and the second equation of (2.38) into the Lagrangian (2.37), (here $(\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi})$ takes the role of $\boldsymbol{x}^*$ and $\boldsymbol{\alpha}$ that of $(\boldsymbol{\lambda}^*,\boldsymbol{\nu}^*)$ in Theorem 2.1) one obtains

$$\begin{aligned} g(\boldsymbol{\alpha}) = \frac{1}{2}\Big\langle(\tilde{X}_1^T\tilde{X}_1{+}\delta I_{p+q+1})(\tilde{X}_1^T\tilde{X}_1{+}\delta I_{p+q+1})^{-1}\tilde{X}_2^T\boldsymbol{\alpha},\, (\tilde{X}_1^T\tilde{X}_1{+}\delta I_{p+q+1})^{-1}\tilde{X}_2^T\boldsymbol{\alpha}\Big\rangle \\ - \Big\langle\tilde{X}_2^T\boldsymbol{\alpha},\, (\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1})^{-1}\tilde{X}_2^T\boldsymbol{\alpha}\Big\rangle + \langle\boldsymbol{\alpha},\boldsymbol{e}_2\rangle. \end{aligned}$$

Because the matrix $\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}$ is self-adjoint, this simplifies to

$$\begin{aligned} g(\boldsymbol{\alpha}) &= \frac{1}{2}\left\langle\left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{\alpha}, \tilde{X}_2^T\boldsymbol{\alpha}\right\rangle \\ &\quad - \left\langle\left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{\alpha}, \tilde{X}_2^T\boldsymbol{\alpha}\right\rangle + \langle\boldsymbol{\alpha},\boldsymbol{e}_2\rangle \\ &= -\frac{1}{2}\left\langle\left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{\alpha}, \tilde{X}_2^T\boldsymbol{\alpha}\right\rangle + \langle\boldsymbol{\alpha},\boldsymbol{e}_2\rangle. \end{aligned}$$

Setting

$$\Omega = \tilde{X}_2\left(\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T \in M_q(\mathbb{R})$$

and expressing the inner product in terms of matrix multiplication, the dual problem (2.13) is here

$$\begin{cases} \max\limits_{\boldsymbol{\alpha}} \ \left[g(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} + \langle\boldsymbol{\alpha},\boldsymbol{e}_2\rangle\right] \\ \qquad \text{subject to} \\ 0 \leq \boldsymbol{\alpha} \leq C_1\boldsymbol{e}_2. \end{cases}$$

Expressed as a minimization problem,

$$\begin{cases} \min\limits_{\boldsymbol{\alpha}} \ \frac{1}{2}\boldsymbol{\alpha}^T\Omega\boldsymbol{\alpha} - \langle \boldsymbol{\alpha}, \boldsymbol{e}_2 \rangle \\ \qquad \text{subject to} \\ 0 \le \boldsymbol{\alpha} \le C_1\boldsymbol{e}_2. \end{cases} \tag{2.41}$$

In a similar way, the dual problem of (2.36) is

$$\begin{cases} \min\limits_{\boldsymbol{\gamma}} \ \frac{1}{2}\boldsymbol{\gamma}^T\Gamma\boldsymbol{\gamma} - \langle \boldsymbol{\gamma}, \boldsymbol{e}_1 \rangle \\ \qquad \text{subject to} \\ 0 \le \boldsymbol{\gamma} \le C_2\boldsymbol{e}_1 \end{cases} \tag{2.42}$$

where

$$\Gamma = \tilde{X}_1 \left(\tilde{X}_2^T \tilde{X}_2 + \delta I_{p+q+1} \right)^{-1} \tilde{X}_1^T \in M_p(\mathbb{R}).$$

Once these optimization problems have been solved, then $\tilde{\boldsymbol{u}}_1$ can be found using (2.39), and similarly,

$$\tilde{\boldsymbol{u}}_2 = \left(\tilde{X}_2^T \tilde{X}_2 + \delta I_{p+q+1} \right)^{-1} \tilde{X}_1^T \boldsymbol{\gamma}. \tag{2.43}$$

Finally, the decision functions (2.31) and (2.33) remain. The former can be written using (2.4.3) as

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \left| \langle \tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{K}}(\boldsymbol{x}) \rangle \right| \le \left| \langle \tilde{\boldsymbol{u}}_2, \tilde{\boldsymbol{K}}(\boldsymbol{x}) \rangle \right| \\ -1 & \text{else} \end{cases} \tag{2.44}$$

where now $\tilde{\boldsymbol{K}}(\boldsymbol{x}) = \left(K(\boldsymbol{x_1}, \boldsymbol{x}), K(\boldsymbol{x_2}, \boldsymbol{x}) \ldots, K(\boldsymbol{x_{p+q}}, \boldsymbol{x}), 1 \right)^T \in \mathbb{R}^{p+q+1}$.

2.4.4 The twin bounded support vector machine

In contrast to the usual support vector machines, twin support vector machines do not maximize the margins of the respective hyperplanes. Shao et al. (2011) introduced a regularization term to (2.19) and (2.20) to obtain the *twin bounded support vector machine* (TBSVM). An alternative name would be *improved twin support vector machine*. In the linear case, (2.19) and (2.20) are modified to

$$\begin{cases} \min\limits_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}} \ \dfrac{1}{2}\|X_1\boldsymbol{w}_1 + b_1\boldsymbol{e}_1\|^2 + C_1\boldsymbol{e}_2^T\boldsymbol{\xi} + \dfrac{C_3}{2}\left(\|\boldsymbol{w}_1\|^2 + b_1^2 \right) \\ \qquad \text{subject to} \\ X_2\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 \ge \boldsymbol{e}_2 - \boldsymbol{\xi}, \qquad \boldsymbol{\xi} \ge 0 \end{cases} \tag{2.45}$$

and

$$\begin{cases} \min\limits_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}} \ \frac{1}{2}\|X_2\boldsymbol{w}_2 + b_2\boldsymbol{e}_2\|^2 + C_2\boldsymbol{e}_1^T\boldsymbol{\eta} + \frac{C_4}{2}\left(\|\boldsymbol{w}_2\|^2 + b_2^2\right) \\ \quad \text{subject to} \\ X_1\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \geq 0. \end{cases} \tag{2.46}$$

The added terms $\frac{C_3}{2}\left(\|\boldsymbol{w}_1\|^2 + b_1^2\right)$ and $\frac{C_4}{2}\left(\|\boldsymbol{w}_2\|^2 + b_2^2\right)$ help maximize the margins.

We focus on the nonlinear version. Using notation (2.34) together with (2.32), then (2.35) and (2.36) become

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_1 \boldsymbol{\xi}} \ \frac{1}{2}\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\|^2 + C_1\langle \boldsymbol{e}_2, \boldsymbol{\xi}\rangle + \frac{C_3}{2}\tilde{\boldsymbol{u}}_1^T\tilde{X}\tilde{\boldsymbol{u}}_1 \\ \quad \text{subject to} \\ \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1 \leq 0, \qquad -\boldsymbol{\xi} \leq 0 \end{cases} \tag{2.47}$$

and

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_2 \boldsymbol{\eta}} \ \frac{1}{2}\|\tilde{X}_2\tilde{\boldsymbol{u}}_2\|^2 + C_2\langle \boldsymbol{e}_1, \boldsymbol{\eta}\rangle + \frac{C_4}{2}\tilde{\boldsymbol{u}}_2^T\tilde{X}\tilde{\boldsymbol{u}}_2 \\ \quad \text{subject to} \\ \boldsymbol{e}_1 - \boldsymbol{\eta} - \tilde{X}_1\tilde{\boldsymbol{u}}_2 \leq 0, \qquad -\boldsymbol{\eta} \leq 0 \end{cases} . \tag{2.48}$$

The Lagrangian of (2.47) looks like that of (2.35) with just one added term,

$$L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\left\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\right\|^2 + C_1\langle e_2, \boldsymbol{\xi}\rangle + \frac{C_3}{2}\tilde{\boldsymbol{u}}_1^T\tilde{X}\tilde{\boldsymbol{u}}_1 + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1\rangle + \langle \boldsymbol{\beta}, -\boldsymbol{\xi}\rangle$$

and after adding a small multiple of the identity matrix ,

$$\begin{aligned} L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\left\langle \left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_1\right\rangle - \left\langle \tilde{X}_2^T\boldsymbol{\alpha}, \tilde{\boldsymbol{u}}_1\right\rangle \\ + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2\rangle + \langle C_1e_2 - \boldsymbol{\alpha} - \boldsymbol{\beta}, \boldsymbol{\xi}\rangle. \end{aligned} \tag{2.49}$$

It vanishes when

$$0 = \nabla_{(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi})}(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{bmatrix} \left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 - \tilde{X}_2^T\boldsymbol{\alpha} \\ C_1e_2 - \boldsymbol{\alpha} - \boldsymbol{\beta} \end{bmatrix}.$$

The solutions are obtained just as in the twin SVM case: Setting

$$\begin{aligned} \Omega &= X_2\left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T \in M_q(\mathbb{R}) \qquad \text{and} \\ \Gamma &= X_1\left(\tilde{X}_2^T\tilde{X}_2 + C_4\tilde{X} + \delta I_{p+q+1}\right)^{-1}\tilde{X}_1^T \in M_p(\mathbb{R}) \end{aligned}$$

one solves the dual minimization problems (2.41) and (2.42) to obtain the solutions (2.39) and (2.43) for the respective hyperplanes,

$$\tilde{\boldsymbol{u}}_1 = \left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{\alpha}$$
$$\tilde{\boldsymbol{u}}_2 = \left(\tilde{X}_2^T\tilde{X}_2 + C_4\tilde{X} + \delta I_{p+q+1}\right)^{-1}\tilde{X}_1^T\boldsymbol{\gamma}$$

and the decision functions are still (2.33) in the normalized case, respectively (2.44) in the non-normalized case.

2.5 Least squares twin support vector machines

One may also modify the twin support vector machine to least square form. As with the regular support vector machine (SVM), the penalty term in the least squares twin support vector machine (LSTSVM) will be quadratic, and the constraints become equality constraints. Again, we only present the more general and difficult nonlinear case. The discussion parallels that of the twin support vector machine. Modifying (2.23) and (2.24) one obtains the problems

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_1},b_1,\xi_1,\ldots\xi_q} \sum_{i=1}^{p}\left|\langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_i)\rangle + b_1\right|^2 + \frac{C_1}{2}\sum_{j=1}^{q}\xi_j^2 \\ \quad\text{subject to} \\ \langle \boldsymbol{w}_1, \Phi(\boldsymbol{x}_{p+j})\rangle + b_1 = 1 - \xi_j \qquad (j = 1, \ldots, q) \end{cases} \tag{2.50}$$

and

$$\begin{cases} \displaystyle\min_{\boldsymbol{w_2},b_2,\eta_1,\ldots\eta_p} \sum_{i=p+1}^{p+q}\left|\langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_i)\rangle + b_2\right|^2 + \frac{C_2}{2}\sum_{j=1}^{p}\eta_j^2 \\ \quad\text{subject to} \\ \langle \boldsymbol{w}_2, \Phi(\boldsymbol{x}_j)\rangle + b_2 = 1 - \eta_j \qquad (j = 1, \ldots, p). \end{cases} \tag{2.51}$$

The equality constraints together with the penalty terms try to place the data of the “other” class proximal to a parallel hyperplane at margin -1. The decision function remains of the form (2.25). Expressed in matrix notation (compare with (2.29) and (2.30)),

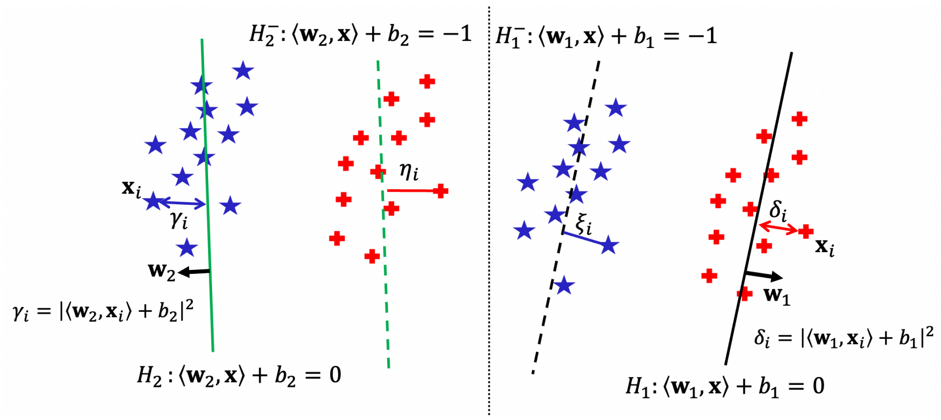

$H_2^-: \langle \mathbf{w}_2, \mathbf{x} \rangle + b_2 = -1$
$H_1^-: \langle \mathbf{w}_1, \mathbf{x} \rangle + b_1 = -1$
$\mathbf{x}_i$
γ_i
η_i
ξ_i
δ_i
$\mathbf{x}_i$
$\mathbf{w}_2$
$\mathbf{w}_1$
$\gamma_i = |\langle \mathbf{w}_2, \mathbf{x}_i \rangle + b_2|^2$
$\delta_i = |\langle \mathbf{w}_1, \mathbf{x}_i \rangle + b_1|^2$
$H_2: \langle \mathbf{w}_2, \mathbf{x} \rangle + b_2 = 0$
$H_1: \langle \mathbf{w}_1, \mathbf{x} \rangle + b_1 = 0$

Figure 2.6 A linear Least Squares Twin SVM.

the above optimization problems become

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}} \ \frac{1}{2}\|\tilde{X}_1 \tilde{\boldsymbol{u}}_1\|^2 + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 \\ \quad \text{subject to} \\ \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2 \tilde{\boldsymbol{u}}_1 = 0 \end{cases} \tag{2.52}$$

and

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_2, \boldsymbol{\eta}} \ \frac{1}{2}\|\tilde{X}_2 \tilde{\boldsymbol{u}}_2\|^2 + \frac{C_2}{2}\|\boldsymbol{\eta}\|^2 \\ \quad \text{subject to} \\ \boldsymbol{e}_1 - \boldsymbol{\eta} - \tilde{X}_1 \tilde{\boldsymbol{u}}_2 = 0. \end{cases} \tag{2.53}$$

The Lagrangian of (2.52) is

$$\begin{aligned} L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}) &= \frac{1}{2}\left\|\tilde{X}_1 \tilde{\boldsymbol{u}}_1\right\|^2 + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2 \tilde{\boldsymbol{u}}_1 \rangle \\ &= \frac{1}{2}\left\langle \tilde{X}_1^T \tilde{X}_1 \tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_1 \right\rangle + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 - \left\langle \tilde{X}_2^T \boldsymbol{\alpha}, \tilde{\boldsymbol{u}}_1 \right\rangle + \langle \boldsymbol{\alpha}, \boldsymbol{e}_2 \rangle - \langle \boldsymbol{\alpha}, \boldsymbol{\xi} \rangle \end{aligned}$$

for $\boldsymbol{\alpha} \in \mathbb{R}^q$. The computations that follow are similar to the least square support vector machine (LSSVM). The gradient of the Lagrangian vanishes when

$$0 = \nabla_{(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi})}(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \begin{bmatrix} \tilde{X}_1^T \tilde{X}_1 \tilde{\boldsymbol{u}}_1 - \tilde{X}_2^T \boldsymbol{\alpha} \\ C_1 \boldsymbol{\xi} - \boldsymbol{\alpha} \end{bmatrix}.$$

which gives

$$\tilde{X}_1^T \tilde{X}_1 \tilde{\boldsymbol{u}}_1 = \tilde{X}_2^T \boldsymbol{\alpha} \qquad \text{and} \qquad \boldsymbol{\xi} = \frac{1}{C_1}\boldsymbol{\alpha}. \tag{2.54}$$

Substituting the second equation of (2.54) into the equality condition of (2.52) and multiplying by $\tilde{X}_2^T$one obtains

$$\tilde{X}_2^T\tilde{X}_2\tilde{\boldsymbol{u}}_1 + \frac{1}{C_1}\tilde{X}_2^T\boldsymbol{\alpha} = \tilde{X}_2^T\boldsymbol{e}_2.$$

Then the first equation in (2.54) then gives

$$\left(\tilde{X}_2^T\tilde{X}_2 + \frac{1}{C_1}\tilde{X}_1^T\tilde{X}_1\right)\tilde{\boldsymbol{u}}_1 = \tilde{X}_2^T\boldsymbol{e}_2.$$

Again, the above matrix my not be positive definite, hence adding a small multiple of the identity matrix one obtains

$$\tilde{\boldsymbol{u}}_1 = \left(\tilde{X}_2^T\tilde{X}_2 + \frac{1}{C_1}\tilde{X}_1^T\tilde{X}_1 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{e}_2. \tag{2.55}$$

Similarly, the solution of (2.53) is

$$\tilde{\boldsymbol{u}}_2 = \left(\tilde{X}_1^T\tilde{X}_1 + \frac{1}{C_2}\tilde{X}_2^T\tilde{X}_2 + \delta I_{p+q+1}\right)^{-1}\tilde{X}_1^T\boldsymbol{e}_1. \tag{2.56}$$

The decision function remains unchanged from (2.31), respectively (2.33).

2.5.1 Improved least squares twin support vector machine

In a way similar to that leading to the twin bounded support vector machine, Xu et al. (2012) added a regularization term to the least squares twin support vector machine (2.52) and (2.53) to obtain the *improved least squares twin support vector machine* (ILSTWM) (which was re-introduced as regularized least squares twin support vector machine by Ali et al. (2022)),

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi}} \dfrac{1}{2}\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\|^2 + \dfrac{C_1}{2}\|\boldsymbol{\xi}\|^2 + \dfrac{C_3}{2}\tilde{\boldsymbol{u}}_1^T\tilde{X}\tilde{\boldsymbol{u}}_1 \\ \qquad \text{subject to} \\ \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1 = 0 \end{cases} \tag{2.57}$$

and

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_2,\boldsymbol{\eta}} \dfrac{1}{2}\|\tilde{X}_2\tilde{\boldsymbol{u}}_2\|^2 + \dfrac{C_2}{2}\|\boldsymbol{\eta}\|^2 + \dfrac{C_4}{2}\tilde{\boldsymbol{u}}_2^T\tilde{X}\tilde{\boldsymbol{u}}_2 \\ \qquad \text{subject to} \\ \boldsymbol{e}_1 - \boldsymbol{\eta} - \tilde{X}_1\tilde{\boldsymbol{u}}_2 = 0. \end{cases} \tag{2.58}$$

Considering the definition of the vectors $\tilde{\boldsymbol{u}}_1$ and $\tilde{\boldsymbol{u}}_2$ in (2.34), this extra term again attempts to maximize the margin. The Lagrangian of (2.57) is

$$L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\left\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\right\|^2 + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 + \frac{C_3}{2}\tilde{\boldsymbol{u}}_1^T\tilde{X}\tilde{\boldsymbol{u}}_1 + \langle\boldsymbol{\alpha}, \boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1\rangle,$$

and after adding a small multiple of the identity matrix,

$$\begin{aligned} L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\left\langle\left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_1\right\rangle + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 \\ - \left\langle\tilde{X}_2^T\boldsymbol{\alpha}, \tilde{\boldsymbol{u}}_1\right\rangle + \langle\boldsymbol{\alpha}, \boldsymbol{e}_2\rangle - \langle\boldsymbol{\alpha}, \boldsymbol{\xi}\rangle \end{aligned}$$

for $\boldsymbol{\alpha} \in \mathbb{R}^q$. Its gradient vanishes when

$$0 = \nabla_{(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi})}L(\tilde{\boldsymbol{u}}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \begin{bmatrix}\left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 - \tilde{X}_2^T\boldsymbol{\alpha} \\ C_1\boldsymbol{\xi} - \boldsymbol{\alpha}\end{bmatrix}.$$

which gives

$$\left(\tilde{X}_1^T\tilde{X}_1 + C_3\tilde{X} + \delta I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 = \tilde{X}_2^T\boldsymbol{\alpha} \qquad \text{and} \qquad \boldsymbol{\xi} = \frac{1}{C_1}\boldsymbol{\alpha}. \tag{2.59}$$

Substituting the second equation of (2.59) into the equality condition of (2.57) and multiplying by $\tilde{X}_2^T$one obtains

$$\tilde{X}_2^T\tilde{X}_2\tilde{\boldsymbol{u}}_1 + \frac{1}{C_1}\tilde{X}_2^T\boldsymbol{\alpha} = \tilde{X}_2^T\boldsymbol{e}_2.$$

Then the first equation in (2.59) gives

$$\left(\tilde{X}_2^T\tilde{X}_2 + \frac{1}{C_1}\tilde{X}_1^T\tilde{X}_1 + \frac{C_3}{C_1}\tilde{X} + \frac{\delta}{C_1}I_{p+q+1}\right)\tilde{\boldsymbol{u}}_1 = \tilde{X}_2^T\boldsymbol{e}_2.$$

The matrix on the left is positive definite because we have added $\delta > 0$, and one obtains

$$\tilde{\boldsymbol{u}}_1 = \left(\tilde{X}_2^T\tilde{X}_2 + \frac{1}{C_1}\tilde{X}_1^T\tilde{X}_1 + \frac{C_3}{C_1}\tilde{X} + \frac{\delta}{C_1}I_{p+q+1}\right)^{-1}\tilde{X}_2^T\boldsymbol{e}_2. \tag{2.60}$$

Similarly, the solution of (2.58) is

$$\tilde{\boldsymbol{u}}_2 = \left(\tilde{X}_1^T\tilde{X}_1 + \frac{1}{C_2}\tilde{X}_2^T\tilde{X}_2 + \frac{C_4}{C_2}\tilde{X} + \frac{\delta}{C_2}I_{p+q+1}\right)^{-1}\tilde{X}_1^T\boldsymbol{e}_1. \tag{2.61}$$

The decision functions are still identical to (2.31) (or equivalently to (2.44)) in the non-normalized case and to (2.33) in the normalized case.

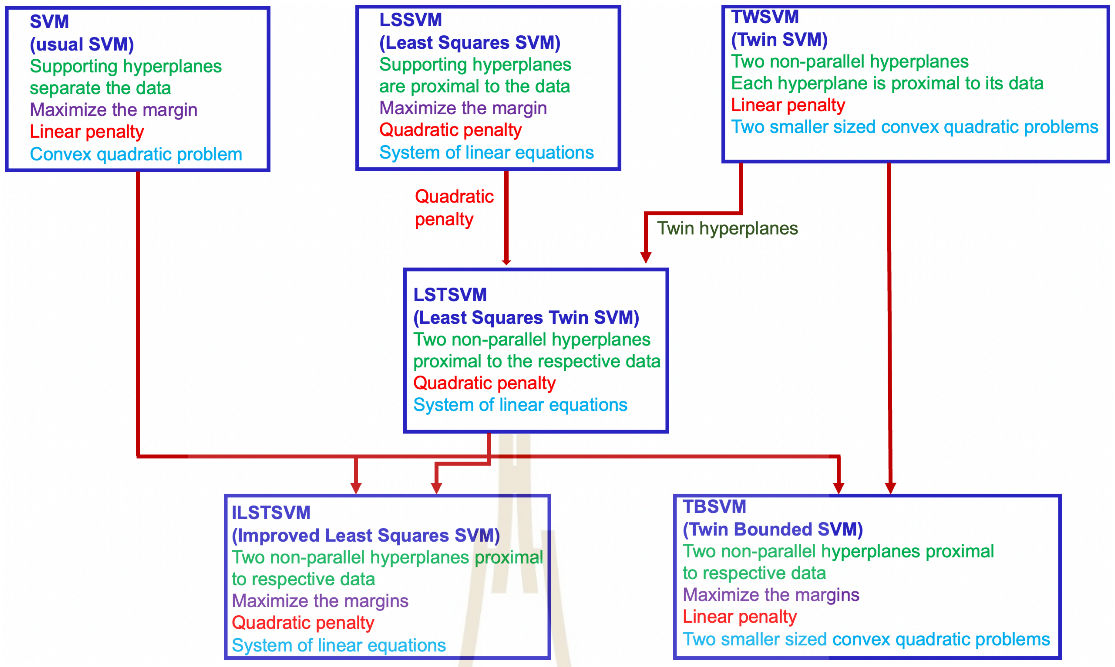

SVM
(usual SVM)
Supporting hyperplanes separate the data
Maximize the margin
Linear penalty
Convex quadratic problem
LSSVM
(Least Squares SVM)
Supporting hyperplanes are proximal to the data
Maximize the margin
Quadratic penalty
System of linear equations
TWSVM
(Twin SVM)
Two non-parallel hyperplanes
Each hyperplane is proximal to its data
Linear penalty
Two smaller sized convex quadratic problems
Quadratic penalty
Twin hyperplanes
LSTSVM
(Least Squares Twin SVM)
Two non-parallel hyperplanes proximal to the respective data
Quadratic penalty
System of linear equations
ILSTSVM
(Improved Least Squares SVM)
Two non-parallel hyperplanes proximal to respective data
Maximize the margins
Quadratic penalty
System of linear equations
TBSVM
(Twin Bounded SVM)
Two non-parallel hyperplanes proximal to respective data
Maximize the margins
Linear penalty
Two smaller sized convex quadratic problems

Figure 2.7 Flowchart of the SVM variations.

2.5.2 Robust energy-based least squares twin support vector machine

Tanveer et al. (2016) have added a parameter to the constraints of the improved LSTSVM to obtain the *Robust energy-based least squares twin support vector machine* (RELS-TSVM):

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_1,\boldsymbol{\xi}} \ \frac{1}{2}\|\tilde{X}_1\tilde{\boldsymbol{u}}_1\|^2 + \frac{C_1}{2}\|\boldsymbol{\xi}\|^2 + \frac{C_3}{2}\tilde{\boldsymbol{u}}_1^T\tilde{X}\tilde{\boldsymbol{u}}_1 \\ \quad \text{subject to} \\ E_1\boldsymbol{e}_2 - \boldsymbol{\xi} - \tilde{X}_2\tilde{\boldsymbol{u}}_1 = 0 \end{cases} \tag{2.62}$$

and

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{u}}_2,\boldsymbol{\eta}} \ \frac{1}{2}\|\tilde{X}_2\tilde{\boldsymbol{u}}_2\|^2 + \frac{C_2}{2}\|\boldsymbol{\eta}\|^2 + \frac{C_4}{2}\tilde{\boldsymbol{u}}_2^T\tilde{X}\tilde{\boldsymbol{u}}_2 \\ \quad \text{subject to} \\ E_2\boldsymbol{e}_1 - \boldsymbol{\eta} - \tilde{X}_1\tilde{\boldsymbol{u}}_2 = 0. \end{cases} \tag{2.63}$$

The additional scaling parameters E_1 and E_2 will place the data of the "other" class proximal to a parallel hyperplane at margin $-E_1$, respectively $-E_2$ instead of -1. The solution steps are essentially identical to the improved least squares twin support vector machine, and give

$$\tilde{\boldsymbol{u}}_1 = \left(\tilde{X}_2^T \tilde{X}_2 + \frac{1}{C_1} \tilde{X}_1^T \tilde{X}_1 + \frac{C_3}{C_1} \tilde{X} + \frac{\delta}{C_1} I_{p+q+1} \right)^{-1} \tilde{X}_2^T E_1 \boldsymbol{e}_2. \tag{2.64}$$

Similarly, the solution of (2.58) is

$$\tilde{\boldsymbol{u}}_2 = \left(\tilde{X}_1^T \tilde{X}_1 + \frac{1}{C_2} \tilde{X}_2^T \tilde{X}_2 + \frac{C_4}{C_2} \tilde{X} + \frac{\delta}{C_2} I_{p+q+1} \right)^{-1} \tilde{X}_1^T E_2 \boldsymbol{e}_1. \tag{2.65}$$

2.5.3 Analysis of the RELS-TVSM

At a first look, it appears that introduction of the parameters E_1 and E_2 gives two more parameters to fine tune the twin SVM. This is, however, not the case. The parameters E_1 and E_2 simply lead to a scaling of the hyperplane vectors $\tilde{\boldsymbol{u}}_1$ and $\tilde{\boldsymbol{u}}_2$ by these parameter values, as can be seen by comparing the pairs of equations (2.60) and (2.64), respectively (2.61) and (2.65).

Thus, the normalized decision function (2.33) remains unchanged: That is, when using the normalized decision function then the RELS-TVSM coincides with the improved least squares support vector machine and the two parameters E_1 and E_2 are superfluous.

On the other hand, when using the non-normalized decision function, there is one change. Write the decision function (2.31) (i.e. (2.44)) in the form

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if} \quad \dfrac{|\langle \tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{K}}(\boldsymbol{x}) \rangle|}{|\langle \tilde{\boldsymbol{u}}_2, \tilde{\boldsymbol{K}}(\boldsymbol{x}) \rangle|} \leq 1 \\ -1 & \text{else} \end{cases} \tag{2.66}$$

it is obvious that its outcome depends on the ratio $\frac{E_1}{E_2}$. That is, one may choose $E_2 = 1$ and there will be only one additional parameter E_1 that needs to be optimized in training.

2.6 Performance evaluation

There are various metrics available to evaluate the performance of machine learning classification models. We specifically focus on the metrics based on the confusion matrix. For a two-class problem, the confusion matrix is given in Table 2.1, where

- TP (true positive) is the number of positive data classified as positive.

- TN (true negative) is the number of negative data classified as negative.
- FP (false positive) is the number of misclassified negative data.
- FN (false negative) is the number of misclassified positive data.

Table 2.1 Confusion matrix.

	Predicted negative	Predicted positive
Actual negative	TN	FP
Actual positive	FN	TP

For the evaluation of classifier performance, some of the commonly used measures based on the above confusion matrix are:

1. Accuracy, also known as a recognition rate, is the ratio of the total number of correctly classified samples to the total number of input samples.

$$\text{Accuracy/Recognition rate} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.67}$$

2. Sensitivity, also known as recall or true positive rate, is ratio of the number of correctly classified positive samples to the total number of positive samples.

$$\text{Sensitivity/Recall} = \frac{TP}{TP+FN} \tag{2.68}$$

3. Specificity or true negative rate is defined as the ratio of the number of correctly classified negative samples to the toal number of negative samples.

$$\text{Specificity} = \frac{TN}{TN+FP} \tag{2.69}$$

4. Precision, also called positive predictive value, is the ratio of the number of true positive predictions to the total number of positive predicted samples.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.70}$$

Precision and recall are the most commonly used metrics in diagnosis of medical problems since misclassification of an unhealthy sample as a normal sample is fatal as compared to healthy sample being classified as disease-free.

CHAPTER III

MULTICLASS SUPPORT VECTOR MACHINES AND DECISION STRATEGIES

In this chapter we first review the various multiclass support vector machines with regards to decision strategies. We then compare the performance of the various variations of multiclass support vector machines and decision strategies, using our own results and those published in the literature. We finally propose a new strategy which we call *minimum average distance* for multiclass decisions and compare its performance with the common strategies.

3.1 Multiclass support vector machines

In real world scenarios one often deals with data that is separated into multiple classes. Because support vector machines perform binary classification, one usually decomposes a multiclass decision problem into an ensemble of binary decision problems. There are several ways of decision making in multiclass problems by combining the binary decisions, namely:

1. One versus all,
2. All versus one,
3. One versus one,
4. Directed acyclic graph.

For the rest of this section, we consider an $N-$class problem whose training data set is given by

$$\Gamma = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$, $y_k \in \{1, 2, \ldots, N\}$, $k = 1, 2, \ldots, l$. Then the problem is to obtain a decision function $f(\boldsymbol{x}) : \mathbb{R}^n \to \{1, 2, \ldots, N\}$ that can predict an appropriate label y for an input $\boldsymbol{x}$:

$$f(\boldsymbol{x}_k) = y_k$$

for all training data.

Much of the presentation in this section is based on (Abe, 2010). For each i, let D_i denote the training data belonging to the i-th class, that is,

$$D_i = \{\boldsymbol{x} : (\boldsymbol{x}, i) \in \Gamma\}, \qquad i = 1, \ldots, N.$$

Throughout we assume that all support vector machines involved in a multiclass problem use the same kernels and the same parameter values. This means in particular that the map Φ is identical throughout.

