

ซอฟต์แวร์เวกเตอร์แมชชีนแบบหลายกลุ่มโดยใช้กราฟไม่มีวงมีทิศทางที่ปรับได้แบบจัดเรียงใหม่



นางสาวฐิติมาพร เพชรแก้ว

สถาบันวิทยบริการ

จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์


คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2547

ISBN 974-17-5914-2

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

MULTICLASS SUPPORT VECTOR MACHINES USING  
REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPHS



Miss Thimaporn Phetkaew

สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

A Dissertation Submitted in Partial Fulfillment of the Requirements  
for the Degree of Doctor of Philosophy in Computer Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic year 2004

ISBN 974-17-5914-2

Thesis Title                      MULTICLASS SUPPORT VECTOR MACHINES USING  
REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPHS  
By                                      MISS THIMAPORN PHETKAEW  
Field of Study                      Computer Engineering  
Thesis Advisor                      Assistant Professor Dr.Boonserm Kijirikul  
Thesis Co-advisor                      Associate Professor Dr.Wanchai Rivepiboon

---

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial  
Fulfillment of the Requirements for the Doctor's Degree

..... Dean of the Faculty of Engineering  
(Professor Direk Lavansiri, Ph.D.)

THESIS COMMITTEE

..... Chairman  
(Associate Professor Prabhas Chongstitvatana, Ph.D.)

..... Thesis Advisor  
(Assistant Professor Boonserm Kijirikul, D.Eng.)

..... Thesis Co-advisor  
(Associate Professor Wanchai Rivepiboon, Ph.D.)

..... Member  
(Professor Chidchanok Lursinsap, Ph.D.)

..... Member  
(Associate Professor Thanaruk Theeramunkong, D.Eng.)

ฐิมาพร เพชรแก้ว : ซัพพอร์ตเวกเตอร์แมชชีนแบบหลายกลุ่มโดยใช้กราฟไม่มีวงมีทิศทางที่ปรับได้แบบจัดเรียงใหม่. (MULTICLASS SUPPORT VECTOR MACHINES USING REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPHS) อ. ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล, อ.ที่ปรึกษาร่วม : รองศาสตราจารย์ ดร.วันชัย รั้วไพบุลย์ จำนวนหน้า 64 หน้า. ISBN 974-17-5914-2.

ปัญหาการพัฒนาซัพพอร์ตเวกเตอร์แมชชีนให้สามารถจำแนกข้อมูลได้หลายกลุ่มยังคงอยู่ในขั้นตอนการวิจัย วิธีดีดีเอจ (Decision Directed Acyclic Graph – DDAG) ให้ความถูกต้องเทียบได้กับวิธีแมกซ์วิน (Max Wins) ที่เป็นอัลกอริทึมที่ให้ค่าความถูกต้องสูงที่สุดในปัจจุบัน แต่ใช้เวลาในการสอนและประมวลผลต่ำกว่า วิธีเอดีเอจ (Adaptive Directed Acyclic Graph – ADAG) สามารถลดปัญหาที่เกิดจากโครงสร้างของดีดีเอจได้ อย่างไรก็ตามลำดับของโนดที่แตกต่างกันในวิธีเอดีเอจอาจให้ความถูกต้องที่แตกต่างกัน

งานวิจัยนี้ได้เสนอวิธีการใหม่สำหรับการจำแนกข้อมูลแบบหลายกลุ่ม เรียกว่าอาร์เอดีเอจ (Reordering Adaptive Directed Acyclic Graph – RADAG) ซึ่งเป็นการปรับปรุงวิธีเอดีเอจเดิม และได้เสนออัลกอริทึมสำหรับการเลือกลำดับที่เหมาะสมของโนดในวิธีเอดีเอจเพื่อนำมาใช้ในการจำแนกข้อมูล โดยพิจารณาจากค่าขอบเขตของความผิดพลาดของตัวจำแนกข้อมูลทั้งหมด และได้นำอัลกอริทึมการจับคู่สมบูรณ์แบบน้ำหนักน้อยสุด (Minimum-weight perfect matching) มาประยุกต์ใช้กับอัลกอริทึมที่ทำการจัดเรียงลำดับเพื่อเลือกตัวจำแนกที่มีค่าขอบเขตของความผิดพลาดต่ำมาใช้ในการจำแนกข้อมูล และเพื่อเลือกลำดับของโนดที่เหมาะสมให้ได้ภายในเวลาพหุนาม (polynomial time)

งานวิจัยนี้ได้เปรียบเทียบประสิทธิภาพของวิธีการใหม่กับวิธีดีดีเอจ เอดีเอจ และแมกซ์วิน ผลการทดลองที่ได้แสดงให้เห็นว่าวิธีการใหม่ให้ความถูกต้องที่สูงกว่า และมีการประมวลผลเร็วกว่าวิธีแมกซ์วิน โดยเฉพาะอย่างยิ่งเมื่อมีจำนวนกลุ่ม (class) และจำนวนมิติของข้อมูล (dimension) สูง งานวิจัยนี้ได้เสนอแนวทางในการเพิ่มประสิทธิภาพของวิธีอาร์เอดีเอจ และวิธีดีดีเอจด้วย

ภาควิชา	วิศวกรรมคอมพิวเตอร์	ลายมือชื่อนิสิต.....
สาขาวิชา	วิศวกรรมคอมพิวเตอร์	ลายมือชื่ออาจารย์ที่ปรึกษา.....
ปีการศึกษา	2547	ลายมือชื่ออาจารย์ที่ปรึกษาร่วม.....

## 4371826121 : MAJOR COMPUTER ENGINEERING

KEY WORD: MULTICLASS CLASSIFICATION / SUPPORT VECTOR MACHINES / RADAG

THIMAPORN PHETKAEW: MULTICLASS SUPPORT VECTOR MACHINES USING REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPHS. THESIS ADVISOR: ASST. PROF. BOONSERM KIJSIRIKUL, D.Eng., THESIS COADVISOR: ASSOC. PROF. WANCHAI RIVEPIBOON, Ph.D., 64 pp. ISBN 974-17-5914-2.

The problem of extending binary support vector machines (SVMs) for multiclass classification is still an ongoing research issue. The Decision Directed Acyclic Graph (DDAG) method reduces training and evaluation time, while maintaining accuracy compared to the Max Wins, which is probably the currently most accurate method for multiclass SVMs. The Adaptive Directed Acyclic Graph (ADAG) approach is proposed to alleviate the problem of the DDAG structure. However, different sequences of binary classifiers in nodes in the ADAG may provide different accuracy.

In this research we present a new method, Reordering Adaptive Directed Acyclic Graph (RADAG), which is the modification of the original ADAG method. We propose an algorithm to choose an optimal sequence of binary classifiers in nodes in the ADAG by considering the generalization error bounds of all classifiers. We apply minimum-weight perfect matching with the reordering algorithm in order to select binary classifiers which have small generalization errors to be used in data classification and in order to find the best sequence of binary classifiers in polynomial time.

We then compare the performance of our method with previous methods including the DDAG, the ADAG and the Max Wins. Experiments denote that our method gives higher accuracy. Moreover it runs faster than Max Wins, especially when the number of classes and/or the number of dimensions are relatively large. In this research we also present alternative ways to enhance the performance of the RADAG and the DDAG as well.

Department	Computer Engineering	Student's signature.....
Field of study	Computer Engineering	Advisor's signature.....
Academic year	2004	Co-advisor's signature.....

## ACKNOWLEDGEMENTS

I am especially deeply grateful to my thesis advisor, Assistant Professor Dr. Boonserm Kijirikul, who provided me with a great deal of guidance in the area of machine learning. He always pushes and motivates me throughout this research period. His valuable suggestions and comments made this work feasible. Moreover, he also gave me an opportunity getting the research fund – the Royal Golden Jubilee Ph.D. Program.

I would like to thank my thesis co-advisor, Associate Professor Dr. Wanchai Rivepiboon, who accepted me to study in Doctor of Philosophy Program in Computer Engineering and allowed me to do research in my interested research field.

I would like to thank my thesis committee members, Associate Professor Dr. Prabhas Chongstitvatana, Professor Dr. Chidchanok Lursinsup and Assistant Professor Dr. Thanaruk Theeramunkong, who provided valuable comments at the committee meetings. I am also specially thankful to Associate Professor Dr. Prabhas Chongstitvatana for giving encouragement and help thru all my Ph.D. student life.

I would like to thank Dr. Athasit Surarerks for correcting time complexity of my algorithm and Professor Dr. Don Mcneil for his valuable suggestion which was about the methodology of evaluation of my algorithm by using statistics.

I would like to thank the Thailand Research Fund (the Royal Golden Jubilee Ph.D. Program) for funding source. This fund gave me a chance to have research collaboration in oversea. It made me open my research vision widely. I also would like to thank Commission on Higher Education Ministry of Education for funding source.

I would like to thank all members of the Machine Intelligence and Knowledge Discovery Laboratory and members of the Software Engineering Laboratory for their helps and discussions on my thesis. My special thanks are given to Miss Sunee Pongpinijpinyo for my English consultation.