3.1.1 One versus all (OVA)

This method is also called “one against the rest”. For each of the data classes, one trains a support vector machine separating the given class from the union of the remaining classes. One thus obtains N support vector machines with N decision functions $f_i(\boldsymbol{x})$ so that

$$f_i(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \text{ is in class } i \\ -1 & \text{if } \boldsymbol{x} \text{ is not in class } i, \end{cases}$$

$(i = 1, \ldots, N)$. Now if $f_i(\boldsymbol{x}) = 1$ for a unique i, then this will be the class to which $\boldsymbol{x}$ belongs. However, it may happen that $f_i(x) = 1$ for more than one i, or that $f_i(\boldsymbol{x}) = -1$ for all i, so that $\boldsymbol{x}$ becomes unclassifiable.

This can be overcome, though, by looking at “continuous” decision values.

1. (The case of single hyperplane SVMs) Recall that the equations

$$g_i(\boldsymbol{x}) = \langle \boldsymbol{w}_i, \Phi(\boldsymbol{x}) \rangle + b_i$$

are a signed measure of the distance of the point $\Phi(\boldsymbol{x})$ from the separating hyperplane $g_i(\boldsymbol{x}) = 0$. One may thus interpret its value as the likelihood of $\boldsymbol{x}$ belonging to class i. Thus, the input data $\boldsymbol{x}$ is assigned the class label

$$i_o = \arg \max_{i=1,\dots,N} g_i(\boldsymbol{x})$$

2. (The case of twin hyperplane SVMs) Recall that each of the N twin support vector machines has two hyperplanes. Modifying the notation (2.25), then

$$g_i^+(\boldsymbol{x}) = \left|\langle \boldsymbol{w}_i^+, \Phi(\boldsymbol{x})\rangle + b_i^+\right|$$

determines the distance of $\Phi(\boldsymbol{x})$ from the hyperplane of class i, while

$$g_i^-(\boldsymbol{x}) = \left|\langle \boldsymbol{w}_i^-, \Phi(\boldsymbol{x})\rangle + b_i^-\right|$$

determines the distance of $\Phi(\boldsymbol{x})$ from the hyperplane of the remaining classes. The input data $\boldsymbol{x}$ is assigned the class label

$$i_o = \arg \min_{i=1,\dots,N} g_i^+(\boldsymbol{x}),$$

to select the class of the hyperplane to which $\boldsymbol{x}$ is closest. In the spirit of (2.26) one may also choose

$$g_i^+(\boldsymbol{x}) = \frac{\left|\langle \boldsymbol{w}_i^+, \Phi(\boldsymbol{x})\rangle + b_i^+\right|}{\|\boldsymbol{w}_i^+\|}$$

representing the actual distance.

3.1.2 All versus one (AVO)

This strategy only applies to twin hyperplane SVMs. In contrast to the one-versus-all strategy, one selects the class where the vector $\Phi(\boldsymbol{x})$ is farthest away from the "other" hyperplane: Thus, the input data $\boldsymbol{x}$ is assigned the class label

$$i_o = \arg \max_{i=1,\dots,N} g_i^-(\boldsymbol{x}) = \arg \max_{i=1,\dots,N} \left|\langle \boldsymbol{w}_i^-, \Phi(\boldsymbol{x})\rangle + b_i^-\right|.$$

Alternatively using real distances,

$$i_o = \arg \max_{i=1,\dots,N} \frac{\left|\langle \boldsymbol{w}_i^-, \Phi(\boldsymbol{x})\rangle + b_i^-\right|}{\|w_i^-\|}.$$

3.1.3 One versus one (OVO)

Here one determines the decision functions for all combinations of class pairs. Let the decision function for class i against class j be denoted by $f_{ij}(\boldsymbol{x})$ so that

$$f_{ij}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \text{ is in class } i \\ -1 & \text{if } \boldsymbol{x} \text{ is in class } j, \end{cases}$$

for $j \neq i$. Then clearly, $f_{ji}(\boldsymbol{x}) = -f_{ij}(\boldsymbol{x})$. Hence, for an N-class problem, $\dfrac{N(N-1)}{2}$ decision functions are required.

Consider the regions

$$R_i = \{\boldsymbol{x} \in \mathbb{R}^n \mid f_{ij}(\boldsymbol{x}) = 1,\ j = 1, \ldots, N, j \neq i\}, \qquad i = 1, \ldots, N$$

That is, for each i, R_i is the region of data where the i-th machine wins over all remaining machines. Clearly, these regions are disjoint, and if $\boldsymbol{x} \in R_i$ then $\boldsymbol{x} \in$ Class i.

However, these regions do not cover the whole space $\mathbb{R}^n$. It may thus be possible that some data point $\boldsymbol{x}$ does not lie in any of these regions, hence is unclassifiable. One therefore modifies the strategy: instead of choosing the class that "always wins" one chooses the class that "wins most often": For each i, set

$$f_i(\boldsymbol{x}) = \sum_{j=1, j\neq i}^{N} \frac{\text{sgn}(f_{ij}(\boldsymbol{x})) + 1}{2},$$

where here

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0. \end{cases}$$

The function $f_i(\boldsymbol{x})$ indicates how many times machine i "wins" over the other machines. Then $\boldsymbol{x}$ gets the class label

$$i_o = \arg \max_{i=1,\ldots,N} f_i(\boldsymbol{x})$$

This is also called the *maximum vote* strategy. Observe that the maximizer i_o need not be unique. In this case, one can assign the sample point $\boldsymbol{x}$ to any maximizing i_o.

3.1.4 Directed acyclic graph (DAG)

This is a modification of the one-versus-one strategy. The decisions are made moving along a directed acyclic graph. This is best explained by the example of Figure 3.1. Suppose, there are $N = 4$ classes. Each vertex represents one of the 6 decision functions f_{ij}. Decisions are made following from the root node to one of the end nodes, and requires only 3 decisions instead of 6.

In general, one still needs to train $\frac{N(N-1)}{2}$ machines. However, only $N-1$ decisions are required.

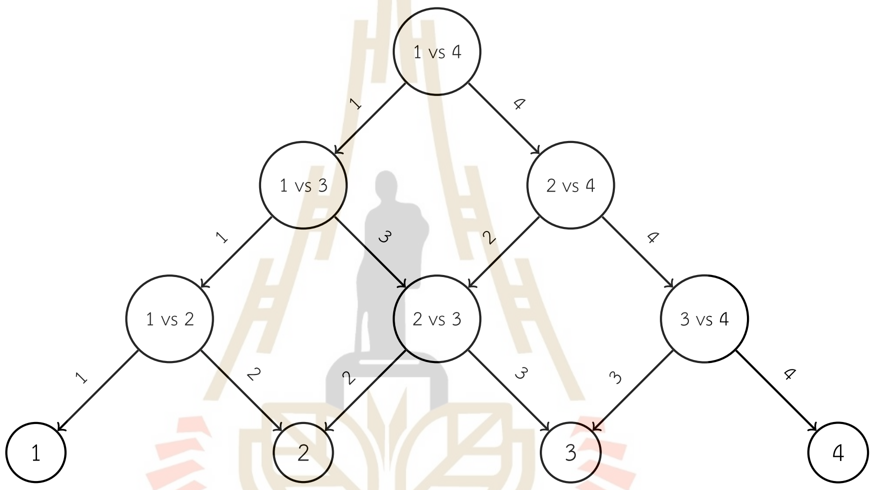


Figure 3.1 DAG decision making ($N = 4$)

3.2 Reviewing performance results from the literature

There is a bulk of literature introducing variations of support vector machines and evaluating their performances by numerical experiments for multiclass data. One difficulty in assessing the best suited type is that the published performance data may use different datasets and thus the results are not comparable.

Tomar and Agarwal (2015) and Ali et al. (2022) present performance comparisons of various support vector machine variations for multiclass data, under different multiclass strategies. They are using highly unbalanced datasets downloadable from the UCI database. Their results seem to show that overall, the least square SVM based models exhibit better performance than the usual SVM or twin SVM based models, and in some cases achieve 100% accuracy. One problem in their presentation, however, is that in the case of the usual SVM / twin SVM models they only consider the basic models, and do not explain the multiclass decision strategies.

We have run their tests for some of the datasets, and our results do not confirm theirs in many instances. In particular, our computations never achieved 100% accuracy as claimed by the authors for some datasets, and our results do not allow the conclusion that least square SVM models give better accuracy in general. The following table shows a comparison of our results with those presented in Ali et al. (2022). All models are nonlinear using the RBF kernel

$$K(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right) \qquad \text{(respectively} \qquad K(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|^2\right)).$$

In each case, initially parameter optimization was performed by grid search (parameters C_i for the model, kernel parameter σ). Then ten runs of ten-fold cross validation were performed using the optimized parameter values. Table 3.1 lists the average accuracies and standard deviations obtained in our computations over the ten runs. The accuracy values correspond to the “accuracy-1” of Chapter 7, and all results listed are percentages. Computations were performed with Matlab R2019b on a Macbook Air notebook, Intel I5 1.8GHz CPU and 8GB Ram.

The following SVMs are included:

- SVM: The usual support vector machine as outlined in section 2.2.
- TWSVM: Twin support vector machine as outlined in section 2.4
- TBSVM: Twin bounded support vector machine as outlined in subsection 2.4.4
- LSTSVM: Least squares twin support vector machine as outlined in section 2.5
- ILSTSVM: Improved least squares twin support vector machine as outlined in subsection 2.5.1

Ali et al. (2022) have published results for only a subset of these SVMs, the missing data are labeled by ‘n/a’ in the table.

3.2.1 Discussion

Balance dataset: The one-versus-one decision strategies come out on top, with accuracies above 99.5% in case of the regular SVM, twin SVM and twin bounded SVM. The least-squares SVMs tend to have lower accuracies. It is noticeable that in our test with the regular SVM, using the libsvm library, an accuracy of 99.8% could be achieved, whereas Tomar and Agarwal (2015) and Ali et al. (2022) show an accuracy of 84.59% only. Unfortunately, the authors do not explain what type of one-vs-one decision strategy they apply. The results presented by these authors suggest that least-squares based twin SVMs show substantially better performance, which is contradicted by our results.

Hayes-Roth dataset: The accuracies of all SVM types and all decision strategies are in similar ranges: 84.48%–87.34%, with the twin bounded SVM giving the best accuracy. Again, our results differ substantially from those by Ali et al. (2022). They claim to have achieved an accuracy of 100% using the improved least squares twin SVM and at all decision strategies.

Ecoli dataset: The accuracies of all SVM types and all decision strategies are in similar ranges: 85.81%–89.91%, with the twin bounded SVM giving the best accuracy. Interestingly, the all-versus-one strategy gave best results. Overall, our results are somehow better than those by Ali et al. (2022).

Glass dataset: The one-versus-one strategy appears to be best for all SVMs. This is a dataset where the twin support SVMs perform noticeably better than the regular SVM. The best accuracy of 74.55% was achieved with the twin bounded SVM. Ali et al. (2022) claim to have achieved a performance of up to 98.47% accuracy. This is in contradiction to all published results known to us and unlikely to be true. The Glass dataset is highly unbalanced, and there are no publications showing accuracies in the upper 80% range or higher.

Iris dataset: The accuracies of all SVM types and all decision strategies are in similar ranges: 97.6%–98.67%, with the twin bounded SVM giving the best accuracy. Overall, our results are in alignment with those by Ali et al. (2022).

Table 3.1 Performance comparison of multiclass SVMs and twin SVM variants.

Dataset	SVM	SVM	TWSVM	TWSVM	TWSVM	TWSVM
	(OVA)	(OVO)	(OVA)	(AVO)	(OVO)	(DAG)
	(C, γ)	(C, γ)	$(C_1 = C_2, \sigma)$	$(C_1 = C_2, \sigma)$	$(C_1 = C_2, \sigma)$	$(C_1 = C_2, \sigma)$
	acc$\pm$std(%)	acc$\pm$std(%)	acc$\pm$std(%)	acc$\pm$std(%)	acc$\pm$std(%)	acc$\pm$std(%)
Balance	$2^9, 2^{-1}$	$2^1, 2^{-7}$	$2^5, 2^4$	$2^4, 2^4$	$2^5, 2^1$	$2^5, 2^1$
	99.26 ± 0.97	99.98 ± 0.16	96.91 ± 1.93	97.15 ± 2.12	99.97 ± 0.7	99.78 ± 0.64
		$10^{-2}, 2^4$		$10^{-6}, 2^2$		
(Ali et al.)	n/a	84.59 ± 6.67	n/a	89.22 ± 5.88	n/a	n/a
Hayes-Roth	$2^{12}, 2^{-7}$	$2^5, 2^{-5}$	$2^2, 2^8$	$2^{-2}, 2^4$	$2^2, 2^3$	$2^2, 2^3$
	86.34 ± 6.57	86.27 ± 8.58	86.19 ± 8.92	86.32 ± 7.75	86.6 ± 7.7	85.92 ± 9.21
		$10^{-3}, 2^4$		$10^{-4}, 2^4$		
(Ali et al.)	n/a	72.30 ± 4.08	n/a	73.35 ± 3.26	n/a	n/a
Ecoli	$2^3, 2^0$	$2^0, 2^4$	$2^{-1}, 2^7$	$2^1, 2^7$	$2^{-3}, 2^5$	$2^{-3}, 2^5$
	89.14 ± 5.04	89.4 ± 4.27	88.84 ± 5.14	89.17 ± 4.83	89.18 ± 4.31	89.24 ± 5
		$10^{-6}, 2^1$		$10^{-6}, 2^7$		
(Ali et al.)	n/a	83.35 ± 6.28	n/a	82.01 ± 5.17	n/a	n/a
Glass	$2^{15}, 2^{-1}$	$2^9, 2^1$	$2^{-4}, 2^1$	$2^{-13}, 2^{-2}$	$2^{-4}, 2^0$	$2^{-4}, 2^0$
	71.96 ± 10.56	71.49 ± 10.32	70.09 ± 9.44	69.29 ± 9.32	72.79 ± 8.2	73.78 ± 7.83
		$10^{-4}, 2^0$		$10^{-6}, 2^1$		
(Ali et al.)	n/a	71.03 ± 6.02	n/a	69.52 ± 5.15	n/a	n/a
Iris	$2^1, 2^{-3}$	$2^8, 2^{-10}$	$2^{-1}, 2^2$	$2^1, 2^5$	$2^{-2}, 2^6$	$2^{-2}, 2^6$
	98.27 ± 3.37	98.67 ± 2.68	97.73 ± 3.45	97.8 ± 3.29	98.27 ± 3.23	98.27 ± 3.23
		$10^0, 2^2$		$10^0, 2^1$		
(Ali et al.)	n/a	97.33	n/a	98.00 ± 2.26	n/a	n/a

Table 3.1 (Continued).

Dataset	TBSVM (OVA) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	TBSVM (AVO) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	TBSVM (OVO) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	LSTSVM (OVA) $(C_1 = C_2, \sigma)$ acc±std(%)	LSTSVM (AVO) $(C_1 = C_2, \sigma)$ acc±std(%)	LSTSVM (OVO) $(C_1 = C_2, \sigma)$ acc±std(%)	LSTSVM (DAG) $(C_1 = C_2, \sigma)$ acc±std(%)	ILSTSVM (OVA) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	ILSTSVM (AVO) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	ILSTSVM (OVO) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)	ILSTSVM (DAG) $(C_1 = C_2, C_3 = C_4, \sigma)$ acc±std(%)
Balance	$2^{-14}, 2^{-18}, 2^{-1}$	$2^{3}, 2^{-18}, 2^{4}$	$2^{4}, 2^{-13}, 2^{1}$	$2^{-15}, 2^{1}$	$2^{-6}, 2^{-1}$	$2^{3}, 2^{2}$	$2^{3}, 2^{-2}$	$2^{-17}, 2^{-17}, 2^{-1}$	$2^{-8}, 2^{-15}, 2^{0}$	$2^{4}, 2^{-12}, 2^{-1}$	$2^{4}, 2^{-12}, 2^{-1}$
	97.78 ± 1.79	96.08 ± 2.61	99.5 ± 0.96	97.6 ± 1.64	91.76 ± 0.95	96.37 ± 2.16	96.59 ± 1.92	96.82 ± 2.23	92.85 ± 2.61	97.33 ± 1.9	97.12 ± 1.87
				$10^{-7}, 2^{3}$	$10^{-7}, 2^{1}$	$10^{1}, 2^{-2}$	$10^{-7}, 2^{3}$	$10^{-5}, 10^{-3}, 2^{3}$	$10^{1}, 10^{-3}, 2^{1}$	$10^{-2}, 10^{-4}, 2^{-2}$	$10^{-2}, 10^{-4}, 2^{3}$
(Ali et al.)	n/a	n/a	n/a	93.75 ± 5.15	86.89 ± 4.02	88.21 ± 3.2	95.05 ± 4.4	95.68 ± 2.93	92 ± 0.76	98.56 ± 0.90	98.57 ± 2.04
Hayes-Roth	$2^{4}, 2^{0}, 2^{1}$	$2^{-2}, 2^{-5}, 2^{2}$	$2^{10}, 2^{-7}, 2^{3}$	$2^{0}, 2^{2}$	$2^{-1}, 2^{4}$	$2^{3}, 2^{6}$	$2^{3}, 2^{6}$	$2^{1}, 2^{-8}, 2^{1}$	$2^{-1}, 2^{-6}, 2^{0}$	$2^{5}, 2^{-9}, 2^{6}$	$2^{5}, 2^{-9}, 2^{6}$
	86.14 ± 8.23	86.37 ± 8.52	87.34 ± 9.23	84.46 ± 8.48	86.25 ± 8.14	86.4 ± 8.9	85.59 ± 9.27	85.92 ± 7.26	86.05 ± 7.46	86.9 ± 9.54	85.81 ± 8.78
				$10^{-3}, 2^{3}$	$10^{-4}, 2^{5}$	$10^{-4}, 2^{4}$	$10^{-4}, 2^{4}$	$10^{-3}, 10^{-4}, 2^{3}$	$10^{-4}, 10^{-5}, 2^{5}$	$10^{-4}, 10^{-5}, 2^{4}$	$10^{-4}, 10^{-5}, 2^{4}$
(Ali et al.)	n/a	n/a	n/a	71.43 ± 3.08	72.86 ± 2.53	76.38 ± 3.63	79.92 ± 3.05	100 ± 0	100 ± 0	100 ± 0	100 ± 0
Ecoli	$2^{-1}, 2^{-14}, 2^{7}$	$2^{0}, 2^{-16}, 2^{3}$	$2^{-2}, 2^{-14}, 2^{6}$	$2^{-4}, 2^{1}$	$2^{-2}, 2^{-2}$	$2^{-2}, 2^{-2}$	$2^{-2}, 2^{-2}$	$2^{0}, 2^{1}, 2^{-1}$	$2^{0}, 2^{1}, 2^{-1}$	$2^{-3}, 2^{-8}, 2^{2}$	$2^{-3}, 2^{-8}, 2^{2}$
	89.16 ± 5.35	89.91 ± 4.64	89.61 ± 4.9	86.97 ± 5.33	87.6 ± 5.59	87.22 ± 4.58	87.13 ± 4.96	89.88 ± 4.79	89.91 ± 4.84	89.67 ± 5.06	85.81 ± 8.78
				$10^{-6}, 2^{1}$	$10^{-5}, 2^{1}$	$10^{-5}, 2^{1}$	$10^{-5}, 2^{1}$	$10^{0}, 10^{-3}, 2^{1}$	$10^{0}, 10^{-3}, 2^{1}$	$10^{0}, 10^{-2}, 2^{1}$	$10^{-5}, 10^{-2}, 2^{1}$
(Ali et al.)	n/a	n/a	n/a	84.38 ± 6.26	82.67 ± 5.35	74.78 ± 5.56	76.13 ± 4.77	85.13 ± 4.80	88.67 ± 6.28	88.99 ± 4.95	88 ± 4.78
Glass	$2^{0}, 2^{-1}, 2^{-1}$	$2^{-2}, 2^{-12}, 2^{0}$	$2^{-4}, 2^{-9}, 2^{0}$	$2^{-1}, 2^{1}$	$2^{-4}, 2^{0}$	$2^{-3}, 2^{1}$	$2^{-3}, 2^{1}$	$2^{-4}, 2^{-8}, 2^{0}$	$2^{4}, 2^{-4}, 2^{0}$	$2^{-3}, 2^{-8}, 2^{0}$	$2^{-3}, 2^{-8}, 2^{0}$
	$72.73 \pm 8, 47$	71.6 ± 9.29	74.55 ± 9	71.62 ± 7.87	72.49 ± 9.2	73 ± 9.97	74.16 ± 6.8	73.59 ± 8.42	73.5 ± 8.9	74.07 ± 8.7	74.16 ± 8.79
				$10^{-7}, 2^{0}$	$10^{-5}, 2^{1}$	$10^{-5}, 2^{7}$	$10^{-7}, 2^{7}$	$10^{-4}, 10^{-5}, 2^{0}$	$10^{-2}, 10^{-3}, 2^{1}$	$10^{-2}, 10^{-3}, 2^{7}$	$10^{-2}, 10^{-3}, 2^{7}$
(Ali et al.)	n/a	n/a	n/a	86.85 ± 5.32	94.82 ± 4.61	92.56 ± 5.3	95.89 ± 3.71	96.95 ± 4.04	98.47 ± 2.51	98.01 ± 3.39	76.63 ± 2.07
Iris	$2^{-1}, 2^{-5}, 2^{3}$	$2^{-1}, 2^{-2}, 2^{2}$	$2^{0}, 2^{-6}, 2^{5}$	$2^{2}, 2^{2}$	$2^{-12}, 2^{5}$	$2^{3}, 2^{8}$	$2^{3}, 2^{8}$	$2^{-1}, 2^{-6}, 2^{2}$	$2^{-12}, 2^{-12}, 2^{5}$	$2^{-1}, 2^{-7}, 2^{4}$	$2^{-1}, 2^{-7}, 2^{4}$
	97.8 ± 3.29	98.4 ± 3.01	98.67 ± 2.84	97.67 ± 3.83	97.6 ± 4.08	98.47 ± 2.82	98.4 ± 3.01	98 ± 3.35	97.73 ± 3.93	98 ± 3.48	98 ± 3.48
				$10^{-1}, 2^{3}$	$10^{0}, 2^{2}$	$10^{-2}, 2^{3}$	$10^{0}, 2^{3}$	$10^{-1}, 10^{-1}, 2^{3}$	$10^{0}, 10^{-1}, 2^{2}$	$10^{1}, 10^{-2}, 2^{3}$	$10^{0}, 10^{0}, 2^{3}$
(Ali et al.)	n/a	n/a	n/a	97.75 ± 2.57	96.45 ± 2.26	96.82 ± 2.03	98.21 ± 1.83	97.90 ± 4.76	98.12 ± 3.09	98.08 ± 3.10	98.30 ± 3.78

We have tried to contact the authors of both papers to clarify the reasons for the discrepancies, but have not received an answer.

Decision strategy:

The one-versus-one decisions tend to show the best accuracies. The exception is the Ecoli dataset, where the all-versus-one strategy gives the best accuracy, if only a small margin. The DAG variation of the one-versus-one strategy shows accuracies that are comparable with the regular one-versus-one strategy.

Single hyperplane SVM versus twin hyperplanes SVM: Both types of support vector machines show similar accuracies. The exception is the Glass dataset, where the twin SVM has an edge in accuracy of 1.3%.

Improved versus non-improved twin SVMs: Introducing the additional regularization parameters C_3 and C_4 increases the accuracy in the great majority of all cases, by up to 2.3%.

Twin versus least-squares twin SVMs: The least squares versions of the twin SVM tend to show lower accuracies for all datasets with the exception of the Ecoli dataset where the improved least squares twin SVM can match the twin bounded SVM.

3.3 A new strategy for multiclass twin support vector machines – decision by minimum average distance

As indicated in subsection 3.1.3, in one-versus-one classification one uses the *maximum vote strategy*. In the case of twin support vector machines, decisions are arrived at by converting continuous values, namely distances, into discrete decisions: ± 1. We therefore propose an alternative decision process which takes the continuous values into consideration, called the *average minimum distance* strategy.

We recall the maximum vote algorithm for the twin support vector machines. First consider the decision between any one pair of classes, class i and class j $(i \neq j)$. Modifying the notation (2.25), the decision is made by considering hyperplanes

$$g_{ij}(\boldsymbol{x}) = \langle \boldsymbol{w}_{ij}, \Phi(\boldsymbol{x}) \rangle + b_{ij}$$

and

$$g_{ji}(\boldsymbol{x}) = \langle \boldsymbol{w}_{ji}, \Phi(\boldsymbol{x})\rangle + b_{ji},$$

and the decision i-versus-j is determined by

$$f_{ij}(\boldsymbol{x}) = -f_{ji}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |g_{ij}(\boldsymbol{x})| \le |g_{ji}(\boldsymbol{x})| \\ -1 & \text{else.} \end{cases}$$

Then the number of votes in favour of the i-th class is

$$f_i(\boldsymbol{x}) = \sum_{j=1, j\neq i}^{N} \frac{\operatorname{sgn}(f_{ij}(\boldsymbol{x}))+1}{2}, \qquad (i = 1, \ldots, N)$$

and the winning class is that of the maximal number of votes,

$$i_o = \arg\max_{i=1,\ldots,N} f_i(\boldsymbol{x}). \tag{3.1}$$

When i_o is not be unique, then a "tie-breaker" is needed.

3.3.1 The minimum average distance vote

Next we describe the new minimum average distance vote. Recall that

$$|g_{ij}(\boldsymbol{x})| = |\langle \boldsymbol{w}_{ij}, \Phi(\boldsymbol{x})\rangle + b_{ij}|$$

is the unnormalized distance of vector $\Phi(\boldsymbol{x})$ from the hyperplane $g_{ij}(\boldsymbol{x}) = 0$. Alternatively, one may work with the real distances,

$$|g_{ij}(\boldsymbol{x})| = \frac{|\langle \boldsymbol{w}_{ij}, \Phi(\boldsymbol{x})\rangle + b_{ij}|}{\|\boldsymbol{w}_{ij}\|}.$$

We therefore set

$$f_i(\boldsymbol{x}) = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} |g_{ij}(\boldsymbol{x})|, \qquad (i = 1, \ldots, N)$$

which is the average distance of $\Phi(\boldsymbol{x})$ from the class i hyperplanes in all the i-versus-j twin SVMs. Then $\boldsymbol{x}$ is assigned to the class which has smallest average distance,

$$i_o = \arg\min_{i=1,\ldots,N} f_i(\boldsymbol{x}),$$

A comparison of accuracy between this method and other one-versus-one decisions is given in Table 3.2. It shows that for three out of the five data sets, the minimum average distance vote gives better accuracy. In case of the Balance dataset, 100% accuracy could be achieved.

Table 3.2 Comparisons between maximum-vote (mv) and minimum average distance (md) in one-versus-one multiclass decisions.

Dataset	TWSVM (OVO-mv) $(C_1 = C_2, \sigma)$ acc±std(%)	TWSVM (OVO-md) $(C_1 = C_2, \sigma)$ acc±std(%)	LSTSVM (OVO-mv) $(C_1 = C_2, \sigma)$ acc±std(%)	LSTSVM (OVO-md) $(C_1 = C_2, \sigma)$ acc±std(%)	ILSTSVM (OVO-mv) $(C_1 = C_2,$ $C_3 = C_4, \sigma)$ acc±std(%)	ILSTSVM (OVO-md) $(C_1 = C_2,$ $C_3 = C_4, \sigma)$ acc±std(%)	TBSVM (OVO-mv) $(C_1 = C_2,$ $C_3 = C_4, \sigma)$ acc±std(%)	TBSVM (OVO-md) $(C_1 = C_2,$ $C_3 = C_4, \sigma)$ acc±std(%)
Balance	$2^5, 2^1$	$2^8, 2^0$	$2^3, 2^2$	$2^{-4}, 2^{-2}$	$2^4, 2^{-12}, 2^{-1}$	$2^{-11}, 2^{-12}, 2^{-1}$	$2^4, 2^{-13}, 2^1$	$2^{11}, 2^{-11}, 2^0$
	99.97 ± 0.7	100 ± 0	96.37 ± 2.16	97.34 ± 1.75	97.33 ± 1.9	98.3 ± 1.42	99.5 ± 0.96	99.95 ± 0.28
Hayes-Roth	$2^2, 2^3$	$2^4, 2^3$	$2^3, 2^6$	$2^{-4}, 2^6$	$2^5, 2^{-9}, 2^6$	$2^{-3}, 2^{-12}, 2^5$	$2^{10}, 2^{-7}, 2^3$	$2^9, 2^{-16}, 2^4$
	86.6 ± 7.7	86.16 ± 7.81	86.4 ± 8.9	86.5 ± 9.74	86.9 ± 9.54	86.64 ± 9.01	87.34 ± 9.23	86.6 ± 8.72
Ecoli	$2^{-3}, 2^5$	$2^{-1}, 2^4$	$2^{-2}, 2^{-2}$	$2^{-2}, 2^{-2}$	$2^{-3}, 2^{-8}, 2^2$	$2^0, 2^0, 2^{-2}$	$2^{-2}, 2^{-14}, 2^6$	$2^{-2}, 2^{-3}, 2^0$
	89.18 ± 4.31	89.55 ± 4.51	87.22 ± 4.58	86.86 ± 5.66	89.67 ± 5.06	89.58 ± 4.48	89.61 ± 4.9	89.91 ± 4.42
Glass	$2^{-4}, 2^0$	$2^{-4}, 2^0$	$2^{-3}, 2^1$	$2^{-3}, 2^1$	$2^{-3}, 2^{-8}, 2^0$	$2^{-1}, 2^{-6}, 2^{-1}$	$2^{-4}, 2^{-9}, 2^0$	$2^{-4}, 2^{-9}, 2^0$
	72.79 ± 8.2	73.37 ± 8.48	73 ± 9.97	75.34 ± 8.43	74.07 ± 8.7	72.75 ± 9.24	74.55 ± 9	74.3 ± 8.99
Iris	$2^{-2}, 2^6$	$2^{-2}, 2^6$	$2^3, 2^8$	$2^1, 2^7$	$2^{-1}, 2^{-7}, 2^4$	$2^{-1}, 2^{-7}, 2^4$	$2^0, 2^{-6}, 2^5$	$2^{-1}, 2^{-3}, 2^3$
	98.27 ± 3.23	97.53 ± 3.5	98.47 ± 2.82	98.67 ± 2.84	98 ± 3.48	98.13 ± 3.15	98.67 ± 2.84	98.4 ± 3.3

3.3.2 The minimum average distance vote as a tie breaker

In the maximum votes strategy (3.1) it may happen that the largest number of votes is obtained for more than one class. The usual strategy is to pick any of these classes randomly as winner. As an alternative, we propose to use the minimum average distance as a tie breaker. We will consider this method later in this thesis.

CHAPTER IV

THE STRUCTURE OF HUMAN EYE AND DIABETIC RETINOPATHY

This chapter gives a brief description of the characteristics and stages of diabetic retinopathy.

4.1 The structure of the eye and vision functions

The eye is a complex and fascinating primary organ of vision. Over 80% of our sensory input comes through sight. Some of the major parts of the eye along with its functions are outlined below.

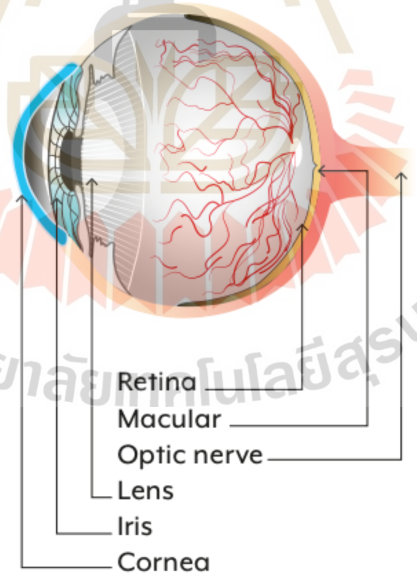


Figure 4.1 Cross-sectional anatomy of the eye. (source: WHO, 2020)

There are three compartments in the eye, namely, the anterior chamber, the posterior chamber, and the vitreous cavity.