My special thanks go to my beloved parents for their everlasting supports and encouragements throughout my study. The success of my study has been paved by their love and vision. I especially thank my dear husband, Mr. Nuchanon Benchaphun, for his patience. My success would not be possible without his encouragement.

# CONTENTS

	Page
ABSTRACT (THAI).....	iv
ABSTRACT (ENGLISH).....	v
ACKNOWLEDGEMENTS.....	vi
CONTENTS.....	vii
LIST OF TABLES.....	ix
LIST OF FIGURES.....	xi
1 INTRODUCTION.....	1
1.1 Motivation.....	1
1.2 Objective.....	2
1.3 Scope and Limitation.....	2
1.4 Contribution.....	2
1.5 Research Methodology.....	3
1.6 Organization of the Thesis.....	3
2 BACKGROUND AND LITERATURE REVIEW.....	4
2.1 Background.....	4
2.1.1 The Problem of Risk Minimization for Pattern Recognition.....	4
2.1.1.1 Empirical Risk Minimization Induction Principle.....	5
2.1.1.2 Equivalent Definition of the VC Dimension.....	6
2.1.1.3 Structural Risk Minimization Induction Principle.....	7
2.1.2 Support Vector Machines.....	9
2.1.2.1 Linear Support Vector Machines.....	9
2.1.2.2 Non-linear Support Vector Machines.....	12
2.2 Literature Review.....	13
2.2.1 One-against-the-rest.....	13
2.2.2 One-against-one.....	14
2.2.3 Decision Directed Acyclic Graph (DDAG).....	14
2.2.4 Adaptive Directed Acyclic Graph (ADAG).....	15
2.2.5 Binary Tree SVMs.....	17
2.2.6 Decision-Tree-Based Multiclass SVMs.....	18

	Page
3 REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPH.....	20
3.1 Reordering Adaptive Directed Acyclic Graph (RADAG).....	20
3.2 Reordering Algorithm.....	21
3.2.1 Generalization Performance of Support Vector Machines .....	21
3.2.2 Reordering Algorithm with Minimum-weight Perfect Matching.....	22
3.3 Time Complexity.....	24
3.4 Estimating the Difference between Means of Generalization Errors.....	25
4 EXPERIMENTAL SETTING and RESULTS.....	27
4.1 Dataset and Experimental Setting.....	27
4.2 Experimental Results.....	28
4.2.1 The Accuracy of Classification.....	28
4.2.2 The Computational Time.....	37
4.3 Evaluate the RADAG.....	40
4.3.1 Estimating the Difference between Accuracies.....	40
5 PERFORMANCE ENHANCEMENT.....	45
5.1 Enhancing Performance of the DDAG.....	45
5.2 Enhancing Performance of the RADAG.....	46
5.2.1 The Problem.....	46
5.2.2 Enhancement Versions.....	48
5.2.3 Results.....	51
6 SUMMARY AND FUTURE WORKS.....	55
6.1 Summary.....	55
6.2 Future Works.....	56
REFERENCES.....	57
APPENDICES.....	62
APPENDIX A. Publications.....	63
BIOGRAPHY.....	64



## LIST OF TABLES

Table	Page
4.1 Description of the datasets used in the experiments.....	28
4.2 A comparison of the accuracy of classification of the glass dataset.....	29
4.3 A comparison of the accuracy of classification of the satimage dataset.....	29
4.4 A comparison of the accuracy of classification of the segment dataset.....	30
4.5 A comparison of the accuracy of classification of the shuttle dataset.....	30
4.6 A comparison of the accuracy of classification of the vowel dataset.....	31
4.7 A comparison of the accuracy of classification of the soybean dataset.....	31
4.8 A comparison of the accuracy of classification of the letter dataset.....	32
4.9 A comparison of the accuracy of classification of the isolet dataset.....	33
4.10 A comparison of the accuracy of classification of the Thai printed character 1 dataset.....	33
4.11 A comparison of the accuracy of classification of the Thai printed character 2 dataset.....	34
4.12 A comparison of the accuracy of classification using the Polynomial kernel.....	35
4.13 A comparison of the accuracy of classification using the RBF kernel.....	35
4.14 A comparison between the RADAG and the minimum and maximum accuracies of the DDAG and the ADAG using the Polynomial kernel.....	36
4.15 A comparison between the RADAG and the minimum and maximum accuracies of the DDAG and the ADAG using the RBF kernel.....	36
4.16 A comparison of the computational time using the Polynomial kernel.....	38
4.17 A comparison of the computational time using the RBF kernel.....	39
4.18 A comparison of estimating the difference between accuracies of classification using the Polynomial kernel.....	41
4.19 A comparison of estimating the difference between accuracies of classification using the RBF kernel.....	41
5.1 A comparison of the accuracy of classification of the DDAG and the SDDAG.....	46
5.2 A comparison of the number of misclassified examples of the ADAG and the RADAG.....	47

Table	Page
5.3 The number of new misclassified examples introduced by the RADAG.....	48
5.4 A comparison of the accuracy of classification of the RADAG and the enhancing methods using the Polynomial kernel.....	53
5.5 A comparison of the accuracy of classification of the RADAG and the enhancing methods using the RBF kernel.....	54



สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

## LIST OF FIGURES

Figure	Page
2.1 The bound on the risk.....	8
2.2 An example of SVM classification.....	10
2.3 Feature space.....	12
2.4 The DDAG finding the best class out of four classes.....	15
2.5 The structure of an ADAG for an 8-class problem.....	16
2.6 The structure of binary tree classifier.....	17
2.7 The existence of unclassifiable regions.....	18
2.8 (a) An example of the division of feature space (b) Expression by a decision tree.....	19
3.1 Classifying process of the RADAG.....	21
3.2 The generalization errors of 325 classifiers.....	22
3.3 (a) A graph for an 8-class problem (b) An example of the reordering algorithm.....	24
3.4 The distribution of the generalization errors.....	26
4.1 The trends of the accuracy of classification when the number of classes is varying.....	42
4.2 The variation of the generalization errors.....	43
5.1 The SDDAG algorithm.....	45
5.2 Classifying process of the RADAG & Method1.....	48
5.3 Classifying process of the RADAG & Method2.....	49
5.4 Classifying process of the RADAG & Method3.....	49
5.5 Classifying process of the RADAG & Method4.....	50
5.6 Classifying process of the RADAG & Method5.....	51

# CHAPTER I

## INTRODUCTION

Support Vector Machines (SVMs) are learning systems that use a hypothesis space of linear functions in a high dimensional feature space. SVMs are trained by a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory (Cristianini and Shawe-Taylor, 2000). This learning strategy introduced by Vapnik (1998, 1999) is a principled and very powerful method. It was originally designed for two-class classification problems with its outstanding performance in real world applications. However, extending SVMs for multiclass classification is still an ongoing research issue.

In this chapter, we discuss motivation, objectives, scope and limitation, and contributions of the thesis.

### 1.1 Motivation

Support Vector Machines (SVMs) were primarily designed for two-class classification problems. However, most of real world problems are multiclass classification problems such as text categorization, computer vision, medical diagnosis, image recognition and speech recognition, etc. Previous methods for solving the multiclass problem of SVMs are typically to consider the problem as the combination of two-class decision functions, e.g. one-against-one and one-against-the-rest (Hsu and Lin, 2002).

Friedman (1996) suggested the Max Wins algorithm in which each one-against-one classifier casts one vote for its preferred class, and the final result is the class with the most votes. The Max Wins algorithm offers faster training time compared to the one-against-the-rest method. The Decision Directed Acyclic Graph (DDAG) method proposed by Platt, Cristianini and Shawe-Taylor (2000) reduces training and evaluation time, while maintaining accuracy compared to the Max Wins. The comparison experiments in several methods on large problems in (Hsu and Lin, 2002) show that the Max Wins algorithm and the DDAG may be more suitable for practical use. Ussivakul and Kijirikul (2001, 2002) proposed the Adaptive Directed Acyclic Graph (ADAG)

method which is the modification of the DDAG. This method reduces the dependency of the sequence of nodes in the structure as well as lowers the number of tests required to evaluate for the correct class. Their approach yields higher accuracy and reliability of classification, especially in such a case that the number of classes is relatively large. There are also other implementations for multiclass SVMs, e.g., Allwein, Schapire and Singer (2000); Kindermann, Leopold and Paass (2000); Takahashi and Abe (2002).

In this thesis we reveal that the ADAG still has the dependency on the sequence of its nodes, although it is less dependent on the order of binary classes in the sequence than the DDAG; there are still differences in accuracy between different sequences. This led to the reliability of the algorithm. Here we propose a novel method that improves reliability by choosing an optimal sequence, which has less chance to predict the wrong class, and dynamically reordering the sequence during classification process according to each test data. We also reveal that the problem of selecting the appropriate sequence can be solved by minimum-weight perfect matching.

## 1.2 Objective

The objective is to develop a multiclass classification technique for SVMs, which enhances the performance in terms of accuracy and running time.