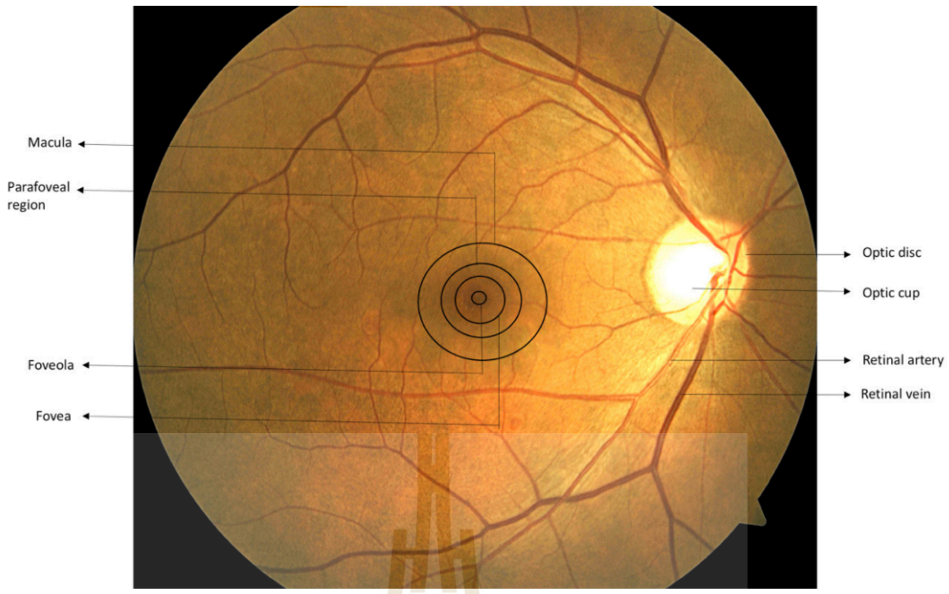


Figure 4.2 Ophthalmoscopic appearance of the eye.

(source: https://www.worldscientific.com/doi/pdf/10.1142/9789813275607_0001)

The **cornea** is a clear avascular tissue covering one-sixth of the surface of the eyeball. Most of the eyes refractive power is produced by the cornea.

The **iris** is the most anterior part of the uvea - the main vascular layer of the eye. It is composed of blood vessels and connective tissue and is responsible for its distinctive color. The contraction and dilation of pupil is allowed due to the mobility of the iris .

The **lens** is a biconvex structure located behind the pupil. It contributes in focusing light or an image onto the retina.

The **retina** is the innermost nervous coat located at the back of the eye. It receives the focused light from the lens, converts it to neural signals and transmit to the brain.

The **macula** is yellowish oval-shaped and located in the centre of the retina. It is highly responsive and sensitive to light and color. It ensures central vision to the eye.

The **optic nerve**, located at the back of the eye, is the largest sensory nerve of the eye. It is responsible for visual communication from the retina to the brain.

4.2 Diabetic retinopathy

4.2.1 Clinical features of diabetic retinopathy

Clinically visible features associated with diabetic retinopathy (DR) are discussed below. From these features found during opthalmoscopic eye examination, ophthalmologists make an appropriate decision on DR grading.

Microaneurysms. These are tiny sac like structures budding from very small blood vessels and often appear as tiny red dots. They are formed when there is not enough oxygen supply to the capillaries in the retina.

Retinal haemorrhages. These are blot-shaped features which result from ruptured microaneurysms, capillaries and venules. They are observed deep within the retina and may be transient in appearance.

Hard exudates. They are yellowish deposits, ranging in size from small dots to ring shaped caused by leakage from abnormal vessels.

Soft-exudates or cotton-wool spots. They are seen as superficial, whitish, fluff-like patches which result from the obstruction of terminal retinal arterioles. A large number of soft exudates often indicate the advancing stage of DR.

Intra-retinal microvascular abnormalities (IRMA). These are microvascular loops that result from their role as shunts between arterioles and venules in areas of increasing capillary closure.

Venous beading. This is a sign of retinal ischemia and occurs adjacent to an area of decreased perfusion. It occurs when the disease has progressed to an advanced stage.

Neovascularization. These are new abnormal growth of blood vessels near or from the optic disc area into the vitreous and towards the centre of the eye. These blood vessels are fragile and highly permeable.

Vitreous haemorrhages. The newly grown blood vessels rupture and bleed into the vitreous gel in the centre of the eyeball behind the lens. The severe bleeding may be associated with total visual loss.

4.2.2 Classification of diabetic retinopathy

Based on the observations of clinical features, DR is classified into four stages.

1. **Mild Non Proliferative Diabetic Retinopathy (NPDR)**

   This is characterized on observation of a few microaneurysms from fundus eye images from funduscopy. There is no effect on vision changes and there are no other significant findings.

2. **Moderate Non Proliferative Diabetic Retinopathy (NPDR)**

   This is characterized on observation of several microaneurysms, haemorrhages, and cotton-wool spots. Also, a mild degree of venous beading and IRMA can be present.

3. **Severe Non Proliferative Diabetic Retinopathy (NPDR)**

   This is characterized on findings of high degree of severity of microaneurysms, haemorrhages, cotton-wool spots, venous beading and IRMA.

4. **Proliferative Diabetic Retinopathy (PDR)**

   This is an advanced stage of DR. Findings such as retinal neovascularization and/or vitreous haemorrhages can be observed in addition to the observations from NPDR. Severe visual loss will be noticed by the patients.

In this thesis, only the various stages of nonproliferative diabetic retinopathy are considered, as patients with proliferative diabetic retinopathy are already noticeably visually impaired and detection is no longer required.

Figure 4.3 shows normal eye fundus and NPDR eye fundus images.

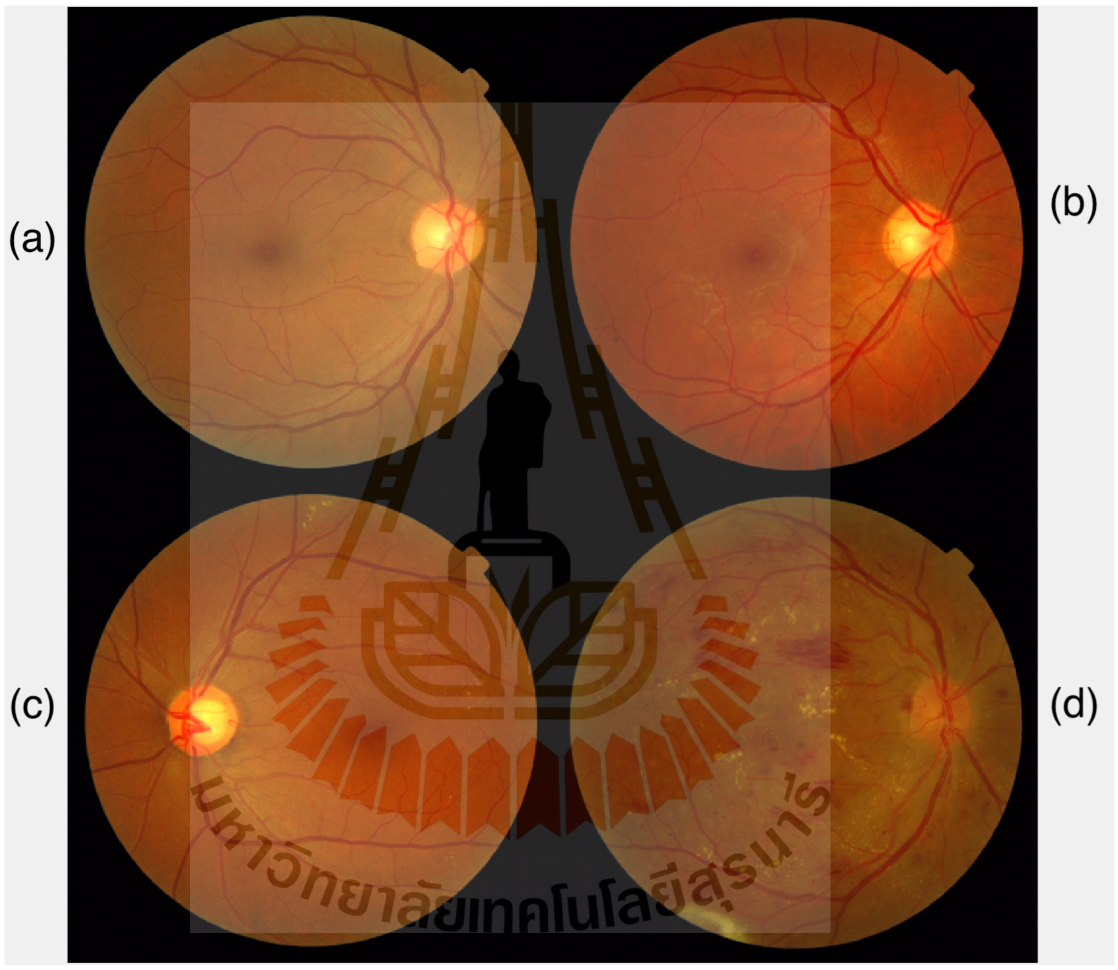


Figure 4.3 Digital eye fundus photographs. (a) Normal. (b) Mild NPDR. (c) Moderate NPDR. (d) Severe NPDR.

CHAPTER V

IMAGE PROCESSING TECHNIQUES

In this chapter, we review some of the fundamentals of digital image processing. These techniques are crucial ingredients in the preparation of the eye images for diabetic retinopathy detection and classification. Most of the presentation is based on the book Digital Image Processing using MATLAB by Gonzalez et al. (2009).

5.1 Digital image processing

Digital image representation.

An image can be represented as a two-dimensional function $f(x, y)$, where x, y are coordinates, and the value of f at the point (x, y) is called the intensity of the image at (x, y). Colour images are formed by the combination of several individual images. Primary components of a colour image are red, green, and blue and they are being referred to as an RGB colour system. Grayscale images have monochrome intensity values (shades of gray) and can be obtained from a RGB image.

In digital image processing, the image region is divided up into small squares called pixels. The image $f(x, y)$ is assumed to be constant on each pixel. Thus one represents an image as a function $f(i, j)$ or $f_{i,j}$ where the integer pairs (i, j) range over the pixel coordinates. We will switch freely between notation $f(x, y)$ and pixel notation $f(i, j)$ in our discussion. Often, the intensity levels $f(i, j)$ will also be in a discrete range, e.g. $f(i, j) \in \{0, 1, \ldots, 255\}$.

In image processing, the term spatial domain means the image plane itself. The methods related to spatial domain processing involve direct manipulations of image pixels. Intensity transformations and spatial filtering are the two most important processing methods in this domain.

Image histogram

Suppose a digital image contains in total L possible integer intensity levels. Then the image histogram is defined as a discrete function

$$h(u_k) = n_k$$

where u_k refers to the kth intensity level and n_k is the number of pixels with intensity level u_k. One obtains a normalized histogram as

$$p(u_k) = \frac{h(u_k)}{n}, \qquad k = 0, 1, \ldots, L-1.$$

where n is the total number of pixels in an image. We notice that the function $p(u_k)$ is the probability of occurrence of the image intensity level u_k.

Histogram equalization

Let $p(u_j),\ j = 0, 1, \ldots, L-1$ denote an image histogram. Define a mapping $T : I \to [0, 1]$ called equalization transformation by

$$\begin{aligned} s_k &= T(u_k) \\ &= \sum_{j=0}^{k} p(u_j) = \sum_{j=0}^{k} \frac{n_j}{n},\ k = 0, 1, \ldots, L-1 \end{aligned}$$

where each s_k is an integer-output value corresponding to the input value u_k. One can show that the output values s_k are approximately uniformly distributed.

The histogram equalization results in an output image with higher contrast since the histogram gets spread considerably over the entire range of intensity.

Contrast limited adaptive histogram equalization (CLAHE)

In CLAHE, the histogram equalization is carried out in small regions of the image called tiles instead of the entire image. The processed tiles are then joined using bilinear interpolation which is effective in overcoming the artificially created boundaries. CLAHE has an advantage over normal histogram equalization as it can avoid over amplification of noise in certain regions with homogeneous intensity.

Median filter

To improve the image quality, reducing certain amount of noise is of paramount. One of the popular spatial filtering techniques (linear and non-linear) is the median filter. The median filter is a non-linear filter which works by running through all image pixels and replacing each pixel value by the computed median value from neighboring pixels. This filter is most effective in preserving edges of structures in the image and removing salt-and-pepper noise.

5.2 Morphological image processing

In this section we review a wide range of image processing techniques based on the shape (morphology) or structure of features in an image. For this purpose, we recall some basic set theory concepts as follows:

Let A and B be subsets of an additive group G [Here the plane $\mathbb{R}^2$].

1. The difference of sets A and B, denoted $A - B$, is

$$A - B = \{w \,|\, w \in A, w \notin B\}$$

2. The reflection of a set B, denoted $\hat{B}$, is

$$\hat{B} = \{w \,|\, w = -b \text{ for } b \in B\}$$

3. The translation of set A by point z, denoted by $(A)_z$, is defined as

$$(A)_z = \{c \,|\, c = a + z \text{ for } a \in A\}$$

Binary image

An image which contains the pixel values 0 or 1 only is called a binary image. Based on this type of image, we define some important morphological operations.

Dilation

It is an operation that makes objects in an image grow or thicken. The growing or thickening of objects in the image is controlled by a specific shaped called structuring element (SE).

Let A be an image and S be a structuring element. Then the dilation of A by S, denoted $A \oplus S$, is defined as

$$A \oplus S = \{z \,|\, (\hat{S})_z \cap A \neq \emptyset\} \tag{5.1}$$

So, the set $A \oplus S$ consists of all the original locations of structuring elements such that the reflected and translated S has at least one element in common.

Erosion

It is an operation which has the effect of making objects in an image shrink or thin. Similar to dilation, the extent of shrinking or thinning is controlled by a structuring element. The erosion of an image A by structuring element S, denoted $A \ominus S$, is defined as

$$A \ominus S = \{z \mid (S)_z \subseteq A\}$$

that is, the set consists of all the points z such that the structuring element S translated by z fits entirely in A.

Opening

The opening of image A by the structuring element S, denoted as $A \circ B$, is defined as

$$\begin{aligned} A \circ S &= (A \ominus S) \oplus S \\ &= \bigcup\{(S)_z \mid (S)_z \subseteq A\} \end{aligned}$$

So opening of A by S is the erosion followed by the dilation operation. Morphological opening has the ability to remove completely those parts of an object in the image that cannot contain the structuring element. It is also used to remove thin connected components, bumps and smoothen the contours.

Closing

The morphological closing of A by S, denoted by $A \bullet B$, is defined as the dilation of A by S followed by the erosion operation on the result by S, that is,

$$A \bullet B = (A \oplus S) \ominus S$$

Closing operations are used to join narrow breaks and fill holes which are smaller than the structuring element. It also smoothen the contours.

Morphological reconstruction

Morphological reconstruction is transformation by iteration involving two images, called marker and mask, and a structuring element. The marker image acts as a starting point in the iteration, while the mask image constrains it. The structuring element provides the connectivity of the image in the transformation.

Suppose A and C are mask and marker images respectively and S a structuring element. Morphologically reconstructing A from C, denoted as $R_C(A)$, involves the following iterations:

1. Initialize u_1 to be the marker image, C.
2. Define the structuring element S.
3. Repeat the iterations $u_{k+1} = (u_k \oplus S) \cap A$
   until $u_{k+1} = u_k$
3. $R_A(C) = u_{K+1}$.

It is noted that $C \subseteq A$.

Opening by reconstruction

The opening by reconstruction of image A using structuring element S is defined as $R_A(A \ominus S)$. This is a transformation whereby the shape of the object is restored to its full form from a certain part left behind after an erosion. Though dilating back after an erosion also has an effect of restoring the shape of an object, its accuracy is heavily dependent in the nature of the structuring element. In this regard, opening by reconstruction is most accurate in retaining the shape.

Gray-scale morphology

Binary morphological operations can be extended to gray-scale images, For dilation and erosion, they are based on the maxima and minima of the neighbourhood pixels. In the following the definitions concerning gray-scale morphology, we suppose f be a gray-scale image and b a structuring element.

Gray-scale dilation

The gray-scale dilation of f by b, denoted as $f \oplus b$, is defined as

$$(f \oplus b)(x, y) = \max\{f(x - x', y - y') + b(x', y') \mid (x', y') \in D_b\}$$

where D_b is the domain of b which is a binary matrix specifying the locations in the neighbourhood included in the max operation.

Flat structuring elements are usually used in gray-scale dilation transformation. For such elements, the height of b is 0 at all coordinates of D_b. So

$$b(x', y') = 0, \quad \forall (x', y') \in D_b.$$

In this case the gray-scale dilation is simply a local-maximum operator and the equation (5.2) becomes

$$(f \oplus b)(x, y) = \max\{f(x - x', y - y') \mid (x', y') \in D_b\}$$

which is completely specified by pattern of 0s and 1s in the binary matrix representing D_b.

Gray-scale erosion

The gray-scale erosion of f by b, denoted as $f \ominus b$, is defined as

$$(f \ominus b)(x, y) = \min\{f(x - x', y - y') - b(x', y') \mid (x', y') \in D_b\}$$

where D_b has a similar meaning as in dilation but in the sense of min operation. Using the flat structuring element, the equation 5.2 reduces to

$$(f \ominus b)(x, y) = \min\{f(x - x', y - y') \mid (x', y') \in D_b\}$$

which is a local minimum operator. The minimum is taken over neighbourhood pixels specified by 1-valued elements in the matrix representing D_b.

Morphological opening and closing operations in binary images can also be extended to gray-scale images.

Opening

The morphological opening of f by b, written $f \circ b$, is defined a the erosion of f by b followed by dilation on the result $f \ominus b$. Mathematically,

$$f \circ b = (f \ominus b) \oplus b$$

The dilation and erosion operations are understood as for the gray-scale image operations.

Closing

The morphological closing of f by b, written $f \bullet b$, is defined as the dilation followed by the erosion, that is,

$$f \bullet b = (f \oplus b) \ominus b$$

In image processing, openings remove small bright objects depending on the shape and size of the structuring element, while leaving bright and large objects untouched. Closings, on the other hand, suppress the small dark details in the image. These operations are often used in combinations in gray-scale images to reduce noise and smoothen the image.

Morphological reconstruction of gray-scale images follow the same iterative procedure as outlined for binary images.

5.3 Image segmentation

Morphological image processing, as discussed, involves input images and through a wide range of techniques, renders an output containing attributes of interest from the input images. In this view, image segmentation is another major technique. We briefly review some of the basic techniques in image segmentation.

Edge detection

One of the most common techniques in image segmentation is to detect a rapid change in the intensity of an image. This is achieved by so called an edge detection. For the detection of such changes, the first order and second order derivatives of the image $f(x, y)$ are used. We know that the first order derivative also known as gradient of f is the vector

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$$

The magnitude of this vector is $|\nabla f| = \sqrt{f_x^2 + f_y^2}$.

For the second-order derivative, we use the Laplacian of f, that is,

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

There are several edge detection methods:

- Sobel edge detector.
- Prewitt edge detector.

- Roberts edge detector.
- Laplacian of log (LoG) detector.
- Zero-crossings detector.
- Canny edge detector.

The Canny edge detector is one of the most popular methods since it outperforms the other methods. We summarize the procedure of this method below:

- The input image, f, is firstly smoothed and denoised by applying a Gaussian filter.
- For every point (x, y) in the image, the gradient ∇f is computed in a neighbourhood of the point. The point is identified as an edge point, if $|\nabla f|$ has a local maximum at (x, y) (or is close to the local max). The direction of the edge is then determined by $\tan^{-1}\left(\frac{f_x}{f_y}\right)$.
- Points which are not identified as edge points will be assigned an intensity of 0.
- Edge points are classified into two types, weak and strong edge points by choosing two threshold levels $T_1 < T_2$. Edge points (x, y) with $f(x, y) \geq T_2$ are identified as strong edge pixels, and those with $T_1 < f(x, y) < T_2$ as weak points. Weak edge pixels are only accepted as edge pixels if they are 8-connected to strong pixels, a process called edge linking.

Thresholding

In various applications of image processing, thresholding an image to segment a region of interest plays a key role. Here we discuss the basic concept of image thresholding.

Let f be a gray-scale image. Suppose we are interested in segmenting out a certain region of image that has intensity of at least T. A point (x, y) with $f(x, y) > T$ is called foreground or object point, and with $f(x, y) \leq T$ is called the background point. The thresholded image region, also called a binary image , $g(x, y)$, is defined by

$$g(x,y) = \begin{cases} u & \text{if } f(x,y) > T \\ v & \text{if } f(x,y) \leq T \end{cases}$$

By convention, one chooses pixel values $u = 1$ and $v = 0$.

If the value of T is kept fixed over the entire image, then the equation (5.3) is called global thresholding. On the other hand, if the value of T changes over an image, then the equation (5.3) is referred to as variable thresholding.

Otsu's global thresholding

Consider an image $f(x, y)$ and its histogram component denoted as

$$p = \frac{n_k}{n}, \qquad k = 0, 1, \ldots, L-1,$$

where n is the total pixels in f, n_k denotes the number of pixels with intensity level k, and L refers to the total possible intensity levels in f.

Suppose that we choose a threshold t such that I_1 has the pixel values $\{0, 1, \ldots, t\}$ and I_2 with $\{t+1, t+2, \ldots, L-1\}$. Let $m_1(t)$ and $m_2(t)$ denote mean intensities of the sets I_1 and I_2 respectively, and m_G denote the mean intensity of the entire image. Then the between class variance, $\sigma_B^2(t)$, is defined as

$$\sigma_B^2(t) = P_1(t)[m_1(t) - m_G]^2 + P_2(t)[m_2(t) - m_G]^2 \tag{5.2}$$

where $P_1(t) = \sum_{i=0}^{t} p_i$ is the probability of occurrence of set I_1 and $P_2(t) = \sum_{i=t+1}^{L-1} p_i = 1 - P_1(t)$ is the probability of occurrence of I_2.

This method chooses a threshold t that maximizes the between class variance $\sigma_B^2(t)$. Rewriting the equation (5.2) as

$$\sigma_B^2(t) = \frac{[m_G P_1(t) - m_1(t)]^2}{P_1(t)[1 - P_1(t)]},$$

one can show that the larger the value of $\sigma_B^2(t)$, the higher the chance of a clearly segmented image with the threshold t.

5.4 Image representation and description

To describe a region of an image, the texture content in it can be quantified. Then a texture computed based on statistical and spectral measures can be used to describe the region.

Statistical measures

In analyzing the texture content in an image, one of the common methods is to use the statistical moments of the values of image intensity.

Suppose z_i is a discrete random variable which denotes the intensity levels of image f. Correspondingly, define the normalized histogram as

$$p(z_i) = \frac{n_i}{n}, i = 0, 1, \dots, L-1$$

where L is the number of possible intensity levels of f. A component $p(z_i)$ can be viewed as a probability that the intensity value z_i will occur.

Description on the shape of histogram can be made by its moments about the mean also called the central moments. The n-th central moment is defined as

$$\mu_n = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i) \tag{5.3}$$

where m is the mean

$$m = \sum_{i=0}^{L-1} z_i p(z_i)$$

Based on the statistical moments, we list some of the features that can be used to represent the image texture content:

1. Mean.

$$m = \sum_{i=0}^{L-1} z_i p(z_i) \tag{5.4}$$

   It is a measure of average intensity of the image.

2. Standard deviation.

$$\sigma = \sqrt{\mu_2} = \sqrt{\sigma^2} \tag{5.5}$$

   It measures the average contrast of the image.

3. Smoothness.
$$R = 1 - \frac{1}{1+\sigma^2}$$
This measures relative smoothness of the pixel in a region R of the image. Notice that if $\sigma^2 = 0$, that is, for a region with same pixel values, $R = 0$. However, if the region has large variation in the values of its intensity levels, $R = 1$.

4. Third moment.
This follows from equation (5.3) when $n = 3$, that is,
$$\mu_3 = \sum_{i=0}^{L-1} (z_i - m)^3 p(z_i)$$
This measures the skewness of a image histogram. $\mu_3 = 0$ if the histogram is symmetric, $\mu_3 > 0$ if the histogram is skewed to the right about the mean, and $\mu_3 < 0$ if the skewness is to the left.

5. Entropy.
It is a measure of randomness, given by
$$e_1 = -\sum_{i=0}^{L-1} p(z_i)\log_2 p(z_i)$$
(Here we take $0 \cdot \infty = 0$).

Gray-level co-occurrence matrix

In the measure of textures in an image, the consideration so far has been solely on the information of its histogram. The downfall of this is that it doesn't carry any information on relative pixel positions with respect to each other. In texture analysis, consideration of relative positions of pixels is also important in image description.
Let $f(x, y)$ be an image with L intensity levels z_i, $(i = 0, \ldots, L-1)$. Let T be an operator relating two pixels to another. Then the gray-level co-occurrence matrix (GLCM) determined by T and denoted by G, is the $L \times L$ matrix whose (i, j)-th entry g_{ij} indicates how many pixel pairs determined by T have identities z_i and z_j, respectively.

Based on this co-occurrence matrix G of the operator T, we define another four texture measures determined by T used in image representation:

6. Energy.

   It is the measured sum of squared elements of G, that is,

$$\sum_{i=1}^{L}\sum_{j=1}^{L} P_{ij}^2 \quad \text{such that } P_{ij} = \frac{g_{ij}}{n}$$

   where n is the total number of elements in G.

7. Homogeneity.

   It is a measure of how close the elements of G lie to the diagonal elements of G, that is,

$$\sum_{i=1}^{L}\sum_{j=1}^{L} \frac{P_{ij}}{1+|i-j|}$$

8. Maximum probability.

   It is a measure of the strongest response of the co-occurrence matrix, that is ,

$$\text{max. probability} = \max_{i,j}\{P_{ij}\}$$

9. Entropy.

   As before, this also measures the randomness, that is,

$$e_2 = -\sum_{i=1}^{L}\sum_{j=1}^{L} P_{ij}\log_2 P_{ij}$$

CHAPTER VI

RESEARCH METHODOLOGY

In this research, we have used the Messidor database, first ophthalmologic department, which consists of 400 eye fundus colour numerical images acquired using a color video 3CCD camera on a Topcon TRC NW6 non-mydriatic retinograph with a 45 degree field of view. The images were captured using 8 bits per colour plane at 2240×1488 pixels. All the images are in TIFF format and an Excel file with medical diagnoses is also provided.

All images were graded based on the severity level of the disease; grade 0 for normal, grade 1 for mild NPDR, grade 2 for moderate NPDR, and grade 3 for severe NPDR. In particular, the chosen database contains 152 normal, 30 mild, 69 moderate and 149 severe NPDR fundus eye images.

6.1 Tools

All computations were performed using MATLAB Version 9.7 (R2019b). For the image segmentation processes we use the image processing toolbox, and for binary and multiclass DR classifications, the statistics and machine learning toolbox, libsvm, and where available, twin SVM packages were used.

We used a personal Macbook CPU version i5 1.8GHz Dual-Core memory 8 GB system MacOS 64 bit.

6.2 Retinal image preprocessing

To enhance the image and for the better segmentation outcomes, the fundus images were processed as follows. Firstly, the images were cropped to 738×727 pixels to get the proper field of view and reduce the computational costs. The images in an RGB colour format had certain degree of variations in the colour. To have some sort of uniform threshold level during segmentation processes, all images were colour normalized using one of the high quality images as a reference image as defined in equation (6.1). Then the images were examined for sharpness level. Blurry images were sharpened accordingly since the object of interest cannot be seen clearly let alone detected. Then each colour band was extracted, namely, the red, green, and blue channels and compared. On a comparative study, it was discovered that the green channel image provided higher contrast on the objects of interest from the background. The red and blue channel images showed poor contrast with noise. For most of the different segmentation processes, we have used the green channel image except for the optic disc detection where we have considered the red band.

6.2.1 Colour normalization

Let $f(x, y)$ be any of the red, green and blue channels of a retinal image and $h(x, y)$ those of a reference image. Then, as per Chen et al. (2020) the colour normalization of image f using h as a reference is defined by the equation

$$f_{\text{cn}} = \frac{\sigma_R}{\sigma}(f - \mu) + \mu_R \tag{6.1}$$

where

- μ and μ_R are the mean intensities of the corresponding channel of images f and h respectively as defined in equation (5.4).
- σ and σ_R are the standard deviations of the corresponding channel of images f and h respectively as defined in equation (5.5).
- f_{cn} is the corresponding channel of new colour normalized image.

In equation 6.1, the images f and h are first converted to type double from uint8 for numerical computation. The image composed of the new channels f_{cn} is said to have similar colour distribution as the reference image h.

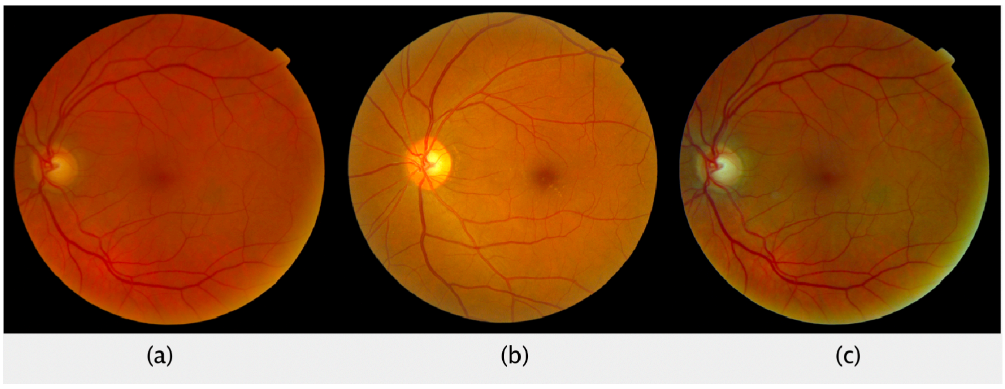


Figure 6.1 Image colour normalization. (a) Input image. (b) Reference image. (c) Colour normalized image.

6.3 Retinal blood vessels segmentation

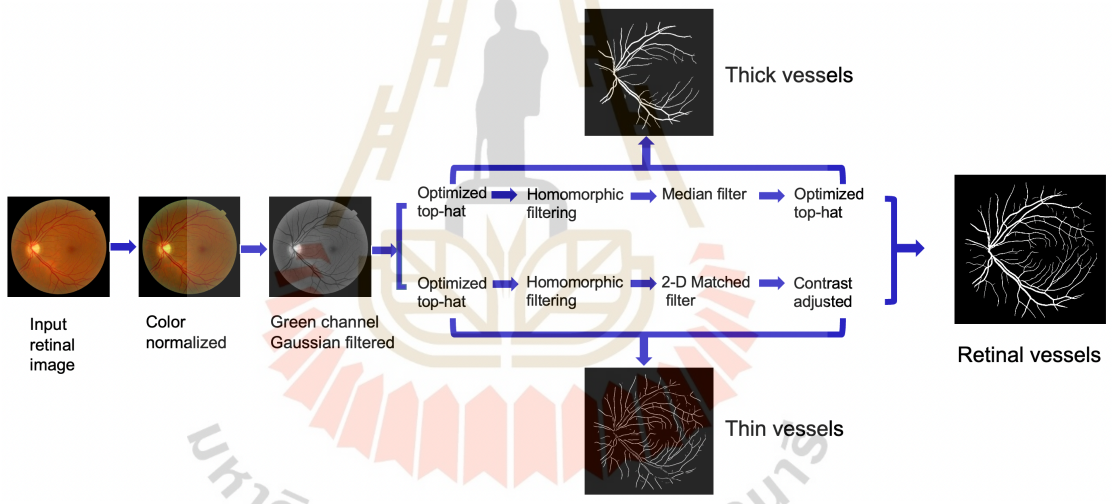


Figure 6.2 Flowchart of retinal vessels segmentation.