## 1.3 Scope and Limitation

The experimental results of the thesis will be compared with the results of three multiclass classification algorithms for SVMs, i.e., the DDAG, the ADAG and Max Wins. All experiments are based on datasets of the UCI Machine Learning Repository (Blake, Keogh and Merz, 1998) and the Thai printed character recognition dataset (Pichitdej, 2001).

## 1.4 Contribution

The goal of this thesis is to develop a new algorithm for multiclass SVMs, which chooses optimal classifiers with small generalization error to be used in data classification.

## 1.5 Research Methodology

- 1) Review and study the research papers that are related to the multiclass SVMs.
- 2) Design a new multiclass classification algorithm.
- 3) Prepare datasets.
- 4) Set up experiments and test.
- 5) Analyze the result and make conclusions.

## 1.6 Organization of the Thesis

The remainder of the thesis is organized into six chapters as follows. In Chapter II, we review the theoretical background including the problem of risk minimization for pattern recognition and the main idea of SVMs. We also review the literature in multiclass SVMs.

In Chapter III, we explain the problem of the structure of the original ADAG. Then we give details about our proposed method, Reordering Adaptive Directed Acyclic Graph (RADAG), which is the modification of the ADAG.

Chapter IV presents the experimental setting and results. We compare the accuracy of classification of the RADAG with those of the DDAG, the original ADAG and the Max Wins algorithm. We also compare the running time of our approach with that of Max Wins, which is probably the currently most accurate method for multiclass SVMs.

In Chapter V, we discuss how to improve the performance of the proposed method — RADAG. We also suggest an alternative way to increase the accuracy of the DDAG method.

Finally, in Chapter VI, we conclude our research work and present some directions for the future work.

## CHAPTER II

### BACKGROUND AND LITERATURE REVIEW

In this chapter, we describe the theoretical background and review the literature on multiclass SVMs. Two previous works on multiclass SVMs which are related to our proposed method, i.e., the DDAG and the ADAG will be discussed.

#### 2.1 Background

In this section we review the theoretical background that are used in this thesis including the problem of risk minimization for pattern recognition and the main idea of SVMs (Vapnik, 1998, 1999).

##### 2.1.1 The Problem of Risk Minimization for Pattern Recognition

The problem of learning is that of choosing from the given set of functions  $f(x, \alpha)$ ,  $\alpha \in \Lambda$ , the one that best approximates the supervisor's response. The selection of the desired function is based on a training set of  $l$  random independent identically distributed observations drawn according to  $P(x, y) = P(x)P(y|x)$

$$(x_1, y_1), \dots, (x_l, y_l) \quad (1)$$

In order to choose the best available approximation to the supervisor's response, one measures the *loss* or discrepancy  $L(y, f(x, \alpha))$  between the response  $y$  of the supervisor to a given input  $x$  and the response  $f(x, \alpha)$  provided by the learning machine. Consider the expected value of the loss, given by the risk functional

$$R(\alpha) = \int L(y, f(x, \alpha)) dP(x, y). \quad (2)$$

The goal is to find the function  $f(x, \alpha_0)$  which minimizes the risk functional  $R(\alpha)$  (over the class of functions  $f(x, \alpha)$ ,  $\alpha \in \Lambda$ ) in the situation where the joint probability distribution  $P(x, y)$  is unknown and the only available information is contained in the training set (1).

Below we consider the problem of pattern recognition. Let the supervisor's output  $y$  take on only two values  $y = \{0,1\}$  and let  $f(x,\alpha)$ ,  $\alpha \in \Lambda$  be a set of *indicator* functions. Consider the following loss-function:

$$L(y, f(x, \alpha)) = \begin{cases} 0 & \text{if } y = f(x, \alpha) \\ 1 & \text{if } y \neq f(x, \alpha). \end{cases} \quad (3)$$

For this loss function, the functional (2) provides the probability of classification error, i.e., when the answers  $y$  given by supervisor and the answers given by indicator function  $f(x,\alpha)$  differ. The problem, therefore, is to find the function which minimizes the probability of classification errors when probability measure  $P(x,y)$  is unknown, but the data (1) are given.

### 2.1.1.1 Empirical Risk Minimization Induction Principle

In order to minimize the risk functional (2), for an unknown probability measure  $P(x,y)$  the following induction principle is usually used. The expected risk functional is replaced by the *empirical risk* functional

$$R_{emp}(\alpha) = \frac{1}{l} \sum_{i=1}^l (L(y, f(x, \alpha))) \quad (4)$$

constructed on the basis of the training set (1). The principle is to approximate the function  $L(y, f(x, \alpha_0))$  which minimizes risk (2) by the function  $L(y, f(x, \alpha_i))$  which minimizes empirical risk (4). This principle is called the empirical risk minimization induction principle (ERM principle) (Vapnik, 1998, 1999).

Backpropagation method calculates the gradient of the empirical risk for the sigmoid approximation of Neural Networks. The three main problems encountered when minimizing the empirical risk using the backpropagation method are as follows.

- 1) The empirical risk functional has many local minima. Optimization procedures guarantee convergence to some local minimum. The quality of the obtained approximation depends on many factors, in particular on the initial parameter values of the algorithm.
- 2) Convergence to a local minimum can be rather slow (due to the high dimensionality of the weight-space).



- 3) The sigmoid function has a scaling factor which affects the quality of the approximation. To choose the scaling factor one has to make a tradeoff between quality of approximation and the rate of convergence.

Therefore, a good minimization of the empirical risk depends in many respects on the art of the researcher.

### 2.1.1.2 Equivalent Definition of the VC Dimension

The VC dimension of a set of indicator functions is the maximum number  $h$  of vectors  $(x_1, y_1), \dots, (x_h, y_h)$  which can be separated into two classes in all  $2^h$  possible ways using functions of this set (shattered by this set of functions). Any indicator function separates a given set of vectors into two subsets: the subset of vectors for which this indicator function takes the value 0 and the subset of vectors for which this indicator function takes the value 1. If for any  $n$  there exists a set of  $n$  vectors which can be shattered by the set of indicator functions, then the VC-dimension is equal to infinity.

We call a hyperplane  $(\mathbf{w} \cdot \mathbf{x}) + b = 0$ ;  $|\mathbf{w}| = 1$  the  $\Delta$ -margin separating hyperplane if it classifies vectors  $x$  as follows:

$$y = \begin{cases} 1, & \text{if } (\mathbf{w} \cdot \mathbf{x}) + b \geq \Delta \\ -1, & \text{if } (\mathbf{w} \cdot \mathbf{x}) + b \leq -\Delta \end{cases}$$

(classifications of vectors  $x$  that fall into the margin  $(-\Delta, \Delta)$  are undefined).

Let vectors  $\mathbf{x} \in X$  belong to a sphere of radius  $R$ . Then the set of  $\Delta$ -margin separating hyperplanes has the VC dimension  $h$  bounded by the inequality

$$h \leq \min \left( \left\lceil \frac{R^2}{\Delta^2} \right\rceil, n \right) + 1.$$

This shows that in general the VC dimension of the set of hyperplanes is equal to  $n+1$ , where  $n$  is dimensionality of input space. However, the VC dimension of the set of  $\Delta$ -margin separating hyperplanes (with a large value of margin  $\Delta$ ) can be less than  $n+1$ . This fact will play an important role for constructing new function estimation methods.

Consider a set of functions which possess a finite VC-dimension  $h$  where the set of loss functions (3) is a set of totally bounded functions. Without restriction in generality, we assume that

$$0 \leq L(y, f(x, \alpha)) \leq B, \quad \alpha \in \Lambda. \quad (5)$$

With probability at least  $1-\eta$ , the inequality

$$R(\alpha) \leq R_{emp}(\alpha) + \frac{B\varepsilon}{2} \left( 1 + \sqrt{1 + \frac{4R_{emp}(\alpha)}{B\varepsilon}} \right) \quad (6)$$

holds true simultaneously for all functions of the set (5), where

$$\varepsilon = 4 \frac{h \left( \ln \frac{2l}{h} + 1 \right) - \ln \eta}{l}.$$

For the set of indicator functions,  $B=1$  [see (Vapnik, 1998, 1999) for more details].

### 2.1.1.3 Structural Risk Minimization Induction Principle

The ERM principle is intended for dealing with a large sample size. Indeed, the ERM principle can be justified by considering the inequalities (6). When  $l/h$  is large, the second summand on the right hand side of inequality (6) becomes small. The actual risk is then close to the value of the empirical risk. In this case, a small value of the empirical risk provides a small value of (expected) risk.

However, if  $l/h$  is small, then even a small  $R_{emp}(\alpha_l)$  does not guarantee a small value of risk. The sample size  $l$  is considered to be small if  $l/h$  is small, say  $l/h < 20$ . In this case the minimization for  $R(\alpha)$  requires a new principle, based on the simultaneous minimization of two terms in (6) one of which depends on the value of the empirical risk while the second depends on the VC-dimension of the set of functions. To minimize risk in this case it is necessary to find a method which, along with minimizing the value of empirical risk, controls the VC-dimension of the learning machine.

The following principle, which is called the principle of structural risk minimization (SRM), is intended to minimize the risk functional with respect to both empirical risk and VC-dimension of the set of functions.