In the detection of diabetic retinopathy from eye fundus photographs, the detection of blood vessels is important. There are certain abnormalities that those vessels can present if the eye is diseased. For instance, if the disease has quite advanced to a later stage, it is observed that there is a change in the shape of those vessels and some abnormal new growth.

In segmenting the retinal vessels, we closely followed the work by Ramos-Sotos et al. (2021), where they introduced optimized top-hat transformation and applied homomorphic filtering. However for the thresholding, we have used Otsu's thresholding technique.

The extracted green channel image was first filtered using a Gaussian filter with a standard deviation of $\sigma = 0.68$. This operation enhances the image structures and suppresses the noise generated from image sharpening. In two dimensions, the Gaussian filter is mathematically expressed as a convolution by

$$g(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right),$$

where σ is the standard deviation. That is, the intensity $f(x, y)$ at point (x_0, y_0) is replaced by

$$g(x_0, y_0) = \frac{1}{2\pi^2} \iint\limits_{\mathbb{R}^2} g(x, y) \exp\left(-\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}\right) dxdy$$

Of course, this formula is adjusted to account for the discrete pixel coordinates.

We focus on the segmentations of thick and thin vessels separately and merge them at the end.

Thick vessels extraction

Represent an image of $M \times N$ pixels as an $M \times N$ matrix $I = [f_{ij}]$ with integer pixel values, $0 \leq f_{ij} \leq 255$.

Firstly, the Gaussian filtered image I was inverted, denoted I^C, defined as,

$$I^C = U - I,$$

where U is the $M \times N$ matrix all of whose entries are the maximum pixel values of 255.

Then we define two disk shaped structuring elements, namely S_1 and S_2, of radius 8 and 16 pixels respectively. Then the morphological opening of I^C by S_1 was performed, that is,

$$I_{\text{op}} = I^C \circ S_1, \tag{6.2}$$

followed by the closing operation on (6.2),

$$I_{\text{cl}} = I_{\text{op}} \bullet S_2 \tag{6.3}$$

Then the image in (6.3) was subtracted from the I^C, resulting in an enhanced image. This process is called an optimized top-hat transformation. This can be summarized as,

$$\begin{aligned} I_{\text{th}} &= I^C - [(I^C \circ S_1) \bullet S_2] \\ &= I^C - (I_{\text{op}} \bullet S_2) \\ &= I^C - I_{\text{cl}} \end{aligned} \tag{6.4}$$

Next, a homomorphic filter was applied on the image in (6.4) which further enhances the thick vessels. Homomorphic filtering is a filtering technique in the frequency domain which enables adjustments on illumination and reflectance components of an image. The resulting image is then filtered by a median filter to remove salt and pepper noise. Once again, the filtered image undergoes another optimized top-hat transformation with disk shaped structuring elements of radius 32 and 64 pixels. This fills small holes in the vessels and enhances the structures. Finally, the image is binarized using Otsu's thresholding method, where the algorithm chooses the threshold automatically. Because haemorrhages and blood vessels have similar intensity values, the haemorrhages are also detected together with the blood vessels. To overcome this, we removed components whose total number of pixels were less than 300. The final image represents the extracted thick vessels. The whole procedure is depicted in Figure 6.3.

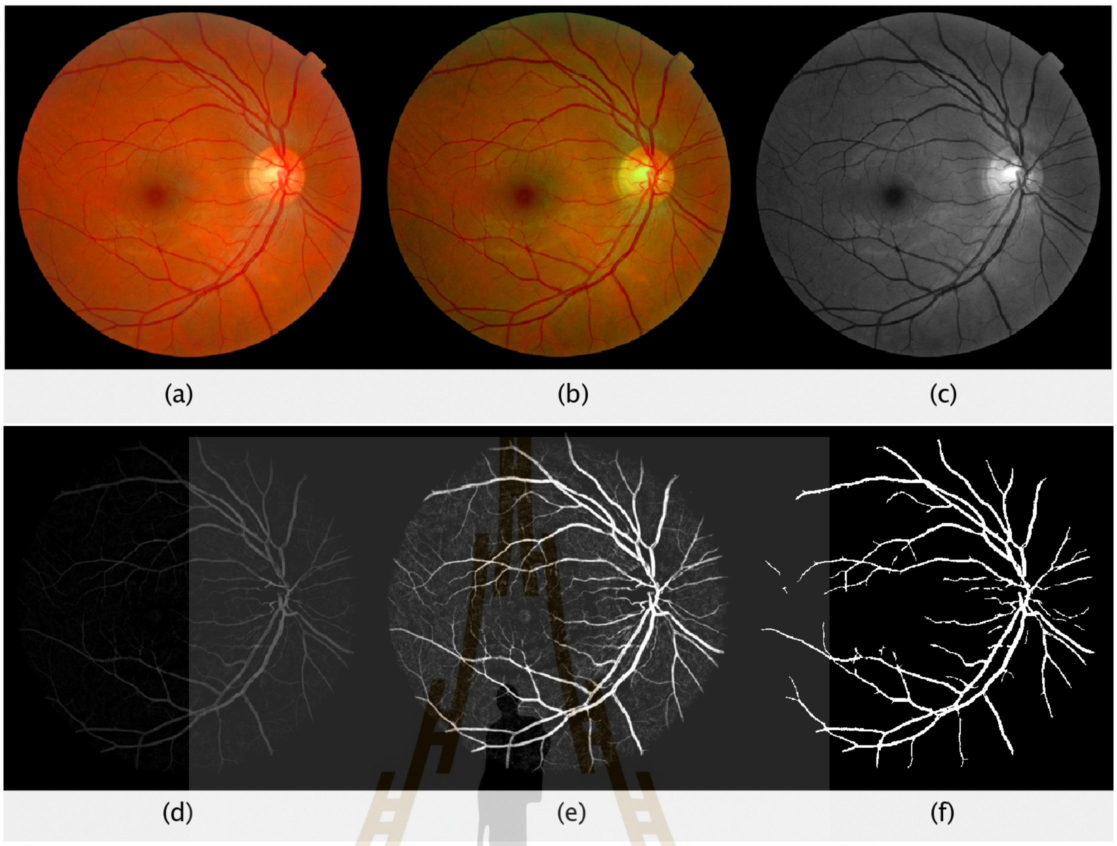


Figure 6.3 Thick vessels extraction. (a) Original image. (b) Colour normalized. (c) Green channel image. (d) Optimized top-hat transformed image. (e) Homomorphic filtered image. (f) Thick vessels extracted by thresholding.

Thin vessels extraction

For this procedure, we also use the optimized top-hat transformation but the radii of S_1 and S_2 are taken as 4 and 20 pixels respectively as shown in Figure 6.4. This is due to the thin vessels spanning over smaller structural regions. Then a homomorphic filter was applied followed by a two-dimensional matched filtering. The concept of two-dimensional matched filtering is that it can enhance certain regions of the image that matches some estimated distribution curve of an image. Thus, this method finds whether the distribution is correlated to to the local image area. This filter highly enhances the tiny blood vessels. The image is then binarized and denoised.

Finally, the thick and thin vessels are merged for a complete retinal blood vessels image.

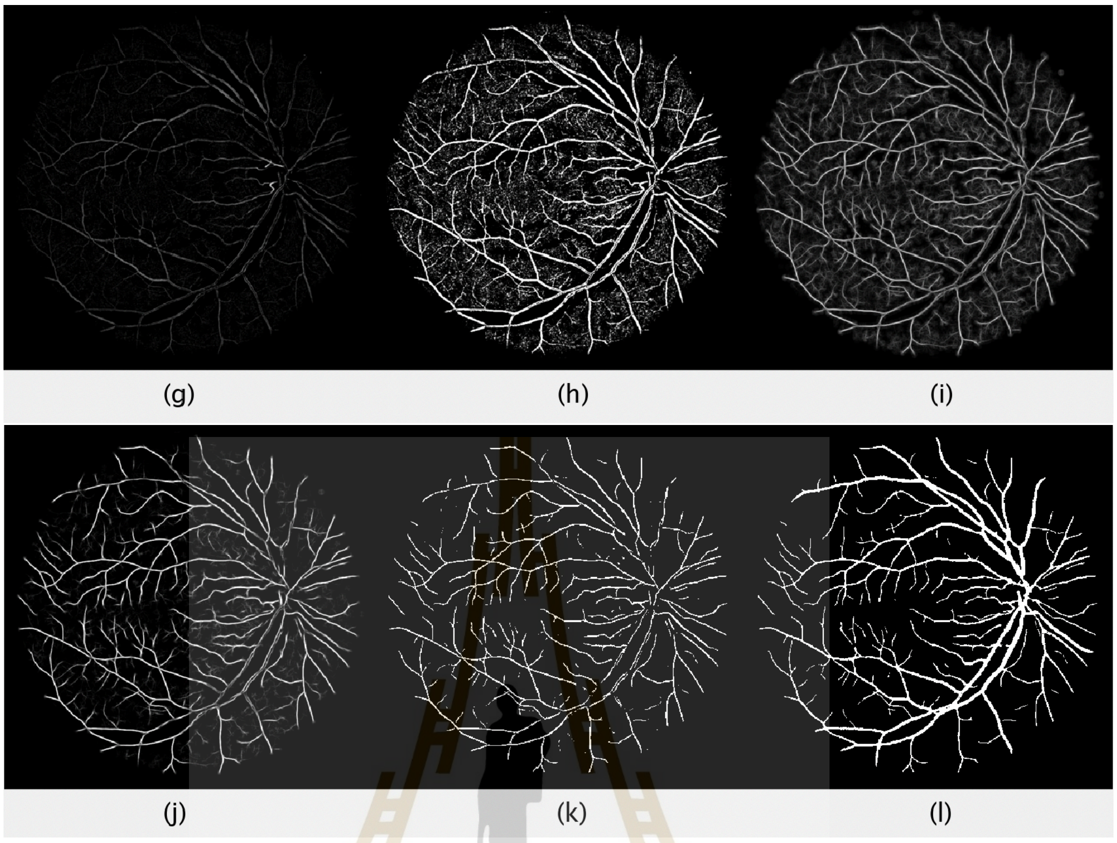


Figure 6.4 Thin vessels extraction and final segmentation. (g) Optimized top-hat for thin vessels. (h) Homomorphic filtered. (i) 2-D match filtered. (j) Contrast adjusted before thresholding. (k) Thin vessels after binarization. (l) Blood vessels segmented after adding image (f) and (k) and noise reduced.

6.4 Microaneurysms detection

Microaneurysms are among the most important clinical pathologies in the identification of diabetic retinopathy. They are the first abnormalities that develop in diabetic retinopathy, and are present as tiny blobs at the end of capillaries. These are the swollen tiny vessels. If this is detected early on, patients can delay the progression of the disease with treatment ultimately saving their eye sight.

Firstly we took the complement I^C of the green component of the original image. Then I^C was corrected for the non-uniform background using morphological open-close filtering called an alternate sequential filtering technique. This method consists of a combination of morphological openings and closings using a series of structuring elements. In

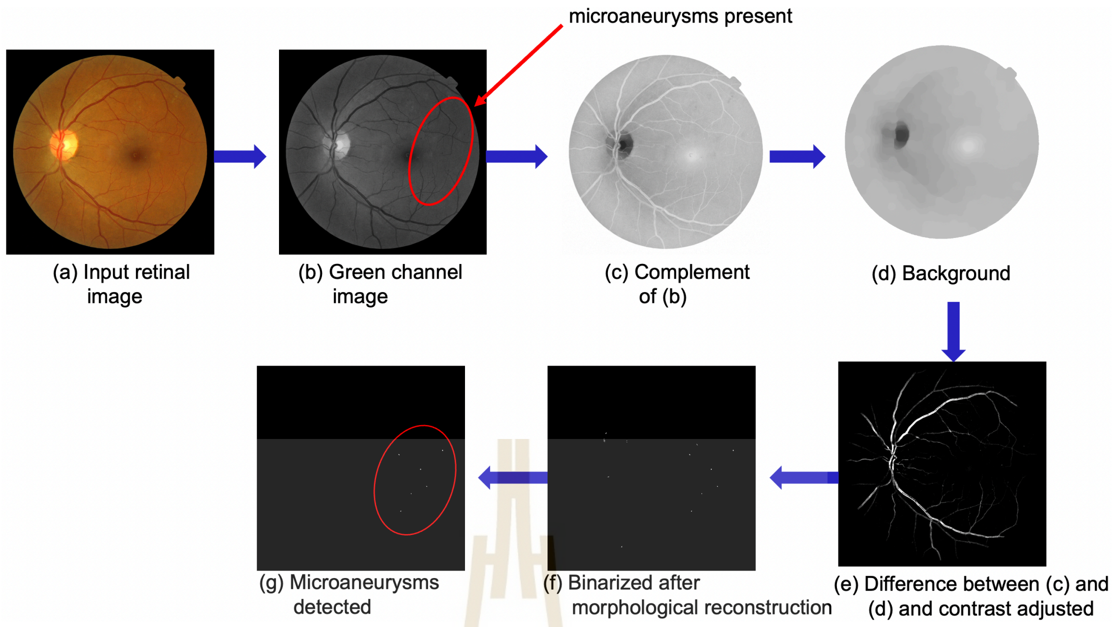

microaneurysms present
(a) Input retinal image
(b) Green channel image
(c) Complement of (b)
(d) Background
(e) Difference between (c) and (d) and contrast adjusted
(f) Binarized after morphological reconstruction
(g) Microaneurysms detected

Figure 6.5 Flowchart showing detection of microaneurysms.

our research, we have chosen flat disk shaped structuring elements $S_i,\ i = 1, 2, \ldots, 12$, with radius $r_i = 2, 3, \ldots, 13$, and performed open-close filtering, that is,

$$I_{\text{ocf}} = (I^C \circ S_i) \bullet S_i,\ \ i = 1, 2, \ldots, 12.$$

The resultant image I_{ocf} yielded a smooth and uniform background. Then the background image I_{ocf} was subtracted from I^C

$$I_{\text{enh}} = I^C - I_{\text{ocf}}$$

where the image I_{enh} is enhanced background corrected. Further, the contrast adjustment was made to this image to highlight the tiny regions corresponding to the microaneurysms. Then this image was morphologically reconstructed using the marker image

$$I_{\text{mark}} = I_{\text{enh}} \otimes I_{\text{bv}},$$

where I_{enh} is the contrast adjusted, I_{bv} is the blood vessels mask from Figure 6.4, and the operation $\otimes$ is the element wise matrix multiplication. The reconstructed image

$$I_{\text{rc}} = R_{I_{\text{mark}}}(I_{\text{enh}})$$

is subtracted from the contrasted adjusted image I_{enh}, that is,

$$I_{\text{ma}} = I_{\text{enh}} - I_{\text{rc}}$$

The image I_{ma} was then binarized, denoted I_{bw} setting threshold $T = 53\%$ of the maximum intensity obtained by trial and error. The binarized image contains the tiny components arising from the microaneurysms, other remnants left behind by the blood vessels and haemorrhages, background artefacts and some noise. We filter the image I_{bw} further using connected component analysis. Each connected component in the image was labelled. Then we filter for microaneurysms using the following set of features in the analysis:

- Area

  Total number of white pixels in each connected component.

- Major axis length

  Length of major axis of the ellipse approximating the connected component. (To be precise, this is the ellipse which has the same normalized second central moment as the region.)

- Minor axis length

  Length of the minor axis of the ellipse approximating the connected component.

- Circularity

  This is a measure of roundness of an object defined by

  $$\text{Circularity} = \frac{4\pi \cdot \text{Area}}{\text{P}^2},$$

  where P denotes the perimeter which is computed as the sum of boundary pixels of an object.

- Eccentricity

  This is defined as the ratio of distance between foci of the ellipse to its major axis length.

Using the features mentioned above, we set that for each connected component to be considered a microaneurysm, then Area $\in [5, 22]$ pixels, Circularity $\in [1, 2.5]$, Eccentricity ≤ 0.9, and the ratio of major axis length to the minor axis length should be in the range $[1, 2.5]$. This lead to removal of noise and objects which are not microaneurysms. The final image contains the detected microaneurysms. The whole process is illustrated in Figure 6.6.

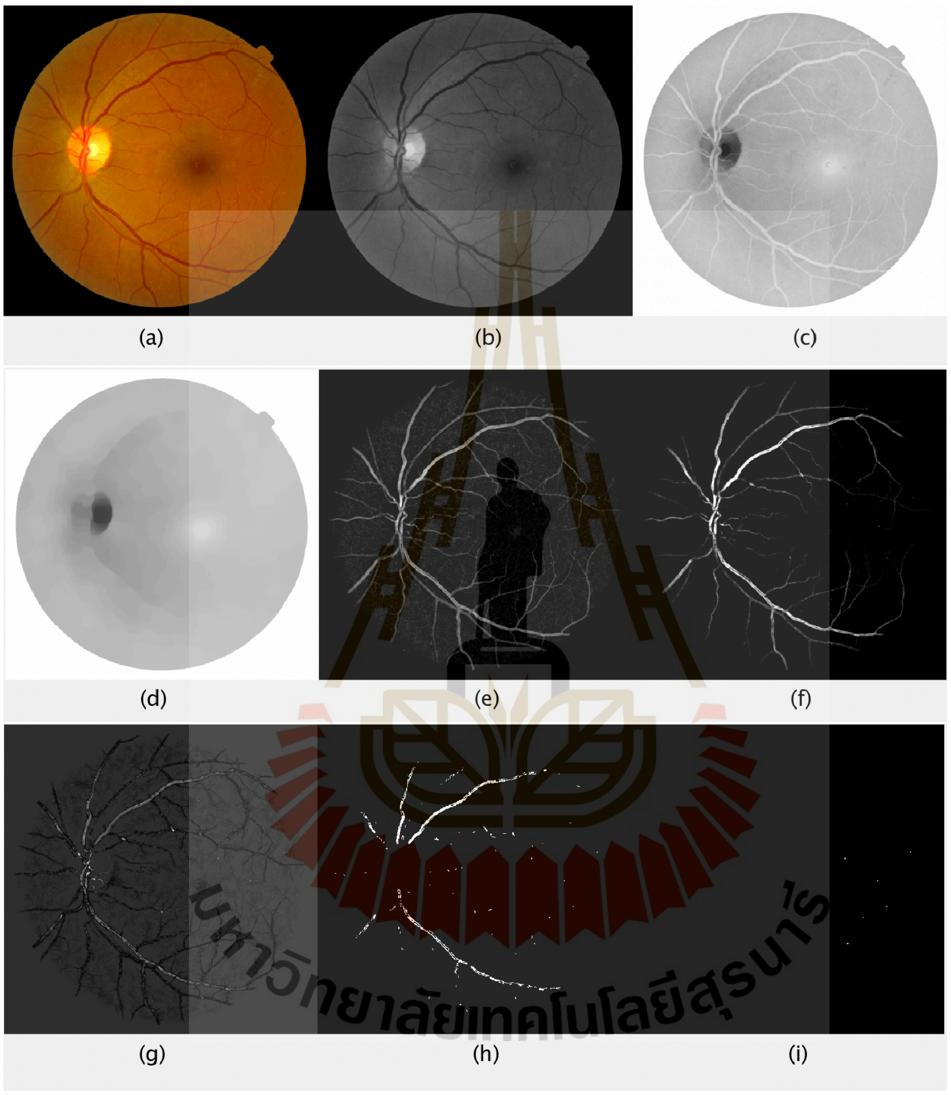


Figure 6.6 Microaneurysms detection. (a) Original image. (b) Green component. (c) Complement of (b). (d) Background estimation. (e) Subtraction of (d) from (c). (f) Contrast adjusted. (g) Morphologically reconstructed image. (h) Binarized image. (i) Microaneurysms detected after connected component analysis.

6.5 Haemorrhages detection

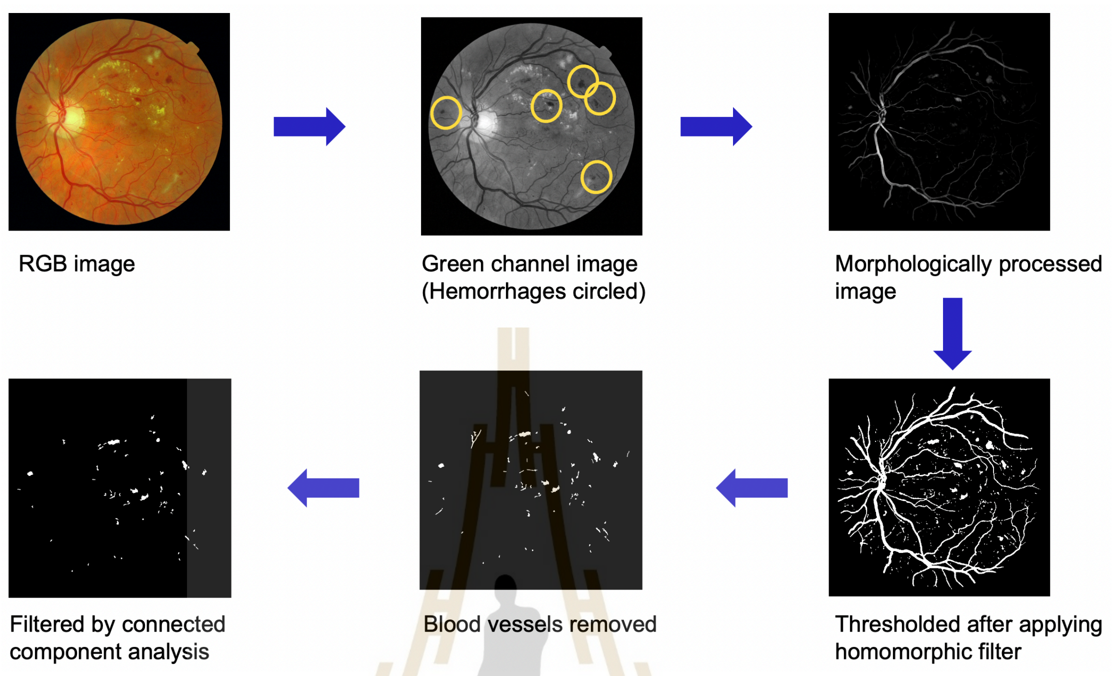

RGB image
Green channel image (Hemorrhages circled)
Morphologically processed image
Thresholded after applying homomorphic filter
Blood vessels removed
Filtered by connected component analysis

Figure 6.7 Flowchart of haemorrhages detection.

We observe that the intensity levels of haemorrhages in the fundus eye images are similar to the blood vessels and microaneurysms. Due to this fact, the segmentation process for the haemorrhages share similar steps to that in the detection of microaneurysms.

Firstly, the inverted green component image was background corrected as in the equations (6.4) and (6.4). Then the image was filtered with median filter to suppress noise and the contrast is adjusted. Next, the homomorphic filter was applied which further enhanced the structure of haemorrhages. The image was then binarized, denoted I_{bwh}. Since I_{bwh} also contains blood vessels, we subtracted the detected blood vessels, that is,

$$I_{\text{hm}} = I_{\text{bwh}} - I_{\text{bv}} \tag{6.5}$$

where I_{bv} is the retinal blood vessels mask obtained in Figure 6.4.

The image I_{hm} still contains false positive foreground pixels from the background and noise. For this, we perform mean intensity pixels analysis by labelling all the connected components:

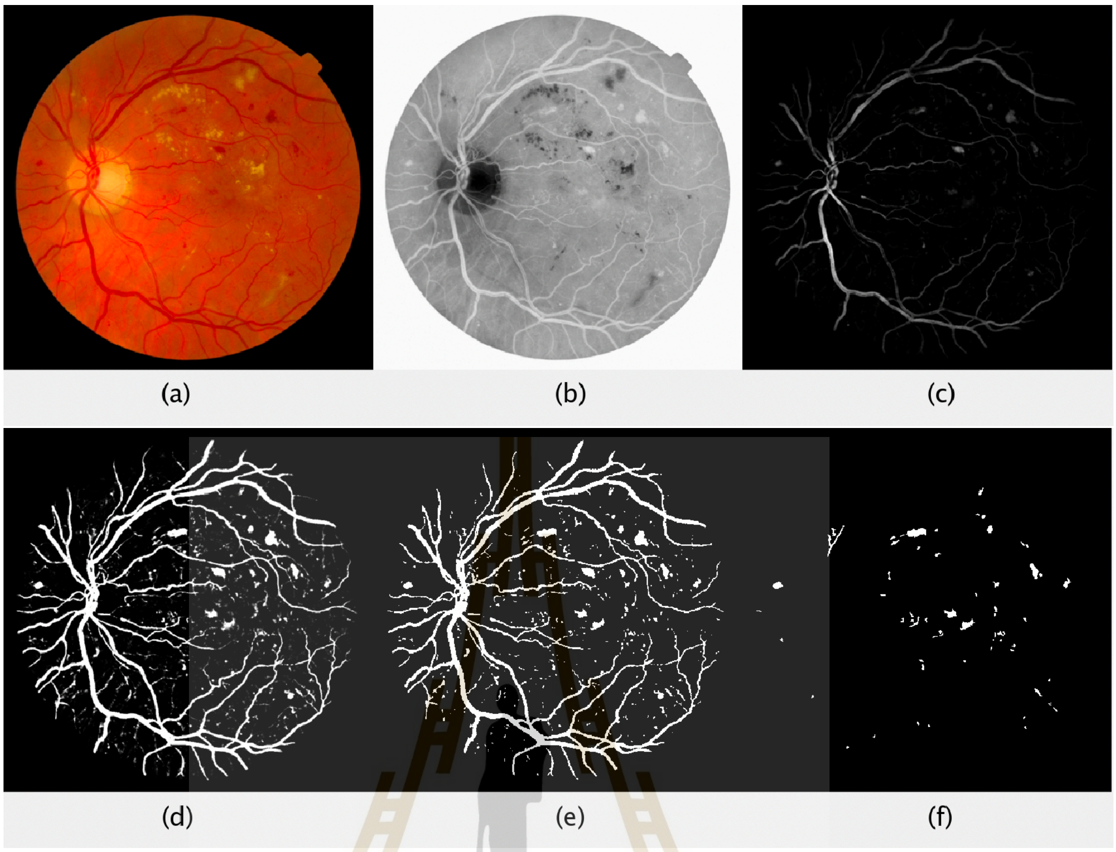


Figure 6.8 Haemorrhages detection. (a) Original colour normalized image. (b) Inverted green channel image. (c) Background corrected. (d) Homomorphic filtered. (e) Thresholded image. (f) Blood vessels subtraction and connected component analyzed leading to haemorrhages segmentation.

- For each connected component, compute its mean intensity value in the green channel.
- Remove the blobs or regions whose mean intensity in the green channel is greater than 95.

Furthermore, some blood vessels can still be observed in the image. To overcome this, we once again performed the connected component analysis based on eccentricity. We disposed of those components whose eccentricity $\notin [0, 0.979]$. Also the connected components with area less than 25 pixels were discarded since they overlap with microaneurysms. The final filtered image contains the retinal haemorrhages as shown in Figure 6.8.

6.6 Exudates detection

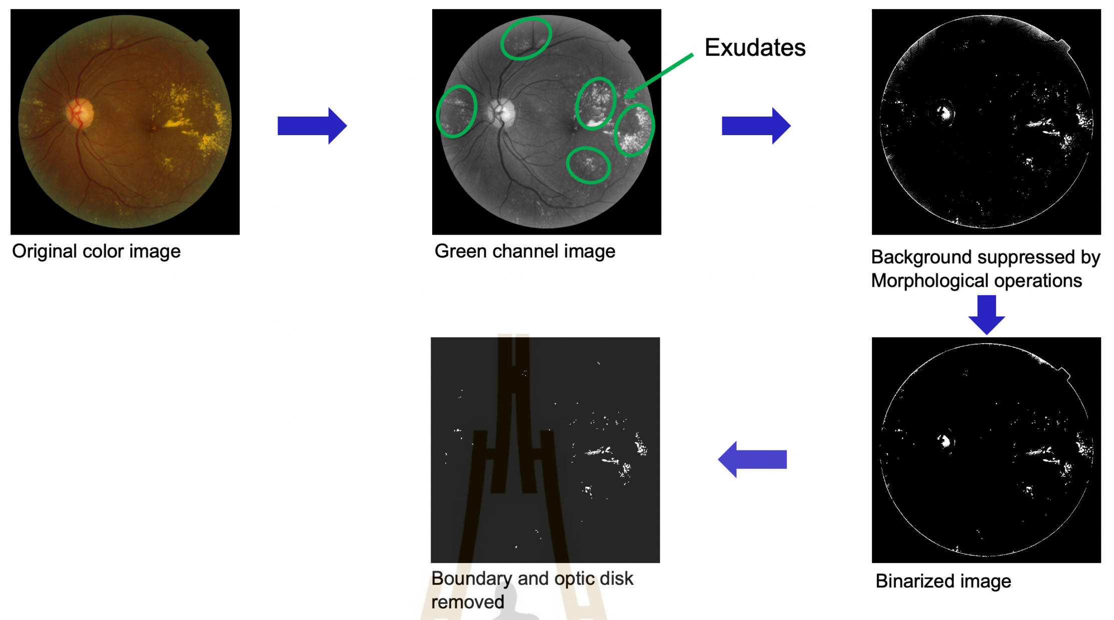

Exudates
Original color image
Green channel image
Background suppressed by Morphological operations
Binarized image
Boundary and optic disk removed

Figure 6.9 Flowchart illustrating segmentation of exudates.

Exudates in contrast to blood vessels, haemorrhages and microaneuryrsms, have higher intensity values since they are yellowish bright fluffy patches seen in the diseased retinal images. Due to this, image colour normalization was not required for this procedure since it was learned that colour normalization reduces the intensity values of exudates which is undesired in this case as our main goal was to extract exudates. The overview of exudates segmentation process is outlined in Figure 6.10. In this procedure, we work with the green channel image itself instead of inverting it. Background correction is also necessary. For this we used morphological close-open filtering technique

$$I_{\text{cof}} = (I_{\text{gch}} \bullet S_1) \circ S_2, \tag{6.6}$$

where I_{gch} is a green channel image, S_1 and S_2 are disk shaped structuring elements with radius 25 and 30 pixels respectively. Then the background corrected image is

$$I_{\text{enhe}} = I_{\text{gch}} - I_{\text{cof}} \tag{6.7}$$

The image I_{enhe} revealed a high level of contrast between the exudates and background while other features were significantly suppressed as desired.

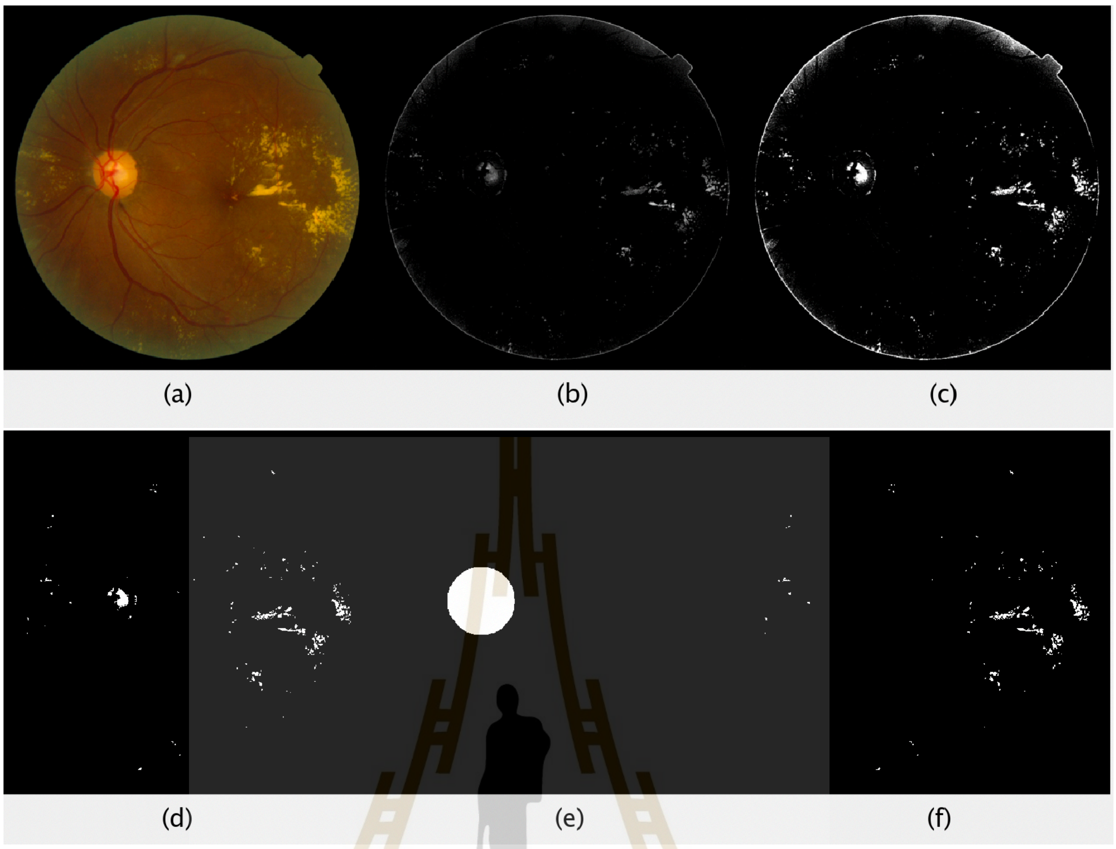


Figure 6.10 Exudates detection. (a) Original image. (b) Exudates highlighted on green component after background illumination correction by close-open filtering. (c) Lower intensity pixel values suppressed. (d) Binary image containing exudates and optic disk. (e) Optic disk mask. (f) Exudates detected after removing OD region by OD mask in (e).

Further structure enhancement was made on the image in (6.7) by the application of a homomorphic filter, and the resulting image is denoted as I_{hf}. On I_{hf} we suppress the objects with pixel values less than 50 since exudates usually exhibit higher intensity levels. That is, pixels of intensity below 50 are given a zero intensity. To denoise the image and remove background artefacts, we labelled the connected components and examined its mean intensity in the green plane. The regions with mean intensity value less than 51 were removed. Also, the connected components whose area is less than 5 pixels were discarded since they are usually the noise.

The binarized image still contains the optic disc region since it has similar intensity values with exudates. Then the optic disc is removed using the optic disc mask created from Figure 6.11. The optic disc segmentation is explained in the next section.

6.7 Optic disc segmentation

The optic disk is seen as a bright circular shaped object and it is an entrance to the major blood vessels supplied to the retina. It carries over a million afferent nerve fibers from retina to the brain. It displays one of the highest intensity values besides exudates in the retinal fundus photographs.

For optic disk detection we first colour normalize the input image as in the equation (6.1). But in this procedure, we decrease the mean value μ of the image f (converted to type double in the [0, 1] range in computation) by subtracting the value of 0.5. This has the outcome of significantly suppressing regions with intensity values below the optic disk's intensity. Then we extracted the red channel from the colour normalized image since this channel possesses higher contrast around the optic disk region. The extracted image was then thresholded and morphologically closed to maintain the connectedness. The binarized image was further filtered labelling connected components. Each component was then examined for the total area and circularity. The conditions set are such that Area $\in [3500, 15000]$ pixels, and the Circularity $\geq$ 0.7. Then the Canny edge detection method was employed to get the round edge of the remaining components. Finally, a circular Hough transform was applied with radius specification given in the range of 45 to 65 pixels. This transformation detected a disk shaped region, which was the optic disk to be segmented.

6.8 Feature extraction

In classification of DR detection, we have used the following set of features as input vectors:

1. Area of retinal blood vessels, which is the sum of non-zero pixels in a segmented image from blood vessels.

2. Area of haemorrhages.

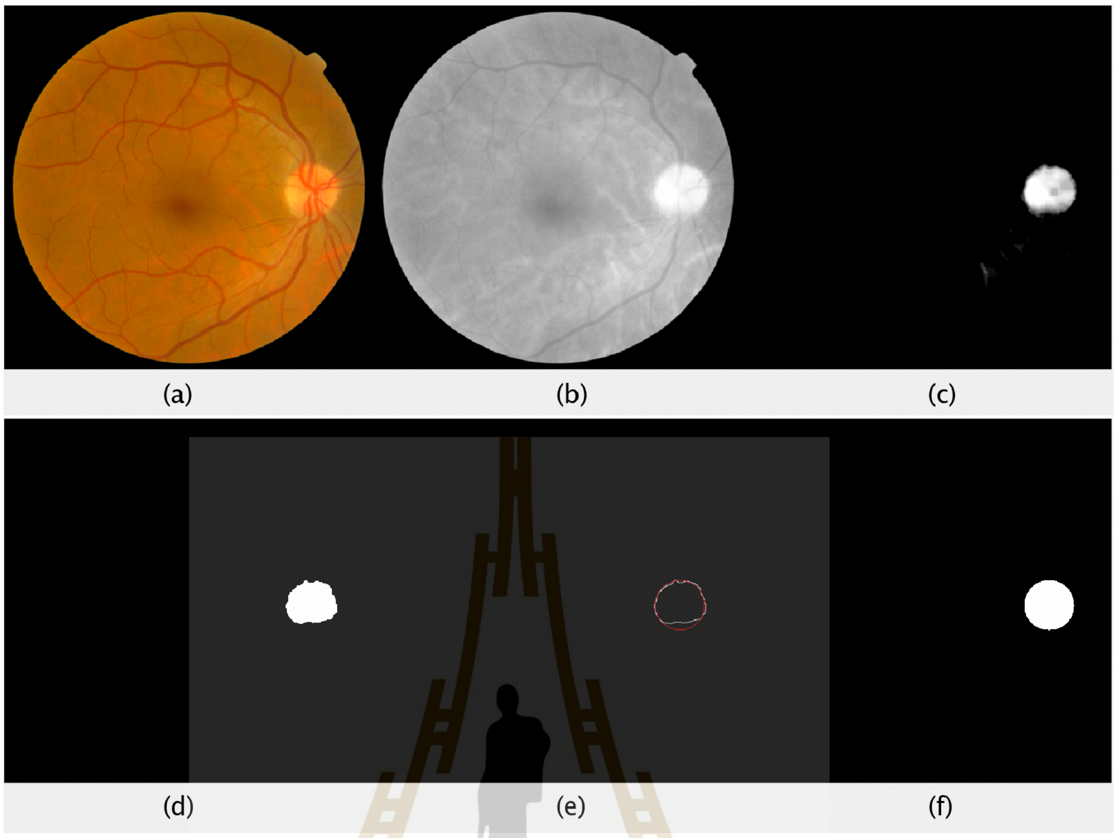


Figure 6.11 Optic disk segmentation. (a) Original image. (b) Red channel image. (c) Background suppressed (b) after colour normalization. (d) Binarized and morphologically closed image. (e) Canny edge detection and circular Hough transform. (f) OD mask.

3. Number of microaneurysms.
4. Area of microaneurysms.
5. Area of exudates.

In addition, we have also employed the following set of statistical texture features and GLCM features based on the green channel image:

6. Average intensity of the image.
7. Average contrast of the image.
8. Smoothness.
9. Third moment.

10. Energy.

11. Entropy.

12. Homogeneity.

13. Maximum probability.

14. Entropy from GLCM.

CHAPTER VII

RESULTS AND DISCUSSION

In this chapter, we present the results from different classification models, namely, SVM, twin SVM, twin bounded SVM, least squares TWSVM, and improved least squares TWSVM, on the detection of non proliferative diabetic retinopathy. We subdivide the results in two sections; (i) binary classifications for detection of severe NPDR and (ii) multiclass classifications concerned with grading different stages of NPDR.

7.1 Preliminaries

We begin by discussing the choices of models and model parameters.

7.1.1 Model Selection

Initial experiments showed that the least squares support vector machine (LSSVM) exhibited lower performance than all the other variations of the support vector machine, and was therefore excluded from further experiments. In addition, the Gaussian (RBF) kernel showed superior performance over the linear model or the polynomial kernels. Therefore, only the Gaussian kernel was considered throughout. This kernel, as indicated in Chapter 3, is of the form

$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{y}||^2}{2\sigma^2}\right)$$

where σ is the kernel parameter.

7.1.2 Validation

To evaluate the performance of the machine learning models and to overcome the problem of over-fitting, one commonly uses a technique called *cross validation*. In k-fold cross validation, the data is randomly partitioned into k equal folds. From the k

folds, the first $k-1$ folds are used for training the model and the retained k-th fold is used to test the model. This procedure is then repeated k times; each time a different fold is chosen as testing fold. The performance measures of the model, such as accuracy etc., is taken as the average values over the k runs. This procedure ensures that each sample in the dataset appears in training and testing. Moreover, each sample is used for testing exactly once.

Each of the results listed in the tables of this chapter is the average over ten runs of 10-fold validation.

7.1.3 Parameter Selection

The SVM classifiers used in this research heavily depend on the parameters that are chosen a priori. Therefore, we optimized the parameters for better performance using the grid search technique.

In the regular SVM there are two parameters, namely, the penalty parameter C and kernel parameter σ. But for the TWSVM and LSTSVM, there are now penalty parameters C_1 and C_2, as well as the kernel parameter σ. Furthermore, the TBSVM and ILSTSVM have five parameters each which include four penalty parameters C_1, C_2, C_3, C_4 and the kernel parameter σ. To reduce the computational complexity in selecting parameters, we set $C_1 = C_2$ and $C_3 = C_4$ throughout. For all the classifiers used, the penalty parameters searched for came from the set $\{2^i \mid i = -20, -19, \ldots, 15\}$ and the kernel parameter was chosen from the set $\{2^i \mid i = -10, -9, \ldots, 10\}$.

Also, since the features in the dataset that we have obtained have varying numerical ranges, they were rescaled to the range of $[0, 1]$ using the following formula:

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} ,$$

where

- x_i is the i-the data.
- z_i is the i-th normalized data.

- min(x) and max(x) are the minimum and maximum values in the dataset.

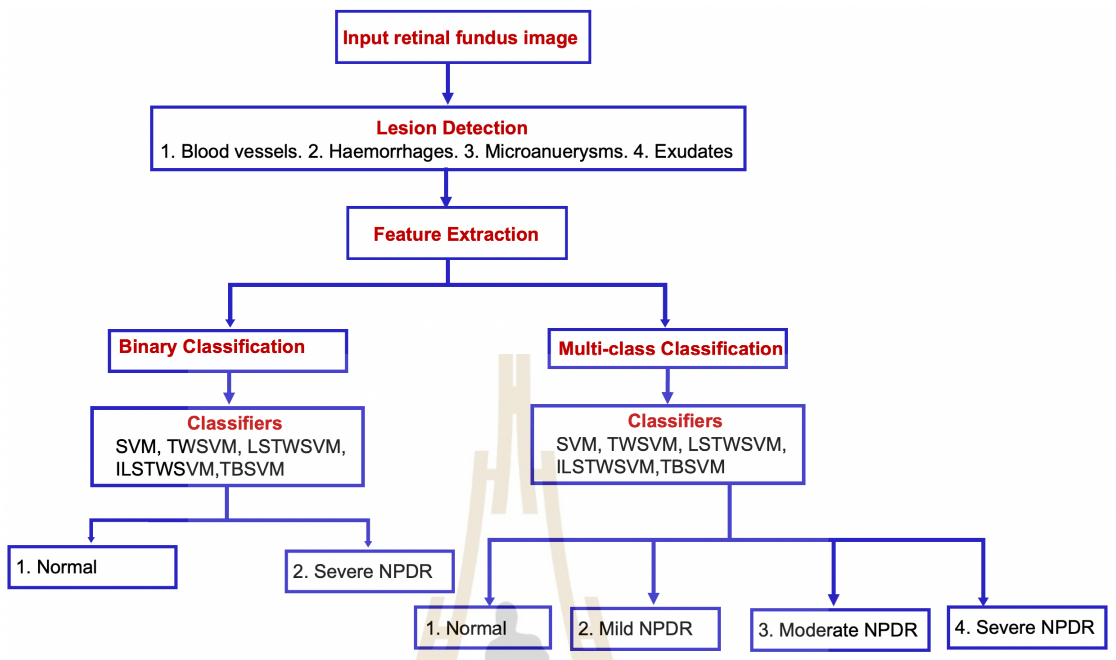

Input retinal fundus image
Lesion Detection
1. Blood vessels. 2. Haemorrhages. 3. Microanuerysms. 4. Exudates
Feature Extraction
Binary Classification
Multi-class Classification
Classifiers
SVM, TWSVM, LSTWSVM, ILSTWSVM,TBSVM
Classifiers
SVM, TWSVM, LSTWSVM, ILSTWSVM,TBSVM
1. Normal
2. Severe NPDR
1. Normal
2. Mild NPDR
3. Moderate NPDR
4. Severe NPDR

Figure 7.1 Flowchart of the proposed methodology.

7.2 Detection of severe nonproliferative diabetic retinopathy

Here the main idea is to separate severe NPDR images from the normal ones. For this binary classification problem, all 152 normal images and all 149 severe NPDR images were used.

7.2.1 Feature selection

Besides the selection of optimized parameters, we also optimized for the number of input features. In the case of binary classification we experimentally identified 7 from the total of 13 extracted features that gave optimal results and were therefore used. These are the the number and area of microaneurysms, the area of haemorrhages, the area of exudates, entropy, the third moment of the green channel image, and finally the measure of randomness computed from the gray-level co-occurrence matrix (GLCM).

Parameter optimization was performed with respect to two metrics: accuracy and sensitivity.

7.2.2 Accuracy optimization

In accuracy optimized classifications, we have employed different variants of SVM for this problem as shown in Table 7.1; all metrics are reported in % numbers. All the performance measures are based on 10-fold cross validation. From the table, one can see that the highest accuracy of 97.44 ± 2.91% was achieved by the regular SVM in comparison to twin SVMs. On the other hand, when it comes to the specificity and precision, the TBSVM obtained the highest rates of 98.68 ± 3.12% and 98.71 ± 2.99% respectively. Of all these different SVMs, the LSTSVMs achieved the least performance measures.

7.2.3 Sensitivity optimization

In medical diagnosis problems, it is most important to catch as many of those infected as possible. A false positive result is more acceptable than a false negative result, as positive results are usually screened further by a specialist. On the other hand, a false negative result means that an infected person remains undetected. That is, instead of accuracy one needs to maximize the ratio

$$\frac{\text{Number of positive results among the infected}}{\text{Total number of infected}}$$

which is sensitivity.

In our case, this means that the machine learning models should not classify eye fundus photographs that show signs of disease as healthy eye images. Therefore, for this binary problem we alternatively optimized the parameters so that the model gives better performance in terms of sensitivity. The results from the different SVMs are shown in Table 7.2. It can be seen that the twin bounded SVM has achieved the highest sensitivity of 99.06 ± 2.52%, whereas accuracy, precision and specificity measures dropped substantially. The least drop was observed with the LSTSVM and ILSTSVM which now have become the leader in all categories other than sensitivity, but still showed good sensitivity.

Table 7.1 Accuracy optimized binary classification for NPDR detection.

	SVM (C, σ)	TWSVM $(C_1 = C_2, \sigma)$	TBSVM $(C_1 = C_2$ $C_3 = C_4, \sigma)$	LSTSVM $(C_1 = C_2, \sigma)$	ILSTSVM $(C_1 = C_2$ $C_3 = C_4, \sigma)$
Parameters	$2^{12}, 2^{0}$	$2^{1}, 2^{-1}$	$2^{0.75}, 2^{-8}, 2^{-2}$	$2^{-6}, 2^{0}$	$2^{-6}, 2^{-12.25}, 2^{-1}$
Accuracy	97.44 ± 2.91	96.44 ± 3.41	96.87 ± 2.71	95.25 ± 4.14	96.11 ± 3.42
Sensitivity	96.65 ± 4.52	95.31 ± 5.86	95.04 ± 4.57	92.69 ± 7.76	95.78 ± 6.06
Specificity	98.22 ± 3.75	97.57 ± 4.46	98.68 ± 3.12	97.75 ± 4.03	97 ± 4.81
Precision	98.3 ± 3.48	97.65 ± 4.19	98.71 ± 2.99	97.79 ± 3.85	97.13 ± 4.41

Table 7.2 Sensitivity optimized binary classification for NPDR detection.

	SVM (C, σ)	TWSVM $(C_1 = C_2, \sigma)$	TBSVM $(C_1 = C_2$ $C_3 = C_4, \sigma)$	LSTSVM $(C_1 = C_2, \sigma)$	ILSTSVM $(C_1 = C_2$ $C_3 = C_4, \sigma)$
Parameters	$2^{-4}, 2^{9.75}$	$2^{-6}, 2^{7}$	$2^{-6}, 2^{-13.5}, 2^{7.5}$	$2^{-8.5}, 2^{1.5}$	$2^{-7.25}, 2^{-11}, 2^{0.5}$
Accuracy	90.17 ± 4.8	91.86 ± 4.64	91.53 ± 4.77	93.52 ± 4.21	94.05 ± 4.19
Sensitivity	98.35 ± 3.35	98.8 ± 2.74	99.06 ± 2.52	98.6 ± 2.92	98.66 ± 3.01
Specificity	82.55 ± 8.94	85.05 ± 9.02	84.18 ± 9.53	88.53 ± 7.99	89.53 ± 7.4
Precision	85.1 ± 6.89	87.17 ± 6.8	86.56 ± 7.26	89.89 ± 6.52	90.63 ± 6.2

7.3 Grading of nonproliferative diabetic retinopathy

Since NPDR has three stages as discussed before, we performed multiclass classification using the same SVM classifiers as in the binary case. We adopted the classification strategies introduced in Chapter 3, namely, one-versus-all and one-versus-one for all types of SVMs, and in addition, all-versus-one for the twin SVMs. In addition, we also used the novel minimum average distance strategy for the twin SVMs.

7.3.1 Multiclass performance metrics

The notions of accuracy, sensitivity, specificity and precision can also be defined in multiclass decision making. To describe these, we let

- S_i denote the number of samples in class i,
- S denote the total number of samples, so that $S = \sum_i S_i$,
- S_{ij} denote the number of samples in class i that have been classified to belong to class j, so that $S_i = \sum_j S_{ij}$.

In the confusion matrices of Figure 7.2, S_{ij} is the ijth matrix entry, S_i is the sum over all entries in the i-th row, and S is the sum of all matrix entries.

Accuracy

There are two ways to measure accuracy, which we will label *accuracy-1* and *accuracy-2*.

- accuracy-1 is measured over the whole dataset. That is

$$\text{accuracy-1} = \frac{\text{Number of correctly classified samples}}{\text{Total number of samples}}$$

In the confusion matrices of Figure 7.2, this would mean

$$\text{accuracy-1} = \frac{\text{sum of diagonal entires}}{\text{sum of all matrix entries}}.$$

To be precise,

$$\text{accuracy-1} = \frac{\sum_i S_{ii}}{\sum_{i,j} S_{ij}}.$$

This type of accuracy is used in the papers of Tomar and Agarwal (2015) and Ali et al. (2021) as well as in Chapter 3.

- accuracy-2 averages over the accuracies of each individual class. Just as in each run in a one-versus-all type classification, the accuracy of the i-th class versus the rest is

$$\text{Acc}_i = \frac{TP + TN}{TP + TN + FP + FN} = \frac{S_{ii} + \sum_{j \neq i} \sum_{k \neq i} S_{jk}}{S}$$

and averaging over all classes,

$$\text{accuracy-2} = \frac{1}{N} \sum_i \text{Acc}_i,$$

where N denotes the total number of classes. This measure of accuracy is used in the paper by Carrera et al. (2017).

Sensitivity, Specificity and Precision

The remaining metrics are obtained similarly, averaging over the one-versus-all metrics:

- The *sensitivity* of the i-th class versus the rest is

$$\text{Sens}_i = \frac{TP}{TP + FN} = \frac{S_{ii}}{S_i},$$

and averaging over all classes,

$$\text{Sensitivity} = \frac{1}{N} \sum_i \text{Sens}_i.$$

- The *specificity* of the i-th class versus the rest is

$$\text{Spec}_i = \frac{TN}{TN + FP} = \frac{\sum_{j \neq i} \sum_{k \neq i} S_{jk}}{\sum_{j \neq i} S_j}$$

and averaging over all classes,

$$\text{Specificity} = \frac{1}{N} \sum_i \text{Spec}_i.$$

- The *precision* of the i-th class versus the rest is

$$\text{Prec}_i = \frac{TP}{TP + FP} = \frac{S_{ii}}{\sum_j S_{ji}}$$

and averaging over all classes,

$$\text{Precision} = \frac{1}{N}\sum_i \text{Prec}_i.$$

All the above metrics are be reported in % values to conform with the literature.

7.3.2 Feature selection

In the multiclass classification, we also optimized for the number of features from the 13 extracted features. Through experiments, we obtained optimal results when supplying 11 of the 13 features to the classifiers. These are the number of microaneurysms, the areas of microaneurysms, of haemorrhages and of exudates, energy, homogeneity, maximum probability, average intensity, average contrast, smoothness and the third moment of the green channel.

Parameters were optimized for a maximum in the accuracy-1 performance.

7.3.3 Classification results

The classification results are shown in Table 7.3 and Table 7.4. It can be observed that all the performance measures of the various classification models degrade as compared to the binary classifications.

One-versus-one results

From the one-versus-one classification results in Table 7.3 one notices that the TBSVM, when coupled with the combined maximum-votes and minimum average distance tiebreaker (mv-md) decision strategy, obtains the highest score in both, accuracy-1 and accuracy-2 at rates of $75.18{\pm}5.52$% and $87.59{\pm}0.31$%, respectively. Other measures like sensitivity, specificity and precision were also highest for this machine in comparison to the other methods. Similarly, for the TWSVM, when adding the minimum average distance method as a tie breaker in the maximum votes strategy (mv-md versus mv), there is a marginal increase in the performance. Overall the LSTSVM gave the least performance rate with an accuracy-1 of $71.35 \pm 5.48\%$ and sensitivity of $50.97 \pm 1.53\%$. It can be noticed, however, that for the two least squares twin SVMs (LSTSVM and ILSTSVM), classification by the minimum average distance (md) gave noticeably better performance than the maximum-votes (mv) strategy.

We illustrate this with the confusion matrices for TBSVM (mv+md) and LSTSVM in Figure 7.2. It is clear from these matrices that the TBSVM performs fairly better in classifying normal and severe NPDR images than the LSTSVM. Yet, most of the images in the mild class and nearly half of the images in the moderate class are being misclassified as normal with the TBSVM. The situation further deteriorates in the LSTSVM as even 17 of the 149 severe images are classified as normal, in addition to misclassified mild and moderate images.

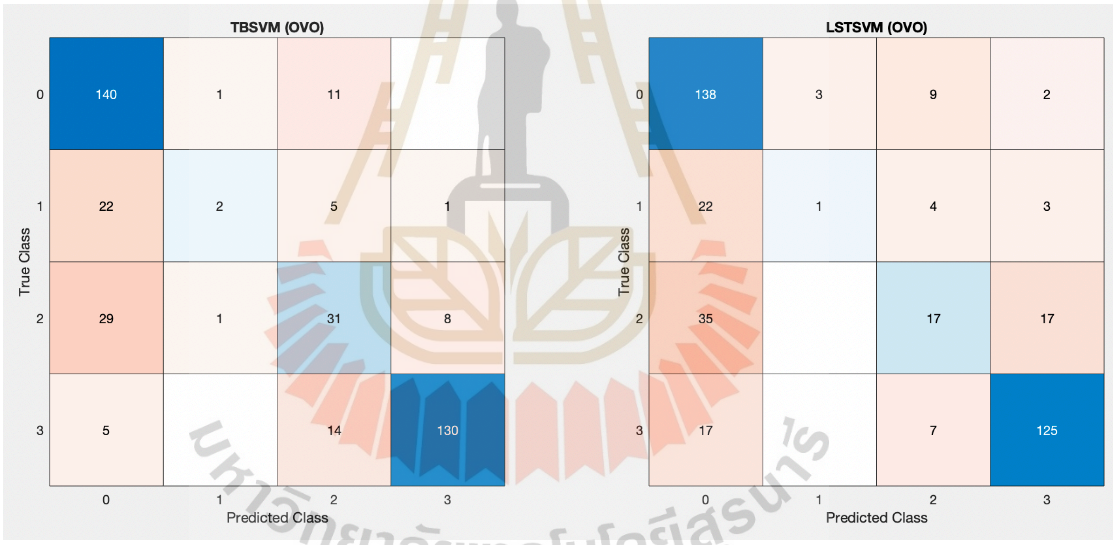


Figure 7.2 Confusion matrices from classifications using TBSVM and LSTSVM.

One-versus-all and All-versus-one results

For one-versus-all and all-versus-one classifications, the results from different machines are not significant. Similar to one-versus-one classifications, SVM, TWSVM and TBSVM gave better performance in general.

Table 7.3 Multiclass classification (one-versus-one).

Classifier	Parameters $C_1 = C_2, C_3 = C_4, \sigma$	Accuracy-1	Accuracy-2	Sensitivity	Specificity	Precision
SVM	$2^8, , 2^{-1}$	74.55 ± 4.98	87.4 ± 0.15	56.39 ± 0.47	90.58 ± 0.11	63.32 ± 1.81
TWSVM (mv)	$2^0, , 2^1$	73.57 ± 5.37	86.79 ± 0.31	54.15 ± 0.56	89.96 ± 0.21	60.51 ± 2.29
TWSVM (mv+md)	$2^{-0.25}, , 2^1$	74.73 ± 5.45	87.36 ± 0.28	56.33 ± 0.73	90.55 ± 0.22	64.29 ± 2.4
TBSVM (mv)	$2^0, 2^{-16}, 2^1$	74.05 ± 4.84	87.03 ± 0.42	55.44 ± 1	90.29 ± 0.3	63.75 ± 4.06
TBSVM (mv+md)	$2^0, 2^{-16}, 2^1$	75.18 ± 5.52	87.59 ± 0.31	56.51 ± 0.8	90.71 ± 0.23	66.37 ± 5.48
LSTSVM (mv)	$2^{-5}, , 2^0$	71.35 ± 5.48	85.68 ± 0.74	50.97 ± 1.53	88.98 ± 0.54	53.25 ± 6.24
LSTSVM (md)	$2^{-4.75}, , 2^{-0.25}$	73.15 ± 5.63	86.58 ± 0.41	53.91 ± 0.89	89.93 ± 0.29	52.56 ± 3.01
ILSTSVM (mv)	$2^{-5}, 2^{-19}, 2^1$	73.03 ± 4.58	86.51 ± 0.46	52.67 ± 0.93	89.74 ± 0.32	55.98 ± 6.07
ILSTSVM (md)	$2^{-5}, 2^{-19}, 2^1$	74.33 ± 4.67	87.06 ± 0.46	55.13 ± 1.02	90.46 ± 0.31	56.03 ± 8.37

Table 7.4 Multiclass classification (one-versus-all and all-versus-one).

Classifier	Parameters $C_1 = C_2, C_3 = C_4, \sigma$	Accuracy-1	Accuracy-2	Sensitivity	Specificity	Precision
			one-versus-all			
SVM	$2^{10}, , 2^{-1.25}$	73.38 ± 4.67	86.69 ± 0.48	53.83 ± 1.03	89.85 ± 0.34	60.46 ± 4.02
TWSVM	$2^{-1}, , 2^{0}$	73.57 ± 5.07	86.79 ± 0.39	54.48 ± 1.11	90.03 ± 0.3	60.8 ± 4.38
TBSVM	$2^{-1}, 2^{-13}, 2^{0}$	73.88 ± 5.03	86.94 ± 0.47	55 ± 0.8	90.16 ± 0.31	62.13 ± 3.53
LSTSVM	$2^{-1}, , 2^{-3}$	68.4 ± 6.04	84.2 ± 0.53	50.71 ± 1.09	88.35 ± 0.36	51.86 ± 1.55
ILSTSVM	$2^{-1}, 2^{-2}, 2^{-3}$	71.35 ± 5.12	85.68 ± 0.35	51.2 ± 0.69	88.8 ± 0.26	67.5 ± 0.62
			all-versus-one			
TWSVM	$2^{1}, , 2^{-1}$	70.83 ± 4.78	85.41 ± 0.33	52.54 ± 0.72	89.05 ± 0.24	58.5 ± 2.67
TBSVM	$2^{1}, 2^{-5}, 2^{-2}$	72.43 ± 5.45	86.21 ± 0.44	53.42 ± 0.83	89.5 ± 0.32	61.32 ± 3.28
LSTSVM	$2^{-3}, , 2^{-1}$	68.43 ± 5.23	84.21 ± 0.56	47.55 ± 1.18	87.77 ± 0.49	50.6 ± 4.45
ILSTSVM	$2^{5}, 2^{-2}, 2^{-4}$	71.95 ± 6.29	85.98 ± 0.36	53.77 ± 0.67	89.39 ± 0.26	59.83 ± 2.01

7.3.4 Parameter influence on accuracy

We compare how changes of the penalty parameters and the RBF kernel parameter σ changes the predictive accuracy rate for some of the machines.

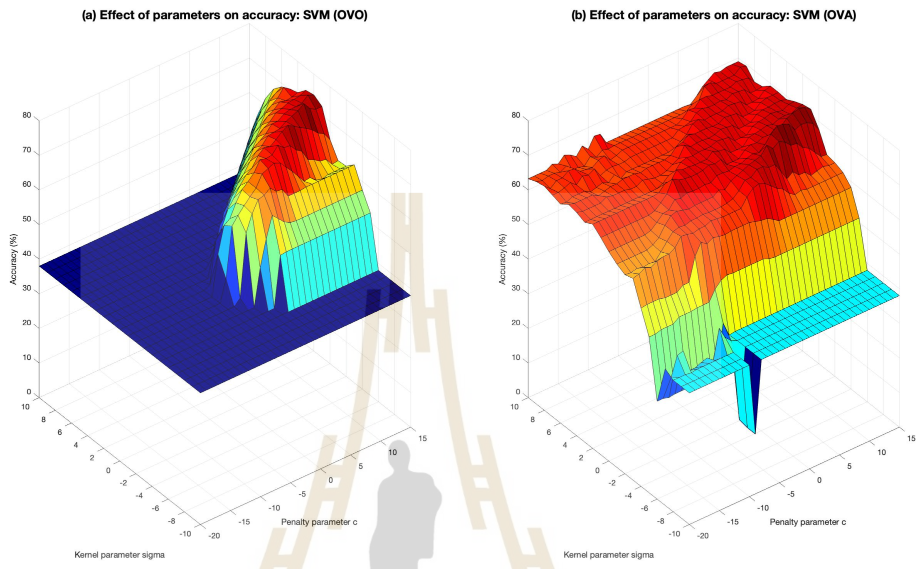


Figure 7.3 Effect of (C, σ) on accuracy of SVM.

From Figure 7.3 we see the influence of the penalty parameter C and the kernel parameter σ on the one-versus-one and one-versus-all SVM recognition rates. In both cases, the graphs show that the penalty parameter has more of an influence on the accuracy than the kernel parameter. High values of C are required for good accuracy, while the kernel parameter σ remains in a medium range. In one-versus-one, the accuracy remains constantly low for small C values and then rapidly increases as C becomes larger. However, in one-versus-all the accuracy shows gradual improvement as C goes higher.