Let  $S$  the set of indicator functions be provided with a *structure*: so that  $S$  is composed of the nested subsets of functions  $S_k = \{L(y, f(x, \alpha)), \alpha \in \Lambda_k\}$ , such that

$$S_1 \subset S_2 \subset \dots \subset S_n \dots \quad (7)$$

And  $S^* = \cup_k S_k$ .

An admissible structure is one satisfying the following three properties.

- 1) The set  $S^*$  is everywhere dense in  $S$ .
- 2) The VC-dimension  $h_k$  of each set  $S_k$  of functions is finite.
- 3) Any element  $S_k$  of the structure contains totally bounded functions

$$0 \leq L(y, f(x, \alpha)) \leq B_k, \quad \alpha \in \Lambda_k.$$

The SRM principle suggests that for a given set of observations  $(x_1, y_1), \dots, (x_l, y_l)$  choose the element of structure  $S_n$ , where  $n = n(l)$  and choose the particular function  $S_n$  for which the guaranteed risk (6) is minimal.

The SRM principle actually suggests a tradeoff between the quality of the approximation and the complexity of the approximating function. As  $n$  increases, the minima of empirical risk are decreased; however, the term responsible for the confidence interval [summand in (6)] is increased.

Support Vector Machines perform the SRM. The SRM principle takes both factors into account (see Figure 2.1). The bound on the risk is the sum of the empirical risk and the confidence interval. The smallest bound of the risk is achieved on some appropriate element of the structure. For any distribution function the SRM method provides convergence to the best possible solution with probability one. In other words SRM method is universally strongly consistent.

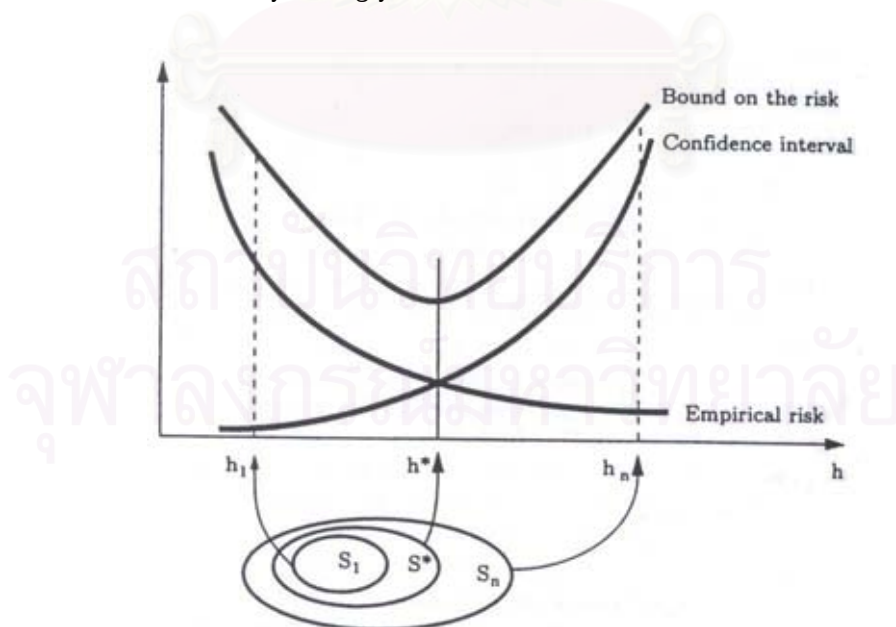


Figure 2.1: The bound on the risk.

## 2.1.2 Support Vector Machines

The main idea of support vector machine classification is to construct a hyperplane to separate the two classes.

### 2.1.2.1 Linear Support Vector Machines

Suppose we have a data set  $D$  of  $l$  samples in an  $n$ -dimensional space belonging to two different classes (+1 and -1):

$$D = \{(\mathbf{x}_k, y_k) \mid k \in \{1, \dots, l\}, \mathbf{x}_k \in \mathfrak{R}^n, y_k \in \{+1, -1\}\} \quad (8)$$

The hyperplane in the  $n$  dimensional space is determined by the pair  $(\mathbf{w}, b)$  where  $\mathbf{w}$  is an  $n$ -dimensional vector orthogonal to the hyperplane and  $b$  is the offset constant. The hyperplane  $(\mathbf{w} \cdot \mathbf{x}) + b$  separates the data if and only if

$$\begin{aligned} (\mathbf{w} \cdot \mathbf{x}_i) + b &> 0 & \text{if } y_i = +1 \\ (\mathbf{w} \cdot \mathbf{x}_i) + b &< 0 & \text{if } y_i = -1. \end{aligned} \quad (9)$$