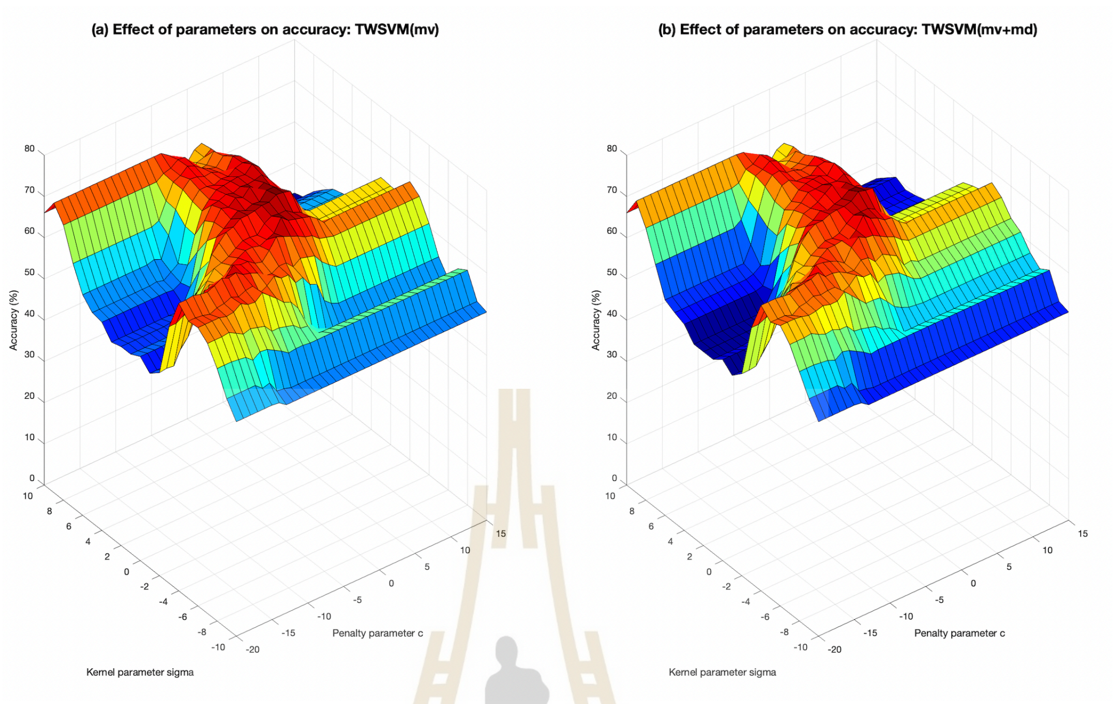


Figure 7.4 Effect of $(C_1 = C_2, \sigma)$ on accuracy of TWSVM (OVO).

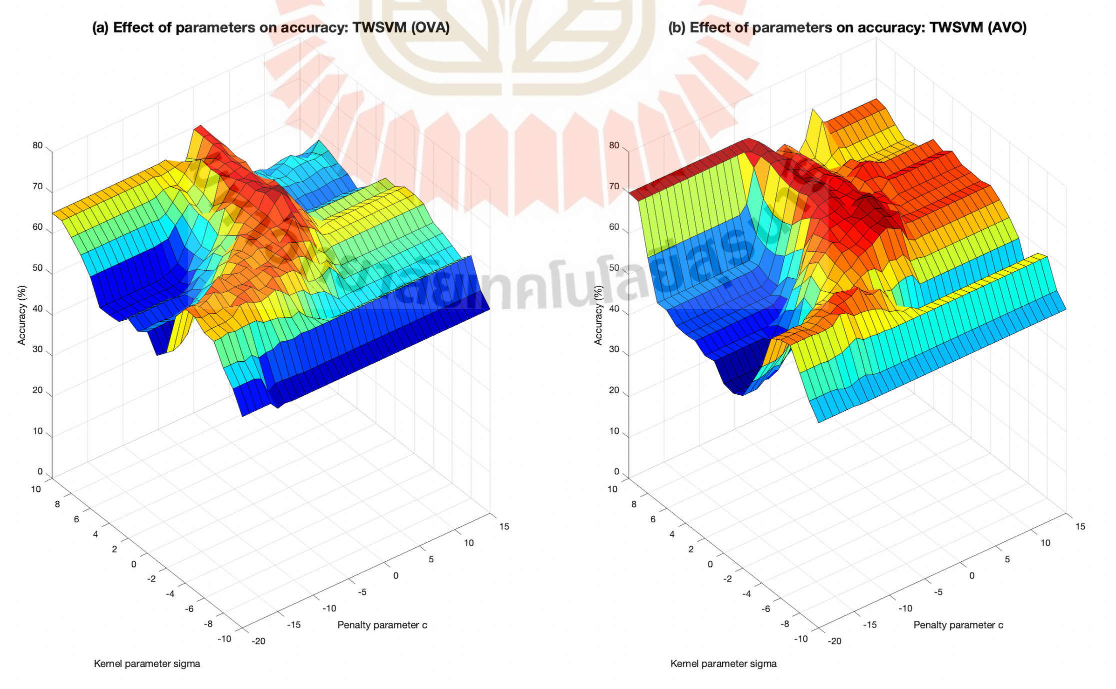


Figure 7.5 Effect of $(C_1 = C_2, \sigma)$ on accuracy of TWSVM (OVA and AVO).

For the TWSVM, from Figures 7.4 and 7.5 it can be seen that the situation is reversed: relatively low values of the penalty parameters and kernel parameter give good accuracy, and the results are similar for all the different decision strategies. In contrast to the SVM, the accuracy suddenly degrades as the value of $C_1 = C_2$ gets higher. However, one can notice a slight improvement in the accuracy rate when using the minimum average distance to break the tie in votes from maximum-votes strategy (mv-md).

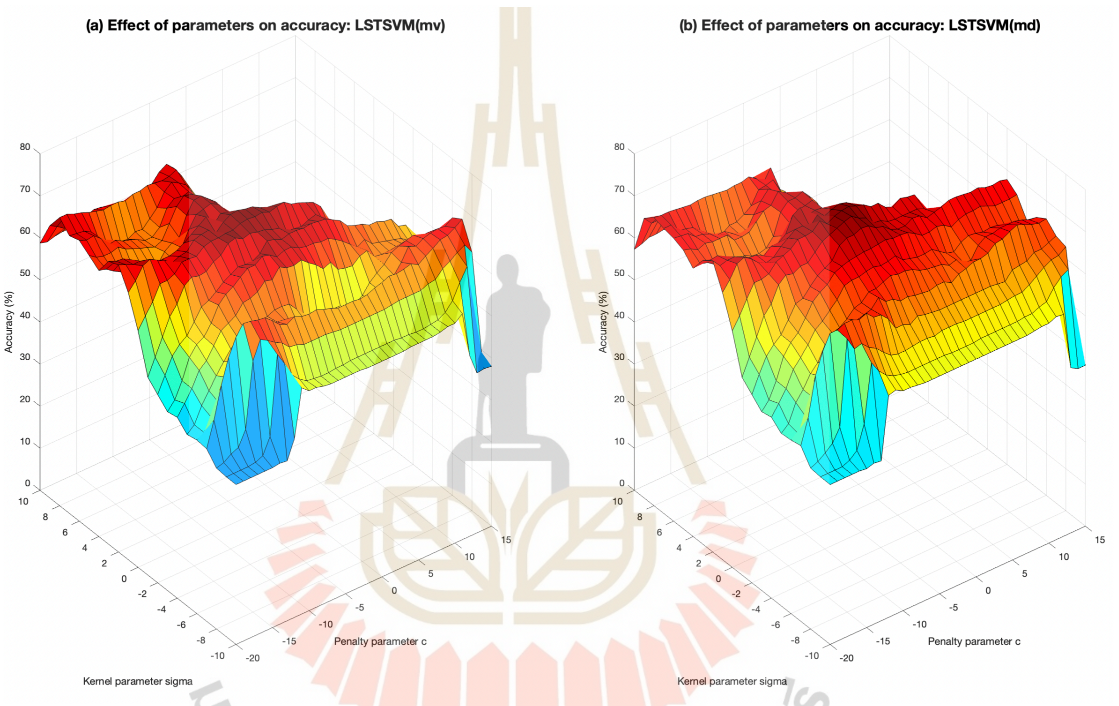


Figure 7.6 Effect of $(C_1 = C_2, \sigma)$ on accuracy of LSTSVM (OVO).

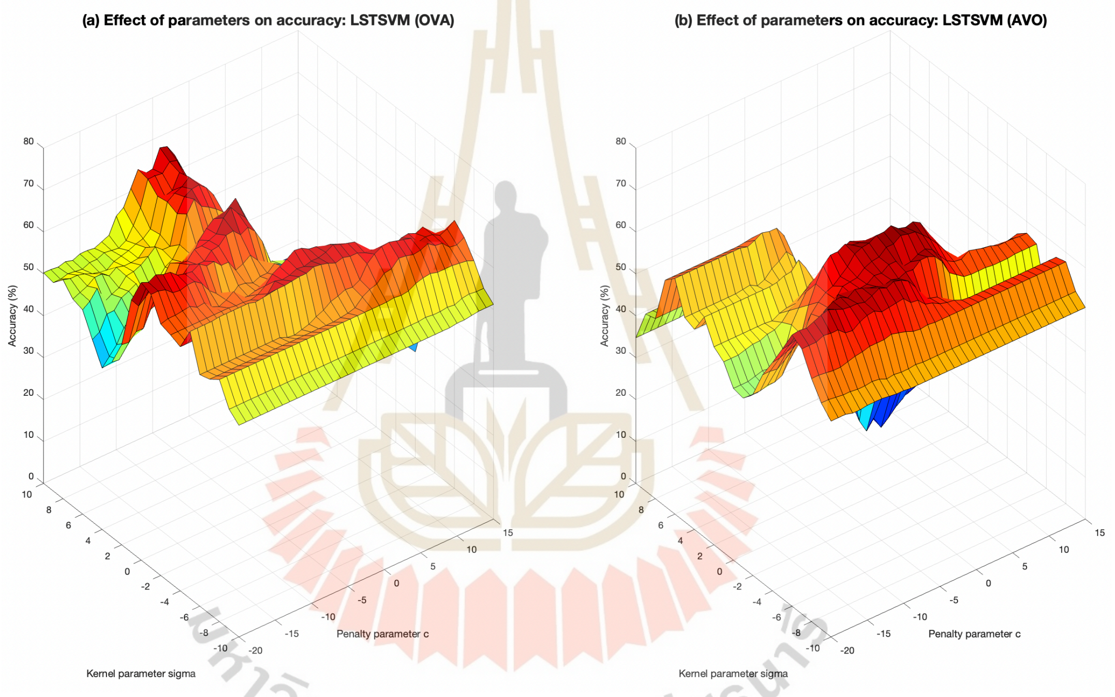


Figure 7.7 Effect of $(C_1 = C_2, \sigma)$ on accuracy of LSTSVM (OVA and AVO).

In case of the LSTSVM, from Figures 7.6 and 7.7 one-versus-one and all-versus-one seem to require relatively low values of $C_1 = C_2$ and σ for good accuracy. The surface remains flat for the one-versus-one case, and we notice some improvement in the overall rate of accuracy when using our new method of classifying by minimum average distance as seen in Figure 7.6 (b). For one-versus-all, a higher penalty parameter gives better performance.

In all cases, performance is best when the kernel parameter σ is in the neighborhood of 2^0. An exception are the least square twin SVMs, where σ may be as low as 2^{-4} which can be seen from Tables 7.3 and 7.4.

7.3.5 Unbalanced dataset mitigation

Since the dataset of 400 images contains only 30 mild and 69 moderate samples in contrast to the 152 normal and 149 severe samples, there is some data imbalance. Synthetic minority over-sampling technique (SMOTE) is a standard technique to create additional samples for the minority classes. Expecting to improve the model's performance we employed the SMOTE technique to deal with this imbalanced data problem. After obtaining a total of 152 samples for each of the image classes, we performed parameter optimization and classification by SVM and TWSVM with one-versus-one, maximum-vote strategy. The confusion matrix in Figure 7.8 shows the results.

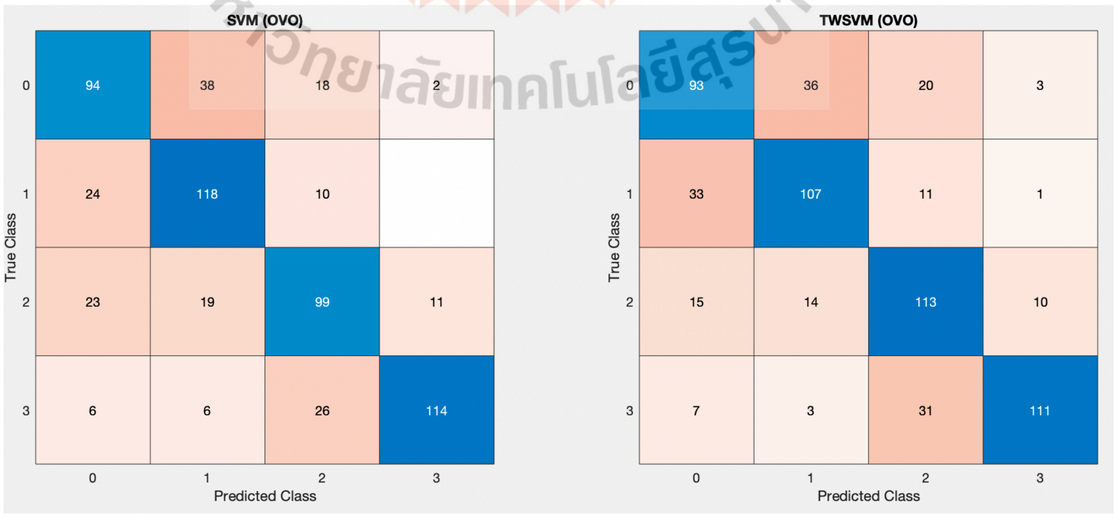


Figure 7.8 Confusion matrices from classifications using SVM and TWSVM.

From the SVM-based confusion matrix which is the better of the two, one obtains the following measurements:

- Accuracy: 69.90%
  (For a balanced dataset, accuracy-1 and accuracy-2 and sensitivity coincide.)
- Sensitivity: 69.90%
- Specificity: 89.97%
- Precision: 70.90%

From this method, the rates of sensitivity and precision improve substantially to about 70% overall, but the rate of accuracy drops noticeably. Moreover, there is now the problem that a large portion of the normal images have been dragged into the mild and moderate classes. Assuming that in practice, most of the eyes examined will be healthy eyes, this will result in a large number of patients requiring an unnecessary second examination by an expert, and makes this method of automatic diagnosis inefficient. The problem is that many of the images in the mild or even moderate category are barely distinct from normal images, and oversampling further blurs this distinction.

7.4 Comparison with results in the literature

Carrera et al. (2017) have used the Messidor database as well and performed binary classifications using regular SVM involving normal and severe NPDR graded images as shown in Table 7.5. They have reflected the rates of accuracy and sensitivity, respectively, as 92.4% and 94.6% from 10-fold cross validation. However, from their confusion matrix one can conclude that their results are based on a single run of a 10-fold cross validation. On the other hand, our results are based on 10 runs of 10-fold cross validation, and we are reporting the average metrics. Therefore, we believe that our performance measures are more robust and reliable. The authors also reported binary classification results using a decision tree (DT). Whether optimized for accuracy or sensitivity, our methods show better results.

They have also evaluated and reported their performance measures of multiclass classifications based on a single run of 10-fold cross validation as given in Table 7.6. They have obtained an accuracy-2 of 85.1%, sensitivity of 49.4%, and specificity of 88.4%. Again, we have achieved better results in all performance evaluations.

We believe that our superior results are mainly due to the careful preprocessing and feature extraction from the images.

7.5 Discussion

In our binary classifications, all types of SVMs produced good results. Twin support vector machine models do not show any advantage over the regular support vector machine, but are close behind. The accuracy of 97.44% achieved by the regular support vector machine compares favorably with other results in the literature. In addition, least-square based models fall somewhat behind the other models. However, by introducing the regularization term, the improved least squares twin SVM (ILSTSVM) comes close to the twin SVM in performance.

On the other hand, when it comes to sensitivity optimization, the least squares twin SVM and improved least squares twin SVM show the most balanced picture regarding all the four performance metrics.

In multiclass problems the overall performance is noticeable lower than for the binary classification. We should note, however, that in binary classification, the mild and moderate images which are the most difficult ones to classify, had been removed from the dataset. The twin bounded support vector machine (TBSVM) showed the best performance in almost all metrics. However, the usual SVM and twin SVM follow close behind. The least squares based models showed again lower performance, although the improved least squares SVM managed to catch up in some of the metrics.

One-versus-one decisions tend to be better than the one-versus-all decisions which in turn tend to be slightly better than the all-versus-one decisions.

The difficulties in the multiclass decisions are mainly due to the fact that most of the mild and moderate images are not being detected as being in the diseased class.

Table 7.5 Binary classification results comparisons with literature.

	Classifier	Accuracy	Sensitivity	Specificity	Precision
		Accuracy optimized			
(This work)	SVM	97.44 ± 2.91	96.65 ± 4.52	98.22 ± 3.75	98.3 ± 3.48
(Carrera et al.)	SVM	92.4	87.3	97.4	97.2*
(Carrera et al.)	DT	92.0	86.6	97.4	n/a
		Sensitivity optimized			
(This work)	TWSVM	91.53 ± 4.77	99.06 ± 2.52	84.17 ± 9.53	86.56 ± 7.16
(This work)	SVM	90.17 ± 4.8	98.35 ± 3.35	82.55 ± 8.94	85.1 ± 6.89
(This work)	ILSTSVM	94.05 ± 4.19	98.66 ± 3.01	89.53 ± 7.4	90.63 ± 6.2
(Carrera et al.)	SVM	80.4	94.6	66.2	n/a
(Carrera et al.)	DT	91.0	94.0	88.1	n/a

Table 7.6 Multiclass classification results comparisons with literature (one-versus-one).

	Classifier	Accuracy-1	Accuracy-2	Sensitivity	Specificity	Precision
(This work)	TBSVM (mv+md)	75.18 ± 5.52	87.59 ± 0.31	56.51 ± 0.8	90.71 ± 0.23	66.37 ± 5.48
(This work)	SVM	74.55 ± 4.98	87.4 ± 0.15	56.39 ± 0.47	90.58 ± 0.11	63.32 ± 1.81
(Carrera et al.)	SVM	70.25*	85.1	49.4	88.4	49.3*

*: not published in Carrera et al.'s paper, but easily computed from the published confusion matrix.

In mild NPDR eye images, the most significant characteristics are the microaneurysms. Their separating out is the most difficult of all the other segmentation processes since it depends on a number of factors such as image quality or the basic or advanced image preprocessing techniques one uses. As for the moderate NPDR images, haemorrhages and cotton wool spots can be observed as per clinical pathologies in general. However, the Messidor dataset that we have used contains several NPDR images labeled moderate which do not exhibit these properties. In addition, only a few microaneurysms are present in these moderate images, just as in the mild case. Thus, the distinction between these classes seems subtle. It remains a challenge to improve on the image pre-processing and feature extraction techniques in order to be better able to detect the mild and moderate cases.

CHAPTER VIII

CONCLUSION

In this research, we have presented the classification of eye fundus images with regards to the various stages of nonproliferative diabetic retinopathy using support vector machines.

In the first part, a range of support vector machines were reviewed and evaluated by experiments with regards to suitability for multiclass decisions using 5 standard datasets. These include the usual support vector machine, the twin support vector machine, the least squares twin support vector machine and improved versions of the latter two, called twin bounded, respectively improved least squares twin support vector machines. While there were noticeable differences among them in accuracy as performance metric, these usually fell in a range of well below 5%. Overall, the twin bounded least squares support vector machine showed the best accuracy. For 4 out of 5 of the datasets, the one-versus-one decision strategy had an edge over the one-versus-all and the all-versus-one strategies. Surprisingly, the all-versus-one strategy proved to be superior for the remaining dataset. It turns out that some of our results differ from those published by a group of other authors; indeed we believe that some of their results cannot be realistic.

We then proposed a new decision strategy in one-versus-one decisions for the twin support vector machine variations, replacing the “maximum-vote” strategy with a “minimum average distance from the hyperplanes” strategy. In half of the experiments this novel strategy gave a slight increase in accuracy.

The second part dealing with the fundus eye images began by delineating the various image processing techniques for the detection of retinal blood vessels, microaneurysms, haemorrhages, exudates, markers for diabetic retinopathy, as well as the optic disk used in this work. These four markers were extracted from the algorithms as the most significant features. They were supplemented by another nine features extracted based on statistical and texture content from the green channel of the image and its gray-level co-occurrence matrix. In total 13 features were extracted in total to be fed to the classifiers.

Then various support vector machine models were employed, first for detection of severe nonproliferative diabetic retinopathy detection, and next for detection and multiclass grading of nonproliferative diabetic retinopathy. For the detection of severe nonproliferative diabetic retinopathy, 7 of the 13 features gave optimal results with an accuracy rate of $97.44 \pm 2.91\%$ from a regular support vector machine. When the goal was to optimize for the sensitivity, the twin bounded support vector machine yielded the highest sensitivity of $99.06 \pm 2.52\%$. In case of nonproliferative diabetic retinopathy grading, the best performances were recorded using 11 features. The different decision strategies on regular as well as twin support vector machine models gave almost similar performances. In particular, the twin bounded support vector machine with a novel "maximum-vote and minimum average distance as the tie breaker" strategy recorded highest performance rates. This classifier realized an average accuracy of 87.59% with a standard deviation of $\pm 0.31\%$. The performance measures other than accuracy were also marginally higher on this machine when compared to the other classifiers. The two twin least square support vector machines showed the least performance of all classifiers, however, in the one-versus-one decision method, their performance was noticeably increased when our new novel "minimum average distance from the hyperplane" strategy was applied.

Our results turned out to be superior to those presented in the literature. This is most likely caused by our careful preprocessing, and the extraction of a total of 13 features from the images. Nevertheless, the detection of mild and moderate diabetic retinopathy still presents a challenge. Many of the images in the Messidor database that are classified as mild or moderate which we have used are difficult to distinguish from healthy images even with the naked eye. It would be worthwhile to investigate feature extraction methods using machine learning methods, such as convolutional neural networks for example, to explore the small specific nuances in the images as new features.

REFERENCES

REFERENCES

Abe, S. (2010). *Support Vector Machines for Pattern Classification*. Springer, 473pp.

Acharya, U.R., Chowriappa, P. and Sree, S.V. (2012). Wavelet-based energy features for glaucomatous image classification. *IEEE Transactions on Information Technology in Biomedicine. 16*: 80–87.

Acharya, U.R., Chua, C. K., Ng, E. Y. K., Yu, W. and Chee, C. (2008). Application of higher order spectra for the identification of diabetes retinopathy stages. *Journal of Medical Systems. 32*: 481–488.

Ali, J., Aldhaifallah, M., Nisar K.S., Aljabr, A.A. and Tanveer, M. (2021). Regularized least squares twin SVM for multiclass classification. *Big Data Research. 27*: 100295.

Allwein, E. L., Shapire, R. E. and Singer, Y. (2000). Reducing multiclass to binary : a unifying approach for margin classifiers. *Journal of Machine Learning Research. 1*: 113–141.

American Academy of Ophthalmology. (2019). *Fundamentals and Principles of Ophthalmology*. Basic amd Clinical Science Course 2019-2020. 546pp.

Bhardwaj, C., Jain, S. and Sood, M. (2020). Hierarchical severity grade classification of non-proliferative diabetic retinopathy. *Journal of Ambient Intelligence and Humanized Computing. 12*: 2649–2670.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press. 727pp.

Carrera, E. V., González, A. and Carrera, R. (2017). Automated detection of diabetic retinopathy using SVM. *2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*. 17286381.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology. 2*(3): 1–27.

Chen, B., Wang, L., Wang, X., Sun, J., Huang, Y., Feng, D. and Xu, Z. (2020). Abnormality detection in retinal image by individualized background learning. *Pattern Recognition. 102*: 107209.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning. 20*(3): 273–297.

Cunha-Vaz, J. (2011). *Diabetic Retinopathy*. World-Scientific. 323pp.

Dandapat, S., Ghosh, S., Si, S. and Datta, A. (2021). Analysis of diabetic retinopathy abnormalities detection techniques. *Advances in Intelligent Systems and Computing. 1255* : 235–247.

Deng, N., Tian, Y. and Zhang, Ch. (2013). *Support Vector Machines: Optimization Based Theory, Algorithms, and Extensions*. CRC Press.

Dietterich, T. F. and Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research. 2*: 263–286.

Decencière, E., Zhang, X., Gazuguel, G., Lay, B., Cochener, B., Trone, C., Gain, P., Ordonez, R., Massin, P., Erginay, A., Charton, B. and Klein, J.-C. (2014). *Feedback on a publicly distributed database: the Messidor database. 33*(3): 231–234.

Ghosh, R., Ghosh, K. and Maitra, S. (2017). Automatic detection and classification of retinopthy stages using CNN. *Proceedings of the 4th International Conference on Signal Processing and Integrated Networks (SPIN)*. 550-554.

Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2009). *Digital Image Processing Using MATLAB*. United States: Gatesmark Publishing.

Hemanth, J., Deperlioglu, O. and Kose, U. (2020). An enhanced diabetic retinopathy detection and classification approach using deep comvolutional neural network. *Neural Computing and Applications. 32*: 707–721.

Imani, E., Pourezza, H.-R. and Banaee, T. (2015). Fully automated diabetic retinopathy screening using morphological component analysis. *Computerized Medical Imaging and Graphics. 43*: 78–88.

James, J., Sharifahmadian, E. and Shih, L. (2018). Automatic severity level classification of diabetic retinopathy. *International Journal of Computer Applications. 180*(12): 30–35.

Kandhasamy, J. P., Balamurali, S., Kadry, S. and Ramasamy, L. K. (2020). Diagnosis of diabetic retinopathy using multi level set segmentation algorithm with feature extraction using SVM with selective features. *Multimedia Tools and Applications. 79*: 10581–10596.

Katada, Y., Ozawa, N., Masayoshi, K., Ofuji, Y., Tsubota , K. and Kurihara, T. (2020). Automatic screening for diabetic retinopathy in interracial fundus images using artificial intelligence. *Intelligence-Based Medicine. 3-4*: 100024.

Li, Y.-H., Yeh, N.-N., Chen,S.-J. and Chung, Y.-C. (2019). Computer-assisted diagnosis for diabetic retinopathy based on fundus images using deep convolutional network. *Mobile Information Systems. 2019*: 6143839.

Lumbroso, B., Rispoli, M. and Savastano, M. C. (2015). *Diabetic Retinopathy*. Jaypee Brothers Medical Publishers. 100pp.

Ramos-Soto, O., Rodríguez-Esparza, E., Balderas-Matta, S. E., Oliva, D., Hassanien., A. E., Meleppat, R. K. and Zawadzki, R. J. (2021). An efficient retinal blood vessel segmentation in eye fundus images by using optimized top-hat and homomorphic filtering. *Computer Methods and Programs in Biomedicine. 201*: 0169–2607..

Saho, Y.-H., Zhang, C.-H., Wang, X.-B. and Deng, N.-Y. (2021). Improvements on Twin Support Vector Machines. *IEEE Transactions on Neural Networks. 22*(6) : 962–968.

Sarki, R., Ahmed, K., Wang, H. and Zhang, Y. (2020). Automatic detection of diabetic eye disease through deep learning using fundus images: a survey. *IEEE Access. 8*: 151133–151148.

Sinthanayothin, C., Boyce, J.F., Williamson, T.H., Cook, H.L., Lal, S. and Usher, D. (2002). Automated detection of diabetic retinopathy on digital fundus images. *Diabetic Medicine. 19*: 105–112.

Suykens, J. A. K. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Processing Letters. 9*(3): 293–300.

Suykens, J. A. K., Van Gestel, T., De Brabanter, J. De Moor, B. and Vandewalle, J. (2002). *Least Squares Support Vector Machines*. World Scientific. 308pp.

Tanveer, M., Gautam, C. and Suganthan, P. N. (2019). Comprehensive evaluation of twin SVM based classifiers on UCI datasets. *Applied Soft Computing Journal. 83*: 106617.

Tanveer, M., Khan, M.A., Ho, SS. (2016). Robust energy-based least squares twin support vector machines. *Applied Intelligence. 45*(1): 174–186.

Tomar, D. and Agarwal, S. (2015). A comparison on multi-class classification methods based on least squares twin support vector machines. *Knowledge-Based Systems. 81*: 131-147.

Welikala, R.A., Fraz, M.M., Dehmeshki, J., Hoppe, A. Tah, V. Mann, S., Williamson, T.H. and Barman, S.A. (2015). Genetic algorithm based feature selection combined with dual classification for the automatic detection of proliferative diabetic retinopathy. *Computerized Medical Imaging and Graphics. 43*: 64-77.

World Health Organization. (2020). *Diabetic Retinopathy Screening: A Short Guide*. (online). Available: https://www.euro.who.int/en/publications/abstracts /diabetic-retinopathy-screening-a-short-guide-2020.

World Health Organization. (2015). *TADDS : Tool for the Assessment of Diabetic Retinopathy and Diabetes Management Systems*. (online). Available: https://www.who.int/ blindness/publications/TADDS_EN.pdf.

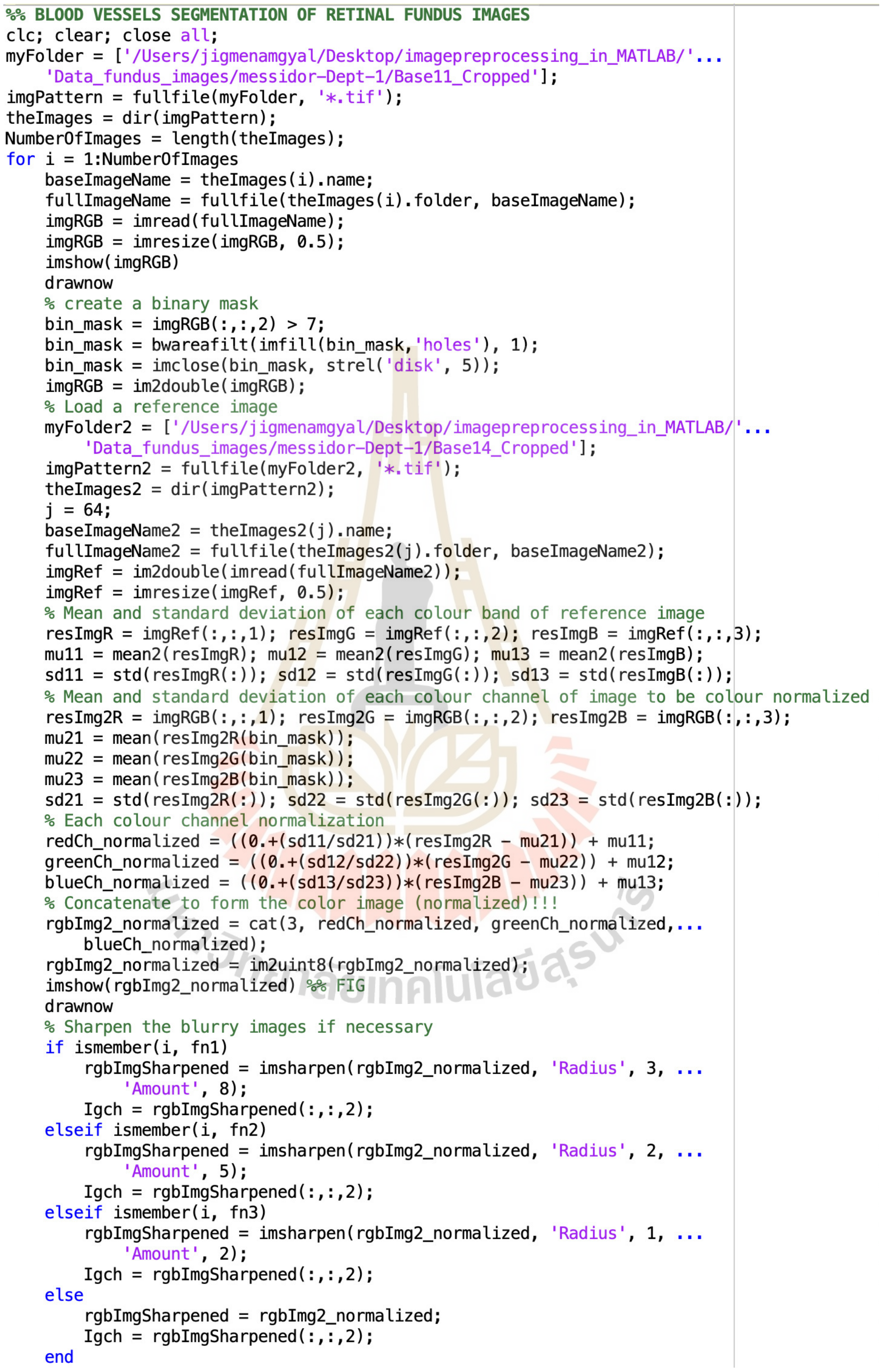
World Health Organization. (2019). *World Report on Vision*. (online). Available: https:// www.who.int/publications-detail/world-report-on-vision.

Xu, Y., Xi, W., Lv, X. and Guo, R. (2012). An improved least squares twin support vector machine. *Journal of Information & Computational Science. 9*(4): 1063–1071.

APPENDICES

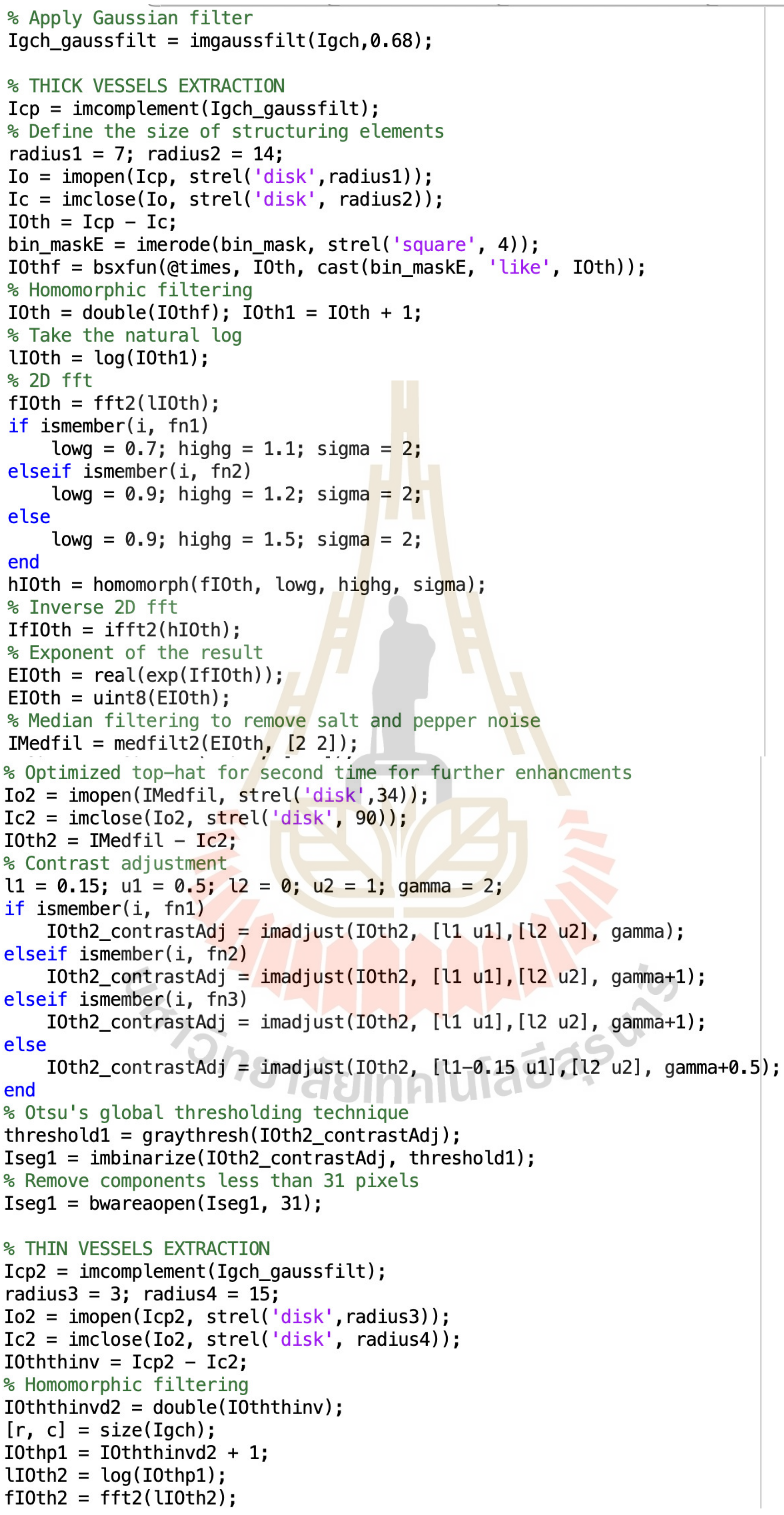
APPENDIX A

MATLAB CODES FOR PROCESSING OF THE EYE IMAGES



```
%% BLOOD VESSELS SEGMENTATION OF RETINAL FUNDUS IMAGES
clc; clear; close all;
myFolder = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_Cropped'];
imgPattern = fullfile(myFolder, '*.tif');
theImages = dir(imgPattern);
NumberOfImages = length(theImages);
for i = 1:NumberOfImages
    baseImageName = theImages(i).name;
    fullImageName = fullfile(theImages(i).folder, baseImageName);
    imgRGB = imread(fullImageName);
    imgRGB = imresize(imgRGB, 0.5);
    imshow(imgRGB)
    drawnow
    % create a binary mask
    bin_mask = imgRGB(:,:,2) > 7;
    bin_mask = bwareafilt(imfill(bin_mask,'holes'), 1);
    bin_mask = imclose(bin_mask, strel('disk', 5));
    imgRGB = im2double(imgRGB);
    % Load a reference image
    myFolder2 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base14_Cropped'];
    imgPattern2 = fullfile(myFolder2, '*.tif');
    theImages2 = dir(imgPattern2);
    j = 64;
    baseImageName2 = theImages2(j).name;
    fullImageName2 = fullfile(theImages2(j).folder, baseImageName2);
    imgRef = im2double(imread(fullImageName2));
    imgRef = imresize(imgRef, 0.5);
    % Mean and standard deviation of each colour band of reference image
    resImgR = imgRef(:,:,1); resImgG = imgRef(:,:,2); resImgB = imgRef(:,:,3);
    mu11 = mean2(resImgR); mu12 = mean2(resImgG); mu13 = mean2(resImgB);
    sd11 = std(resImgR(:)); sd12 = std(resImgG(:)); sd13 = std(resImgB(:));
    % Mean and standard deviation of each colour channel of image to be colour normalized
    resImg2R = imgRGB(:,:,1); resImg2G = imgRGB(:,:,2); resImg2B = imgRGB(:,:,3);
    mu21 = mean(resImg2R(bin_mask));
    mu22 = mean(resImg2G(bin_mask));
    mu23 = mean(resImg2B(bin_mask));
    sd21 = std(resImg2R(:)); sd22 = std(resImg2G(:)); sd23 = std(resImg2B(:));
    % Each colour channel normalization
    redCh_normalized = ((0.+(sd11/sd21))*(resImg2R - mu21)) + mu11;
    greenCh_normalized = ((0.+(sd12/sd22))*(resImg2G - mu22)) + mu12;
    blueCh_normalized = ((0.+(sd13/sd23))*(resImg2B - mu23)) + mu13;
    % Concatenate to form the color image (normalized)!!!
    rgbImg2_normalized = cat(3, redCh_normalized, greenCh_normalized,...
        blueCh_normalized);
    rgbImg2_normalized = im2uint8(rgbImg2_normalized);
    imshow(rgbImg2_normalized) %% FIG
    drawnow
    % Sharpen the blurry images if necessary
    if ismember(i, fn1)
        rgbImgSharpened = imsharpen(rgbImg2_normalized, 'Radius', 3, ...
            'Amount', 8);
        Igch = rgbImgSharpened(:,:,2);
    elseif ismember(i, fn2)
        rgbImgSharpened = imsharpen(rgbImg2_normalized, 'Radius', 2, ...
            'Amount', 5);
        Igch = rgbImgSharpened(:,:,2);
    elseif ismember(i, fn3)
        rgbImgSharpened = imsharpen(rgbImg2_normalized, 'Radius', 1, ...
            'Amount', 2);
        Igch = rgbImgSharpened(:,:,2);
    else
        rgbImgSharpened = rgbImg2_normalized;
        Igch = rgbImgSharpened(:,:,2);
    end
```

Figure A.1 Matlab code: Retinal blood vessels segmentation.

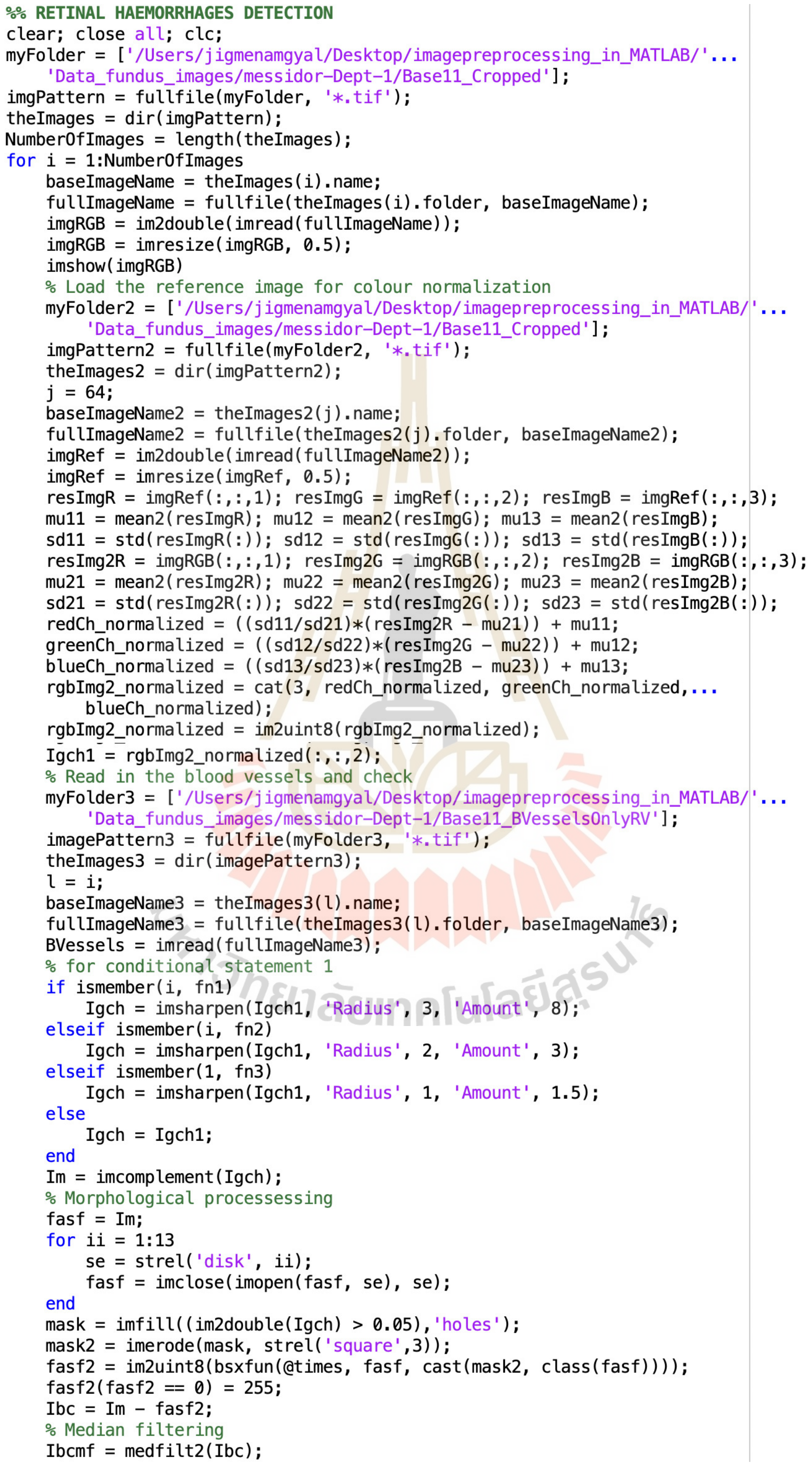


```
% Apply Gaussian filter
Igch_gaussfilt = imgaussfilt(Igch,0.68);

% THICK VESSELS EXTRACTION
Icp = imcomplement(Igch_gaussfilt);
% Define the size of structuring elements
radius1 = 7; radius2 = 14;
Io = imopen(Icp, strel('disk',radius1));
Ic = imclose(Io, strel('disk', radius2));
IOth = Icp - Ic;
bin_maskE = imerode(bin_mask, strel('square', 4));
IOthf = bsxfun(@times, IOth, cast(bin_maskE, 'like', IOth));
% Homomorphic filtering
IOth = double(IOthf); IOth1 = IOth + 1;
% Take the natural log
lIOth = log(IOth1);
% 2D fft
fIOth = fft2(lIOth);
if ismember(i, fn1)
    lowg = 0.7; highg = 1.1; sigma = 2;
elseif ismember(i, fn2)
    lowg = 0.9; highg = 1.2; sigma = 2;
else
    lowg = 0.9; highg = 1.5; sigma = 2;
end
hIOth = homomorph(fIOth, lowg, highg, sigma);
% Inverse 2D fft
IfIOth = ifft2(hIOth);
% Exponent of the result
EIOth = real(exp(IfIOth));
EIOth = uint8(EIOth);
% Median filtering to remove salt and pepper noise
IMedfil = medfilt2(EIOth, [2 2]);
% Optimized top-hat for second time for further enhancments
Io2 = imopen(IMedfil, strel('disk',34));
Ic2 = imclose(Io2, strel('disk', 90));
IOth2 = IMedfil - Ic2;
% Contrast adjustment
l1 = 0.15; u1 = 0.5; l2 = 0; u2 = 1; gamma = 2;
if ismember(i, fn1)
    IOth2_contrastAdj = imadjust(IOth2, [l1 u1],[l2 u2], gamma);
elseif ismember(i, fn2)
    IOth2_contrastAdj = imadjust(IOth2, [l1 u1],[l2 u2], gamma+1);
elseif ismember(i, fn3)
    IOth2_contrastAdj = imadjust(IOth2, [l1 u1],[l2 u2], gamma+1);
else
    IOth2_contrastAdj = imadjust(IOth2, [l1-0.15 u1],[l2 u2], gamma+0.5);
end
% Otsu's global thresholding technique
threshold1 = graythresh(IOth2_contrastAdj);
Iseg1 = imbinarize(IOth2_contrastAdj, threshold1);
% Remove components less than 31 pixels
Iseg1 = bwareaopen(Iseg1, 31);

% THIN VESSELS EXTRACTION
Icp2 = imcomplement(Igch_gaussfilt);
radius3 = 3; radius4 = 15;
Io2 = imopen(Icp2, strel('disk',radius3));
Ic2 = imclose(Io2, strel('disk', radius4));
IOththinv = Icp2 - Ic2;
% Homomorphic filtering
IOththinvd2 = double(IOththinv);
[r, c] = size(Igch);
IOthp1 = IOththinvd2 + 1;
lIOth2 = log(IOthp1);
fIOth2 = fft2(lIOth2);
```

Figure A.1 Matlab code: Retinal blood vessels segmentation (continued).

```
    if (ismember(i, fn1) || ismember(i, fn2))
        lowg2 = 0.5; highg2 = 1.5; sigma2 = 20;
    elseif ismember(i, fn3)
        lowg2 = 0.5; highg2 = 1.8; sigma2 = 20;
    else
        lowg2 = 0.5; highg2 = 2.5; sigma2 = 20;
    end
    hIOth2 = homomorph(fIOth2, lowg2, highg2, sigma2);
    IfIOth2 = ifft2(hIOth2);
    EIOth2 = real(exp(IfIOth2));
    EIOth2 = uint8(EIOth2);

    % Two-Dimensional Matched Filtering
    S = 1.2; L = 13;
    theta = 0:7:182; % different rotations considered
    Imout = zeros(size(Igch));
    M = max(ceil(3*S),(L-1)/2);
    [X, Y] = meshgrid(-M:M, -M:M);
    for t = theta
       t_rad = t/180 * pi;
       U = cos(t_rad)*X - sin(t_rad)*Y;
       V = sin(t_rad)*X + cos(t_rad)*Y;
       P = (abs(U) <= 3*S) & (abs(V) <= L/2);
       K = exp(-U.^2/(2*S.^2));
       K = K - mean(K(P));
       K(~P) = 0;
       Res_Img = conv2(EIOth2, K, 'same');
       Imout = max(Imout,Res_Img);
    end
    Imout = Imout/max(Imout(:));
    Imout_u8 = im2uint8(Imout);
    g2 = imtophat(Imout_u8, strel('disk', 7));
    % Contrast adjustments
    lo1 = 0.005; ub1 = 0.6; lo2 = 0; ub2 = 1; gamma = 2;
    if ismember(i, fn1)
        h1 = imadjust(g2, [lo1 ub1], [lo2 ub2], gamma-0.1);
    elseif ismember(i, fn2)
        h1 = imadjust(g2, [lo1 ub1], [lo2 ub2], gamma-0.4);
    elseif ismember(i, fn3)
        h1 = imadjust(g2, [lo1 ub1], [lo2 ub2], gamma-0.5);
    else
        h1 = imadjust(g2, [lo1 ub1], [lo2 ub2], gamma);
    end
    % Thresholding/binarization
    threshold2 = graythresh(h1);
    Iseg2 = imbinarize(h1, threshold2);
    % FINAL BLOOD VESSELS
    BW_BVessels = Iseg1 + Iseg2;
    BW_BVessels_NR = bwareaopen(BW_BVessels, 15);
    BW_BVessels_final = bwmorph(BW_BVessels_NR, 'bridge', 50);
    BW_BVessels_final = bwmorph(BW_BVessels_final, 'fill');
    imshow(BW_BVessels_final)
    % Write the image to local disk
    destination = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor/Base11_BVesselsOnlyRV_2/'];
    imwrite(BW_BVessels_final, strcat(destination, theImages(i).name));
    drawnow
    % Extract the area of retinal vessels
    AreaOfVessels(i,1) = nnz(BW_BVessels_final);
end
```

Figure A.1 Matlab code: Retinal blood vessels segmentation (continued).

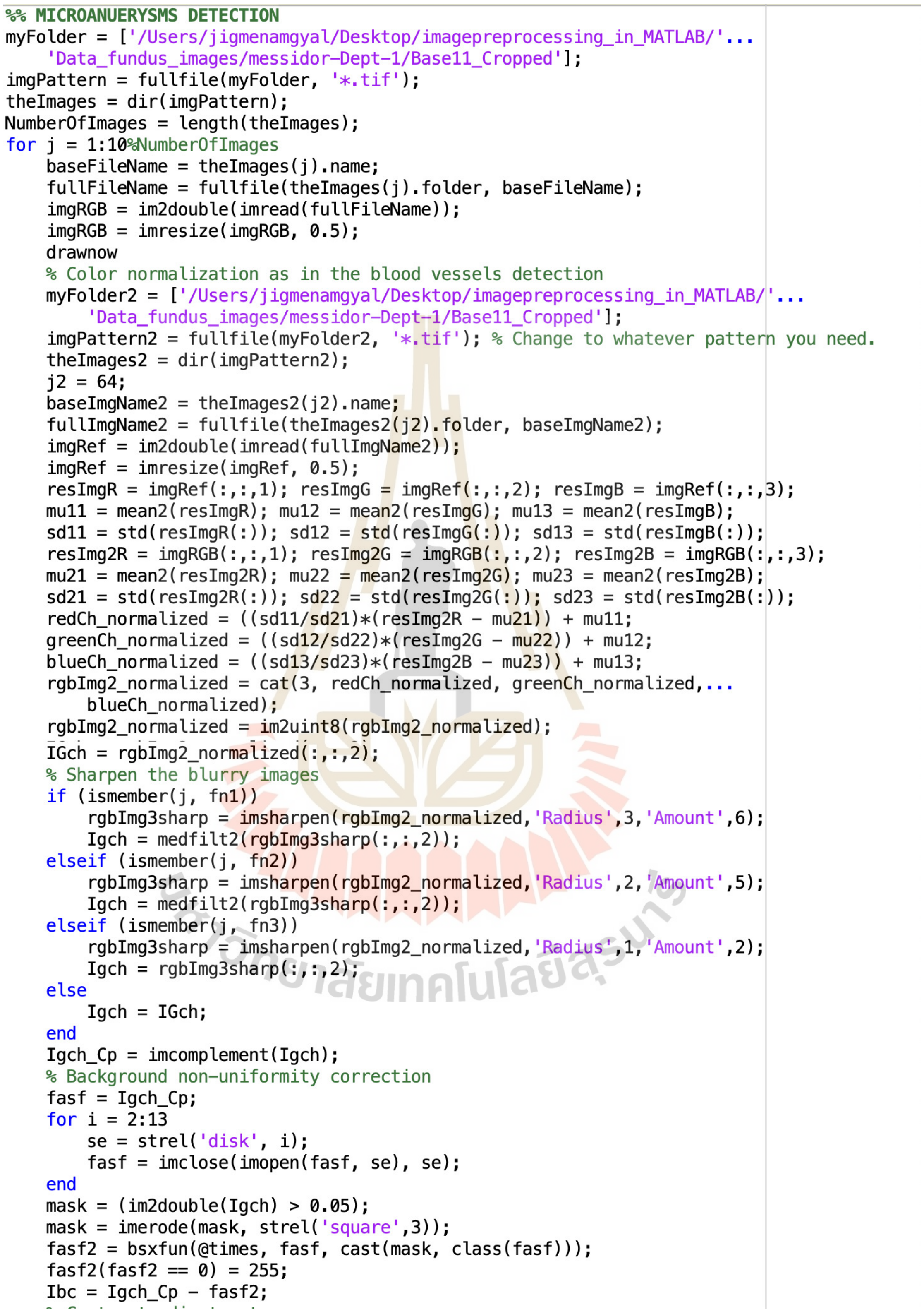


```
%% RETINAL HAEMORRHAGES DETECTION
clear; close all; clc;
myFolder = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_Cropped'];
imgPattern = fullfile(myFolder, '*.tif');
theImages = dir(imgPattern);
NumberOfImages = length(theImages);
for i = 1:NumberOfImages
    baseImageName = theImages(i).name;
    fullImageName = fullfile(theImages(i).folder, baseImageName);
    imgRGB = im2double(imread(fullImageName));
    imgRGB = imresize(imgRGB, 0.5);
    imshow(imgRGB)
    % Load the reference image for colour normalization
    myFolder2 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base11_Cropped'];
    imgPattern2 = fullfile(myFolder2, '*.tif');
    theImages2 = dir(imgPattern2);
    j = 64;
    baseImageName2 = theImages2(j).name;
    fullImageName2 = fullfile(theImages2(j).folder, baseImageName2);
    imgRef = im2double(imread(fullImageName2));
    imgRef = imresize(imgRef, 0.5);
    resImgR = imgRef(:,:,1); resImgG = imgRef(:,:,2); resImgB = imgRef(:,:,3);
    mu11 = mean2(resImgR); mu12 = mean2(resImgG); mu13 = mean2(resImgB);
    sd11 = std(resImgR(:)); sd12 = std(resImgG(:)); sd13 = std(resImgB(:));
    resImg2R = imgRGB(:,:,1); resImg2G = imgRGB(:,:,2); resImg2B = imgRGB(:,:,3);
    mu21 = mean2(resImg2R); mu22 = mean2(resImg2G); mu23 = mean2(resImg2B);
    sd21 = std(resImg2R(:)); sd22 = std(resImg2G(:)); sd23 = std(resImg2B(:));
    redCh_normalized = ((sd11/sd21)*(resImg2R - mu21)) + mu11;
    greenCh_normalized = ((sd12/sd22)*(resImg2G - mu22)) + mu12;
    blueCh_normalized = ((sd13/sd23)*(resImg2B - mu23)) + mu13;
    rgbImg2_normalized = cat(3, redCh_normalized, greenCh_normalized,...
        blueCh_normalized);
    rgbImg2_normalized = im2uint8(rgbImg2_normalized);
    Igch1 = rgbImg2_normalized(:,:,2);
    % Read in the blood vessels and check
    myFolder3 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base11_BVesselsOnlyRV'];
    imagePattern3 = fullfile(myFolder3, '*.tif');
    theImages3 = dir(imagePattern3);
    l = i;
    baseImageName3 = theImages3(l).name;
    fullImageName3 = fullfile(theImages3(l).folder, baseImageName3);
    BVessels = imread(fullImageName3);
    % for conditional statement 1
    if ismember(i, fn1)
        Igch = imsharpen(Igch1, 'Radius', 3, 'Amount', 8);
    elseif ismember(i, fn2)
        Igch = imsharpen(Igch1, 'Radius', 2, 'Amount', 3);
    elseif ismember(1, fn3)
        Igch = imsharpen(Igch1, 'Radius', 1, 'Amount', 1.5);
    else
        Igch = Igch1;
    end
    Im = imcomplement(Igch);
    % Morphological processessing
    fasf = Im;
    for ii = 1:13
        se = strel('disk', ii);
        fasf = imclose(imopen(fasf, se), se);
    end
    mask = imfill((im2double(Igch) > 0.05),'holes');
    mask2 = imerode(mask, strel('square',3));
    fasf2 = im2uint8(bsxfun(@times, fasf, cast(mask2, class(fasf))));
    fasf2(fasf2 == 0) = 255;
    Ibc = Im - fasf2;
    % Median filtering
    Ibcmf = medfilt2(Ibc);
```

Figure A.2 Matlab code: Haemorrhages detection.

```
    % Contrast adjustment
    lb1 = 0.005; ub1 = 0.5; lb2 = 0; ub2 = 1; gamma = 1.5;
    Ibc_inadj = imadjust(Ibcmf, [lb1 ub1], [lb2 ub2], gamma);
    % Homomorphic filtering
    Ienf = double(Ibc_inadj);
    Ien1 = Ienf + 1;
    lIen = log(Ien1);
    fIen = fft2(lIen);
    if ismember(i, fn1)
        lowg = 0.7; highg = 1.3; sigma = 10;
    elseif ismember(i, fn2)
        lowg = 0.7; highg = 1.7; sigma = 10;
    elseif ismember(i, fn3)
        lowg = 0.7; highg = 2; sigma = 10;
    else
        lowg = 0.7; highg = 2.5; sigma = 10;
    end
    hIen = homomorph(fIen, lowg, highg, sigma);
    IfIen = ifft2(hIen);
    EIen = real(exp(IfIen));
    EIen = uint8(EIen);
    % Thresholding
    threshold = graythresh(EIen);
    bw = imbinarize(EIen, threshold);
    % Detect only the blood vessels through linear structuring elements
    deg = -75:15:90;
    Imout = uint8(zeros(size(Igch)));
    for k = deg
        Io2 = imopen(im2uint8(BVessels), strel('line', 25, k));
        Imout = max(Imout, Io2);
    end
    recon1 = imreconstruct(logical(Imout), BVessels);
    recon2 = imreconstruct(recon1, bw);
    g1 = bw - recon2;
    if ismember(i, fn1)
        g1filtered = bwareaopen(g1, 17);
    else
        g1filtered = bwareaopen(g1, 20);
    end
    % Intensity analysis to remove false positives from the background
    [HEMs_image, numComponents] = bwlabel(g1filtered);
    props = regionprops(HEMs_image ,Igch, 'MeanIntensity', 'PixelIdxList');
    for jj = 1 : numComponents
    PixelIndices = props(jj).PixelIdxList;
    MeanIntensity = props(jj).MeanIntensity;
    if MeanIntensity > 95
        HEMs_image(PixelIndices) = 0;
    else
        HEMs_image(PixelIndices) = 1;
    end
    end
    HEMs_image = logical(HEMs_image);
    % Connected component analysis
    cc = bwconncomp(HEMs_image);
    stats = regionprops(cc, 'Eccentricity');
    idx = find([stats.Eccentricity] >= 0 & [stats.Eccentricity] <= 0.979);
    bw_final = ismember(labelmatrix(cc), idx);
    g2 = bsxfun(@times, bw_final, cast(~BVessels, class(bw_final)));
    HEMs_imgfinal = bwareaopen(g2, 6);
    imshow(HEMs_imgfinal)
    % Extract the area of haemorrhages
    AreaOfHEMs(i,1) = nnz(HEMs_imgfinal);
end
```

Figure A.2 Matlab code: Haemorrhages detection (continued).

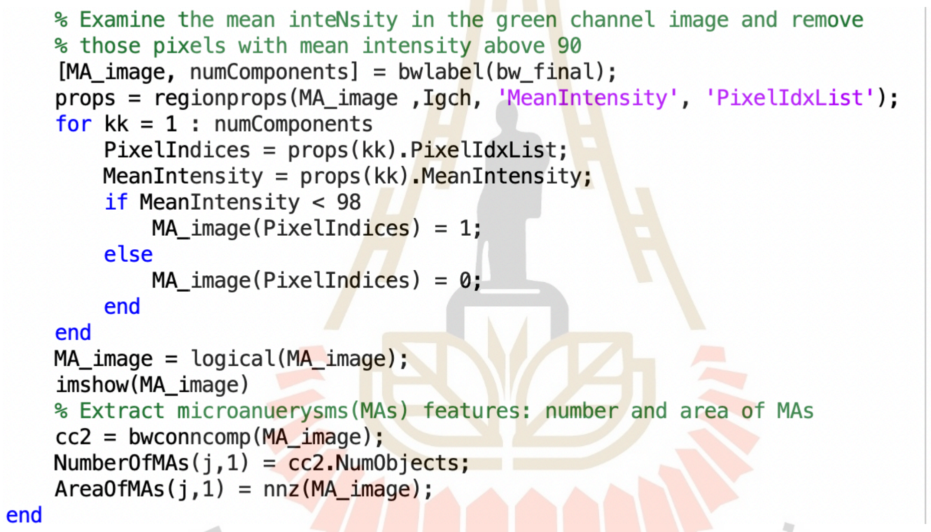


```
%% MICROANUERYSMS DETECTION
myFolder = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_Cropped'];
imgPattern = fullfile(myFolder, '*.tif');
theImages = dir(imgPattern);
NumberOfImages = length(theImages);
for j = 1:10%NumberOfImages
    baseFileName = theImages(j).name;
    fullFileName = fullfile(theImages(j).folder, baseFileName);
    imgRGB = im2double(imread(fullFileName));
    imgRGB = imresize(imgRGB, 0.5);
    drawnow
    % Color normalization as in the blood vessels detection
    myFolder2 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base11_Cropped'];
    imgPattern2 = fullfile(myFolder2, '*.tif'); % Change to whatever pattern you need.
    theImages2 = dir(imgPattern2);
    j2 = 64;
    baseImgName2 = theImages2(j2).name;
    fullImgName2 = fullfile(theImages2(j2).folder, baseImgName2);
    imgRef = im2double(imread(fullImgName2));
    imgRef = imresize(imgRef, 0.5);
    resImgR = imgRef(:,:,1); resImgG = imgRef(:,:,2); resImgB = imgRef(:,:,3);
    mu11 = mean2(resImgR); mu12 = mean2(resImgG); mu13 = mean2(resImgB);
    sd11 = std(resImgR(:)); sd12 = std(resImgG(:)); sd13 = std(resImgB(:));
    resImg2R = imgRGB(:,:,1); resImg2G = imgRGB(:,:,2); resImg2B = imgRGB(:,:,3);
    mu21 = mean2(resImg2R); mu22 = mean2(resImg2G); mu23 = mean2(resImg2B);
    sd21 = std(resImg2R(:)); sd22 = std(resImg2G(:)); sd23 = std(resImg2B(:));
    redCh_normalized = ((sd11/sd21)*(resImg2R - mu21)) + mu11;
    greenCh_normalized = ((sd12/sd22)*(resImg2G - mu22)) + mu12;
    blueCh_normalized = ((sd13/sd23)*(resImg2B - mu23)) + mu13;
    rgbImg2_normalized = cat(3, redCh_normalized, greenCh_normalized,...
        blueCh_normalized);
    rgbImg2_normalized = im2uint8(rgbImg2_normalized);
    IGch = rgbImg2_normalized(:,:,2);
    % Sharpen the blurry images
    if (ismember(j, fn1))
        rgbImg3sharp = imsharpen(rgbImg2_normalized,'Radius',3,'Amount',6);
        Igch = medfilt2(rgbImg3sharp(:,:,2));
    elseif (ismember(j, fn2))
        rgbImg3sharp = imsharpen(rgbImg2_normalized,'Radius',2,'Amount',5);
        Igch = medfilt2(rgbImg3sharp(:,:,2));
    elseif (ismember(j, fn3))
        rgbImg3sharp = imsharpen(rgbImg2_normalized,'Radius',1,'Amount',2);
        Igch = rgbImg3sharp(:,:,2);
    else
        Igch = IGch;
    end
    Igch_Cp = imcomplement(Igch);
    % Background non-uniformity correction
    fasf = Igch_Cp;
    for i = 2:13
        se = strel('disk', i);
        fasf = imclose(imopen(fasf, se), se);
    end
    mask = (im2double(Igch) > 0.05);
    mask = imerode(mask, strel('square',3));
    fasf2 = bsxfun(@times, fasf, cast(mask, class(fasf)));
    fasf2(fasf2 == 0) = 255;
    Ibc = Igch_Cp - fasf2;
```

Figure A.3 Matlab code: Microaneurysms detection.

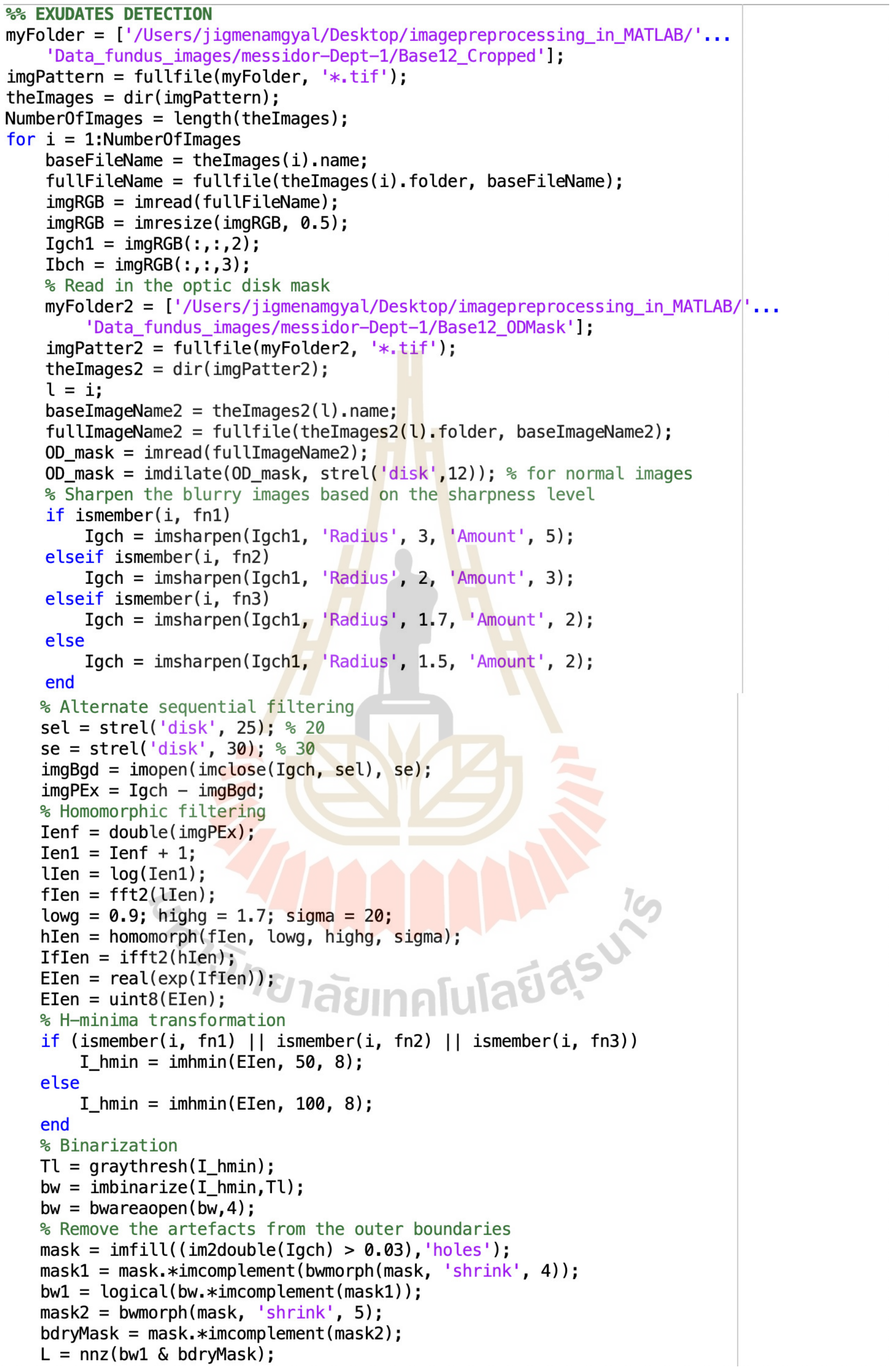
```
% Contrast adjustment
if (ismember(j, fn1))
    la = 0.01; lb = 0.2; ua = 0.5; ub = 0.7; gamma = 2.5;
    Ibcca = imadjust(Ibc, [la lb], [ua ub], gamma);
elseif (ismember(j, fn2))
    la = 0.01; lb = 0.17; ua = 0.5; ub = 0.7; gamma = 2.5;
    Ibcca = imadjust(Ibc, [la lb], [ua ub], gamma);
elseif (ismember(j, fn3))
    la = 0.01; lb = 0.17; ua = 0.5; ub = 0.7; gamma = 1.8;
    Ibcca = imadjust(Ibc, [la lb], [ua ub], gamma);
else
    la = 0.01; lb = 0.15; ua = 0.5; ub = 0.7; gamma = 1.8;
    Ibcca = imadjust(Ibc, [la lb], [ua ub], gamma);
end
% Detection of blood vessels only
deg = -75:15:90;
Imout = uint8(zeros(size(IGch)));
for k = deg
    Io = imopen(Ibc, strel('line', 15, k));
    Imout = max(Imout, Io);
end
% Morphological reconstruction
Irc = imreconstruct(Ibcca, Imout);
I_ma = Ibcca - Irc;
bin_mask = Igch > 7;
I_ma2 = bsxfun(@times, I_ma, cast(bin_mask, 'like', I_ma));
% Binarization
if (ismember(j, fn1) || ismember(j, fn2) || ismember(j, fn3))
    bw = imbinarize(I_ma, 0.53);
else
    bw = imbinarize(I_ma, 0.51);
end
% Read optic disk mask to eliminate false positives around optic disk
myFolder3 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_ODMask'];
imgPattern3 = fullfile(myFolder3, '*.tif');
theImages3 = dir(imgPattern3);
jj = j;
baseImgName3 = theImages3(jj).name;
fullImgName3 = fullfile(theImages3(jj).folder, baseImgName3);
ODmask = imread(fullImgName3);
ODmask = imdilate(ODmask, strel('disk', 7));
bw = logical(bsxfun(@times, bw, cast(~ODmask, class(bw))));
bw = bwareaopen(bw, 5);
% Read in blood vessels image
myFolder4 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_BVesselsOnlyRV'];
imgPattern4 = fullfile(myFolder4, '*.tif'); % Change to whatever pattern you need.
theImages4 = dir(imgPattern4);
jj2 = j;
baseImgName4 = theImages4(jj2).name;
fullImgName4 = fullfile(theImages4(jj2).folder, baseImgName4);
BVessels = imread(fullImgName4);
% Remove remnants from blood vessels
bw2 = bsxfun(@times, bw, cast(~BVessels, 'like', bw));
% Connected Component Analysis to filter further
cc = bwconncomp(bw2);
stats = regionprops(cc, 'Area', 'MajorAxisLength', 'MinorAxisLength',...
    'Circularity', 'Eccentricity');
idx = find([stats.MajorAxisLength]./[stats.MinorAxisLength] >= 1.0 &...
    [stats.MajorAxisLength]./[stats.MinorAxisLength] <= 2.5 & ...
    [stats.Area] >= 5 & [stats.Area] <= 22 & [stats.Circularity] >= 1 ...
    & [stats.Circularity] <=2.5 & [stats.Eccentricity] <= 0.9);
bw_final = ismember(labelmatrix(cc), idx);
```

Figure A.3 Matlab code: Microaneurysms detection (continued).



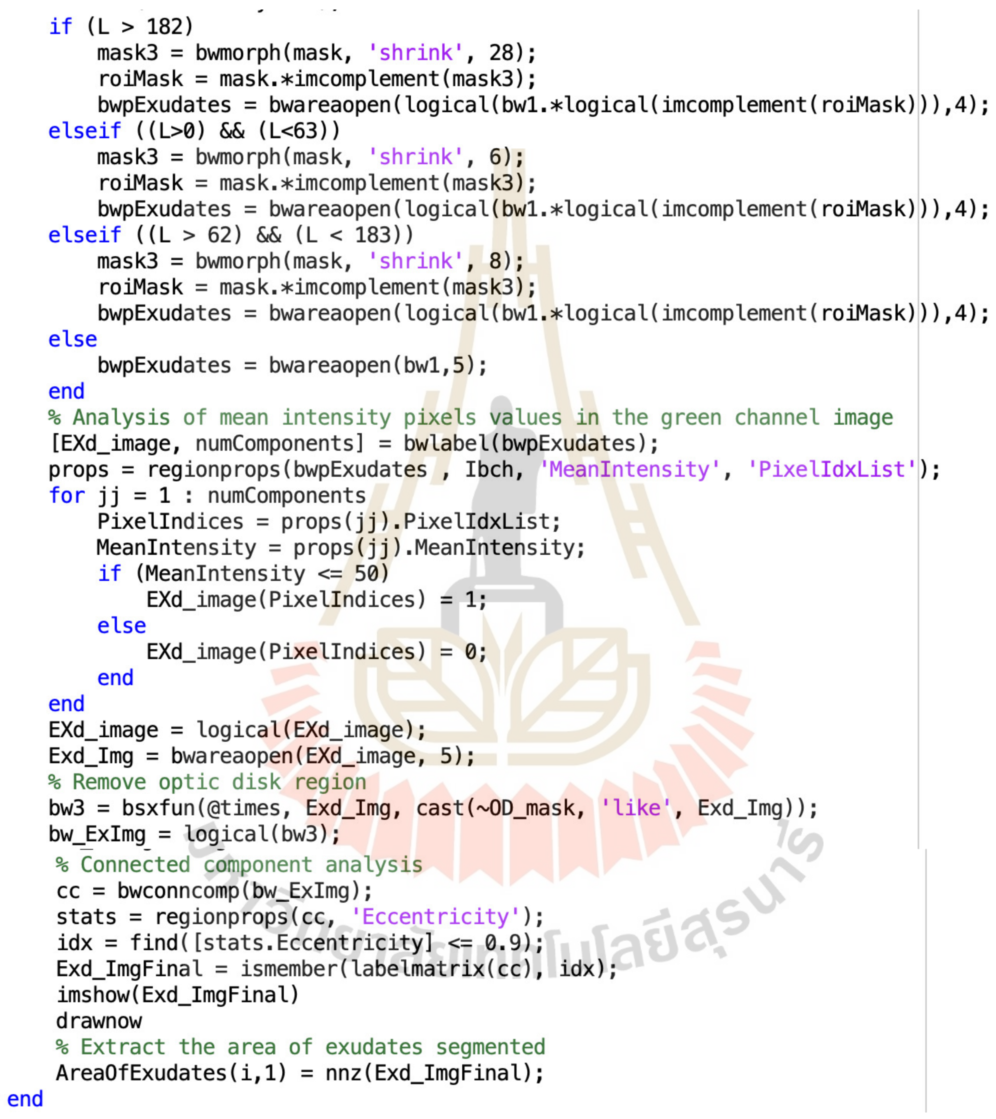
```
    % Examine the mean inteNsity in the green channel image and remove
    % those pixels with mean intensity above 90
    [MA_image, numComponents] = bwlabel(bw_final);
    props = regionprops(MA_image ,Igch, 'MeanIntensity', 'PixelIdxList');
    for kk = 1 : numComponents
        PixelIndices = props(kk).PixelIdxList;
        MeanIntensity = props(kk).MeanIntensity;
        if MeanIntensity < 98
            MA_image(PixelIndices) = 1;
        else
            MA_image(PixelIndices) = 0;
        end
    end
    MA_image = logical(MA_image);
    imshow(MA_image)
    % Extract microanuerysms(MAs) features: number and area of MAs
    cc2 = bwconncomp(MA_image);
    NumberOfMAs(j,1) = cc2.NumObjects;
    AreaOfMAs(j,1) = nnz(MA_image);
end
```

Figure A.3 Matlab code: Microaneurysms detection (continued).



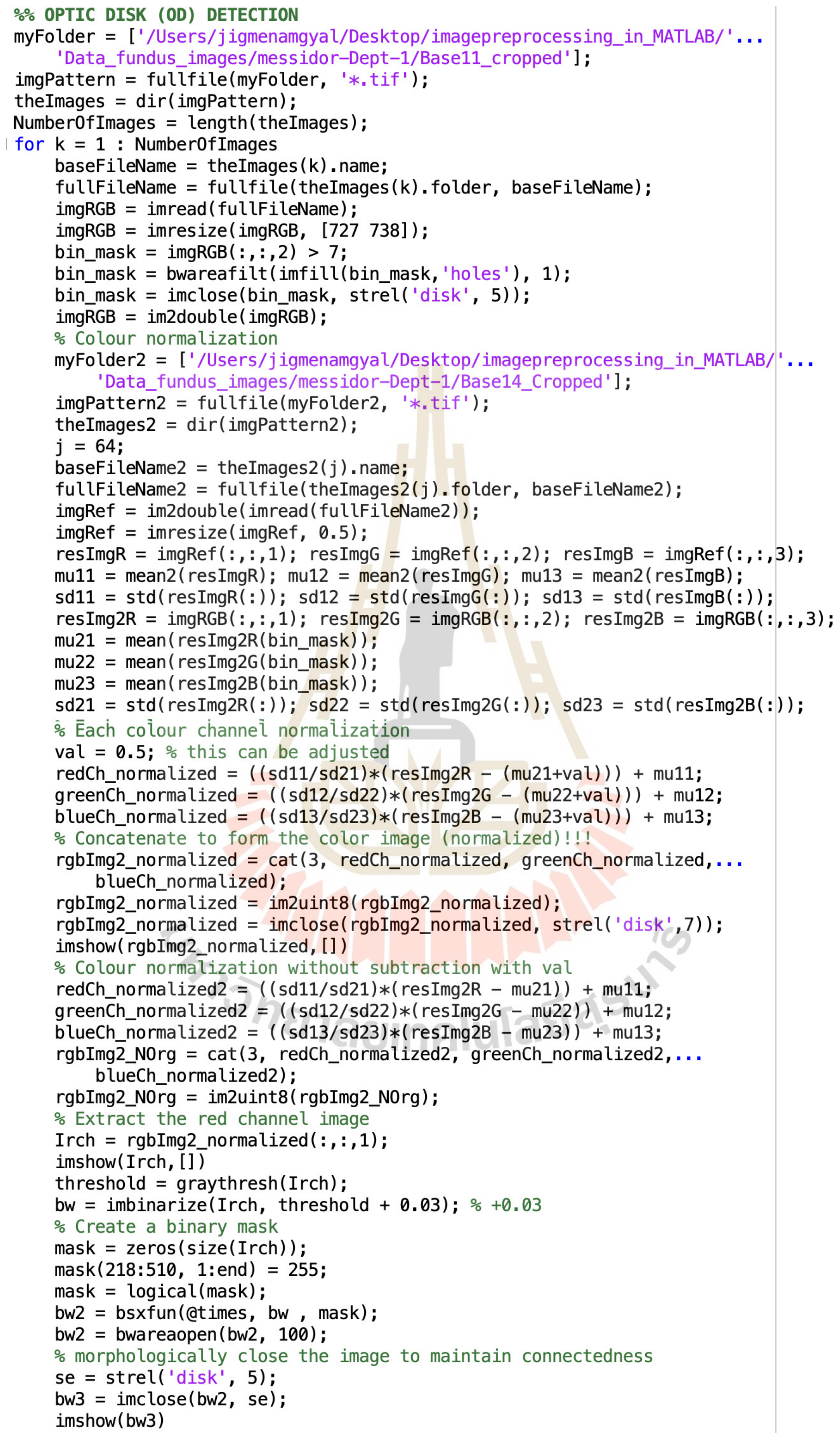
```
%% EXUDATES DETECTION
myFolder = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base12_Cropped'];
imgPattern = fullfile(myFolder, '*.tif');
theImages = dir(imgPattern);
NumberOfImages = length(theImages);
for i = 1:NumberOfImages
    baseFileName = theImages(i).name;
    fullFileName = fullfile(theImages(i).folder, baseFileName);
    imgRGB = imread(fullFileName);
    imgRGB = imresize(imgRGB, 0.5);
    Igch1 = imgRGB(:,:,2);
    Ibch = imgRGB(:,:,3);
    % Read in the optic disk mask
    myFolder2 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base12_ODMask'];
    imgPatter2 = fullfile(myFolder2, '*.tif');
    theImages2 = dir(imgPatter2);
    l = i;
    baseImageName2 = theImages2(l).name;
    fullImageName2 = fullfile(theImages2(l).folder, baseImageName2);
    OD_mask = imread(fullImageName2);
    OD_mask = imdilate(OD_mask, strel('disk',12)); % for normal images
    % Sharpen the blurry images based on the sharpness level
    if ismember(i, fn1)
        Igch = imsharpen(Igch1, 'Radius', 3, 'Amount', 5);
    elseif ismember(i, fn2)
        Igch = imsharpen(Igch1, 'Radius', 2, 'Amount', 3);
    elseif ismember(i, fn3)
        Igch = imsharpen(Igch1, 'Radius', 1.7, 'Amount', 2);
    else
        Igch = imsharpen(Igch1, 'Radius', 1.5, 'Amount', 2);
    end
    % Alternate sequential filtering
    sel = strel('disk', 25); % 20
    se = strel('disk', 30); % 30
    imgBgd = imopen(imclose(Igch, sel), se);
    imgPEx = Igch - imgBgd;
    % Homomorphic filtering
    Ienf = double(imgPEx);
    Ien1 = Ienf + 1;
    lIen = log(Ien1);
    fIen = fft2(lIen);
    lowg = 0.9; highg = 1.7; sigma = 20;
    hIen = homomorph(fIen, lowg, highg, sigma);
    IfIen = ifft2(hIen);
    EIen = real(exp(IfIen));
    EIen = uint8(EIen);
    % H-minima transformation
    if (ismember(i, fn1) || ismember(i, fn2) || ismember(i, fn3))
        I_hmin = imhmin(EIen, 50, 8);
    else
        I_hmin = imhmin(EIen, 100, 8);
    end
    % Binarization
    Tl = graythresh(I_hmin);
    bw = imbinarize(I_hmin,Tl);
    bw = bwareaopen(bw,4);
    % Remove the artefacts from the outer boundaries
    mask = imfill((im2double(Igch) > 0.03),'holes');
    mask1 = mask.*imcomplement(bwmorph(mask, 'shrink', 4));
    bw1 = logical(bw.*imcomplement(mask1));
    mask2 = bwmorph(mask, 'shrink', 5);
    bdryMask = mask.*imcomplement(mask2);
    L = nnz(bw1 & bdryMask);
```

Figure A.4 Matlab code: Exudates detection.



```
    if (L > 182)
        mask3 = bwmorph(mask, 'shrink', 28);
        roiMask = mask.*imcomplement(mask3);
        bwpExudates = bwareaopen(logical(bw1.*logical(imcomplement(roiMask))),4);
    elseif ((L>0) && (L<63))
        mask3 = bwmorph(mask, 'shrink', 6);
        roiMask = mask.*imcomplement(mask3);
        bwpExudates = bwareaopen(logical(bw1.*logical(imcomplement(roiMask))),4);
    elseif ((L > 62) && (L < 183))
        mask3 = bwmorph(mask, 'shrink', 8);
        roiMask = mask.*imcomplement(mask3);
        bwpExudates = bwareaopen(logical(bw1.*logical(imcomplement(roiMask))),4);
    else
        bwpExudates = bwareaopen(bw1,5);
    end
    % Analysis of mean intensity pixels values in the green channel image
    [EXd_image, numComponents] = bwlabel(bwpExudates);
    props = regionprops(bwpExudates , Ibch, 'MeanIntensity', 'PixelIdxList');
    for jj = 1 : numComponents
        PixelIndices = props(jj).PixelIdxList;
        MeanIntensity = props(jj).MeanIntensity;
        if (MeanIntensity <= 50)
            EXd_image(PixelIndices) = 1;
        else
            EXd_image(PixelIndices) = 0;
        end
    end
    EXd_image = logical(EXd_image);
    Exd_Img = bwareaopen(EXd_image, 5);
    % Remove optic disk region
    bw3 = bsxfun(@times, Exd_Img, cast(~OD_mask, 'like', Exd_Img));
    bw_ExImg = logical(bw3);
    % Connected component analysis
    cc = bwconncomp(bw_ExImg);
    stats = regionprops(cc, 'Eccentricity');
    idx = find([stats.Eccentricity] <= 0.9);
    Exd_ImgFinal = ismember(labelmatrix(cc), idx);
    imshow(Exd_ImgFinal)
    drawnow
    % Extract the area of exudates segmented
    AreaOfExudates(i,1) = nnz(Exd_ImgFinal);
end
```

Figure A.4 Matlab code: Exudates detection (continued).

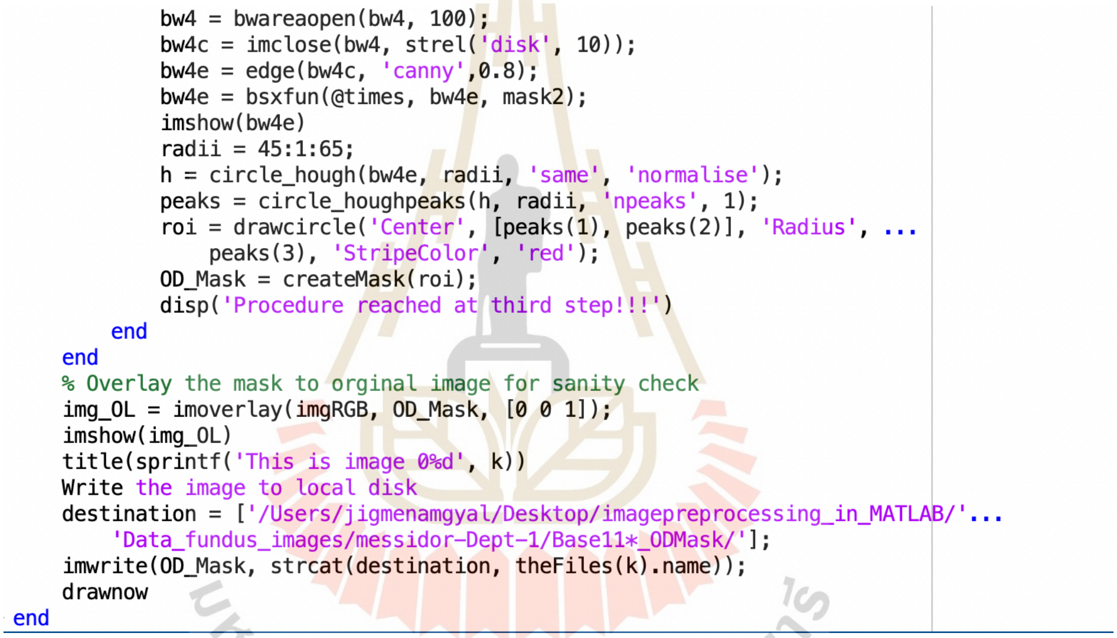


```
%% OPTIC DISK (OD) DETECTION
myFolder = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
    'Data_fundus_images/messidor-Dept-1/Base11_cropped'];
imgPattern = fullfile(myFolder, '*.tif');
theImages = dir(imgPattern);
NumberOfImages = length(theImages);
for k = 1 : NumberOfImages
    baseFileName = theImages(k).name;
    fullFileName = fullfile(theImages(k).folder, baseFileName);
    imgRGB = imread(fullFileName);
    imgRGB = imresize(imgRGB, [727 738]);
    bin_mask = imgRGB(:,:,2) > 7;
    bin_mask = bwareafilt(imfill(bin_mask,'holes'), 1);
    bin_mask = imclose(bin_mask, strel('disk', 5));
    imgRGB = im2double(imgRGB);
    % Colour normalization
    myFolder2 = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base14_Cropped'];
    imgPattern2 = fullfile(myFolder2, '*.tif');
    theImages2 = dir(imgPattern2);
    j = 64;
    baseFileName2 = theImages2(j).name;
    fullFileName2 = fullfile(theImages2(j).folder, baseFileName2);
    imgRef = im2double(imread(fullFileName2));
    imgRef = imresize(imgRef, 0.5);
    resImgR = imgRef(:,:,1); resImgG = imgRef(:,:,2); resImgB = imgRef(:,:,3);
    mu11 = mean2(resImgR); mu12 = mean2(resImgG); mu13 = mean2(resImgB);
    sd11 = std(resImgR(:)); sd12 = std(resImgG(:)); sd13 = std(resImgB(:));
    resImg2R = imgRGB(:,:,1); resImg2G = imgRGB(:,:,2); resImg2B = imgRGB(:,:,3);
    mu21 = mean(resImg2R(bin_mask));
    mu22 = mean(resImg2G(bin_mask));
    mu23 = mean(resImg2B(bin_mask));
    sd21 = std(resImg2R(:)); sd22 = std(resImg2G(:)); sd23 = std(resImg2B(:));
    % Each colour channel normalization
    val = 0.5; % this can be adjusted
    redCh_normalized = ((sd11/sd21)*(resImg2R - (mu21+val))) + mu11;
    greenCh_normalized = ((sd12/sd22)*(resImg2G - (mu22+val))) + mu12;
    blueCh_normalized = ((sd13/sd23)*(resImg2B - (mu23+val))) + mu13;
    % Concatenate to form the color image (normalized)!!!
    rgbImg2_normalized = cat(3, redCh_normalized, greenCh_normalized,...
        blueCh_normalized);
    rgbImg2_normalized = im2uint8(rgbImg2_normalized);
    rgbImg2_normalized = imclose(rgbImg2_normalized, strel('disk',7));
    imshow(rgbImg2_normalized,[])
    % Colour normalization without subtraction with val
    redCh_normalized2 = ((sd11/sd21)*(resImg2R - mu21)) + mu11;
    greenCh_normalized2 = ((sd12/sd22)*(resImg2G - mu22)) + mu12;
    blueCh_normalized2 = ((sd13/sd23)*(resImg2B - mu23)) + mu13;
    rgbImg2_NOrg = cat(3, redCh_normalized2, greenCh_normalized2,...
        blueCh_normalized2);
    rgbImg2_NOrg = im2uint8(rgbImg2_NOrg);
    % Extract the red channel image
    Irch = rgbImg2_normalized(:,:,1);
    imshow(Irch,[])
    threshold = graythresh(Irch);
    bw = imbinarize(Irch, threshold + 0.03); % +0.03
    % Create a binary mask
    mask = zeros(size(Irch));
    mask(218:510, 1:end) = 255;
    mask = logical(mask);
    bw2 = bsxfun(@times, bw , mask);
    bw2 = bwareaopen(bw2, 100);
    % morphologically close the image to maintain connectedness
    se = strel('disk', 5);
    bw3 = imclose(bw2, se);
    imshow(bw3)
```

Figure A.5 Matlab code: Optic disk segmentation.

```
% Analyze the possible optic disk locations
mask1 = mask;
mask1(:, 369:end) = false;
mask2 = mask;
mask2(:,1:369) = false;
f1 = bsxfun(@times, bw3, mask1);
f2 = bsxfun(@times, bw3, mask2);
if (nnz(f1) > nnz(f2))
    g = f1;
    mask_select = mask1;
elseif (nnz(f1) < nnz(f2))
    g = f2;
    mask_select = mask2;
else
    g = f2;
    mask_select = mask;
end
imshow(g)
% Perform connected component analysis
cc = bwconncomp(g);
stat = regionprops(cc, 'Circularity', 'Area');
idx = find([stat.Area] > 3500 & [stat.Area] < 15000 & ...
    [stat.Circularity] > 0.5);
g1 = ismember(labelmatrix(cc), idx);
g1e = edge(g1, 'canny');
imshow(g1e)
if nnz(g1) ~= 0
    % Hough transform
    imshow(g1e)
    radii = 45:1:65;
    h = circle_hough(g1e, radii, 'same', 'normalise');
    peaks = circle_houghpeaks(h, radii, 'npeaks', 1);

    roi = drawcircle('Center', [peaks(1), peaks(2)], 'Radius', ...
        peaks(3), 'StripeColor', 'red');
    OD_Mask = createMask(roi);
    disp('Procedure ended at first attempt!!!')
else
    img_hsv = rgb2hsv(rgbImg2_NOrg);
    img_hsv3 = img_hsv(:,:,3);
    img_hsv3 = imgaussfilt(img_hsv3, 2.5);
    img_hsv3 = imadjust(img_hsv3, [0 0.65],[0 1],5);
    img_hsv3 = imclose(img_hsv3, strel('disk',7));
    bin_maskE = imerode(bin_mask, strel('square', 10));
    mask2 = bsxfun(@times, mask_select, bin_maskE);
    img_edge = edge(img_hsv3, 'canny',0.3);
    img_edge = bsxfun(@times, img_edge, mask2);
    imshow(img_edge)
    if nnz(img_edge) > 300
        imshow(img_edge)
        radii = 45:1:75;
        h = circle_hough(img_edge, radii, 'same', 'normalise');
        peaks = circle_houghpeaks(h, radii, 'npeaks', 1);
        roi = drawcircle('Center', [peaks(1), peaks(2)], 'Radius', ...
            peaks(3), 'StripeColor', 'red');
        OD_Mask = createMask(roi);
        disp('Procedure reached at second step!!!')
    else
        imgB = rgbImg2_NOrg(:,:,3);
        threshold2 = adaptthresh(imgB);
        bw4 = imbinarize(imgB, threshold2);
```

Figure A.5 Matlab code: Optic disk segmentation (continued).



```
            bw4 = bwareaopen(bw4, 100);
            bw4c = imclose(bw4, strel('disk', 10));
            bw4e = edge(bw4c, 'canny',0.8);
            bw4e = bsxfun(@times, bw4e, mask2);
            imshow(bw4e)
            radii = 45:1:65;
            h = circle_hough(bw4e, radii, 'same', 'normalise');
            peaks = circle_houghpeaks(h, radii, 'npeaks', 1);
            roi = drawcircle('Center', [peaks(1), peaks(2)], 'Radius', ...
                peaks(3), 'StripeColor', 'red');
            OD_Mask = createMask(roi);
            disp('Procedure reached at third step!!!')
        end
    end
    % Overlay the mask to orginal image for sanity check
    img_OL = imoverlay(imgRGB, OD_Mask, [0 0 1]);
    imshow(img_OL)
    title(sprintf('This is image 0%d', k))
    Write the image to local disk
    destination = ['/Users/jigmenamgyal/Desktop/imagepreprocessing_in_MATLAB/'...
        'Data_fundus_images/messidor-Dept-1/Base11*_ODMask/'];
    imwrite(OD_Mask, strcat(destination, theFiles(k).name));
    drawnow
end
```

Figure A.5 Matlab code: Optic disk segmentation (continued).

CURRICULUM VITAE

NAME : Jigme Namgyal **GENDER :** Male

EDUCATION BACKGROUND:

- Bachelor of Science (Mathematics), University of New England, Armidale, New South Wales, Australia, 2014

SCHOLARSHIP:

- King's Scholarship, His Majesty's Secretariat, Office of the Gyalpoi Zimpon, Youth Welfare and Education
- SUT Scholarships for Graduate International Students (Vithedbundit)

EXPERIENCE:

- Assistant lecturer in College of Science and Technology, Royal University of Bhutan since September, 2015
- Teaching assistant in Suranaree University of Technology, Calculus I and Calculus II (International course)