If we additionally require that  $\mathbf{w}$  and  $b$  be such that the point closest to the hyperplane has a distance of  $1/|\mathbf{w}|$ , then we have

$$\begin{aligned} (\mathbf{w} \cdot \mathbf{x}_i) + b &\geq 1 & \text{if } y_i = +1 \\ (\mathbf{w} \cdot \mathbf{x}_i) + b &\leq -1 & \text{if } y_i = -1 \end{aligned} \quad (10)$$

which is equivalent to

$$y_i [(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 \quad \forall i. \quad (11)$$

To find the optimal separating hyperplane, we have to find the hyperplane that maximizes the minimum distance between the hyperplane and any sample of training data (see Figure 2.2). The distance between two closest samples from different classes is

$$d(\mathbf{w}, b) = \min_{\{\mathbf{x}_i \mid y_i = 1\}} \frac{(\mathbf{w} \cdot \mathbf{x}_i) + b}{|\mathbf{w}|} - \max_{\{\mathbf{x}_i \mid y_i = -1\}} \frac{(\mathbf{w} \cdot \mathbf{x}_i) + b}{|\mathbf{w}|}. \quad (12)$$

From (12), we can see that the appropriate minimum and maximum values are  $\pm 1$ . Therefore, we need to maximize

$$d(\mathbf{w}, b) = \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}. \quad (13)$$

Thus, the problem is equivalent to:

- minimize  $\|\mathbf{w}\|^2/2$
- subject to the constraints
  - (1)  $y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 \quad \forall i.$

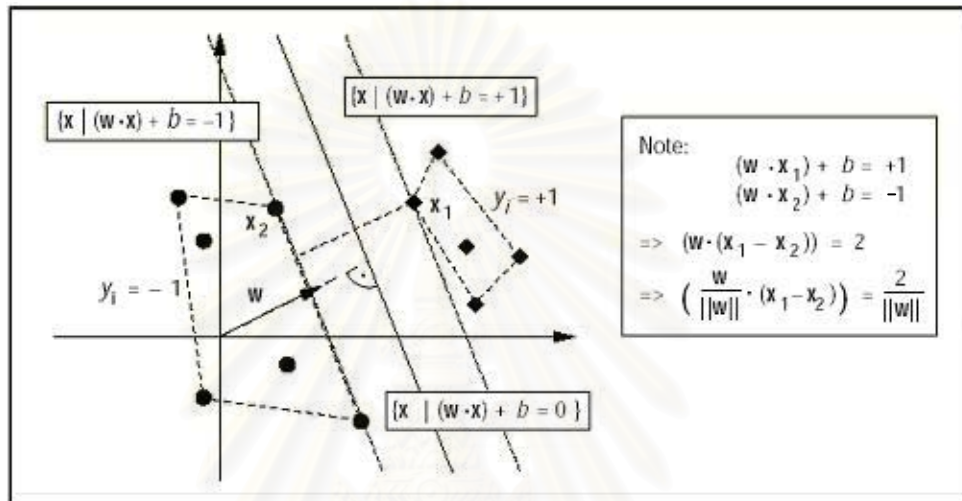


Figure 2.2: An example of SVM classification.

For non-separable case, the training data cannot be separated by a hyperplane without error. The previous constraints then must be modified. A penalty term consisting of the sum of deviations  $\xi_i$  from the boundary is added to the minimization problem. Now, the problem is to

- minimize  $\frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^l \xi_i$
- subject to the constraints
  - (1)  $y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 - \xi_i,$
  - (2)  $\xi_i \geq 0 \quad \forall i.$

The penalty term for misclassifying training samples is weighted by a constant  $C$ . Selecting a large value of  $C$  puts a high price on deviations and increases computation by effecting a more exhaustive search for ways to minimize the number of misclassified samples.

By forming the Lagrangian and solving the dual problem, this problem can be translated into:

- minimize

$$L(\mathbf{w}, b, \alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \quad (14)$$

- subject to the constraints:

$$(1) 0 \leq \alpha_i \leq C, \forall i$$

$$(2) \sum_{i=1}^l \alpha_i y_i = 0$$

where  $\alpha_i$  are called Lagrange multipliers. There is one Lagrange multiplier for each training sample. In the solution, those samples for which  $\alpha_i > 0$  are called *support vectors*, and are ones such that the equality in (11) holds. All other training samples having  $\alpha_i = 0$  could be removed from the training set without affecting the final hyperplane.

Let  $\alpha^0$ , an  $l$ -dimensional vector denote the minimum of  $L(\mathbf{w}, b, \alpha)$ . If  $\alpha_i^0 > 0$  then  $\mathbf{x}_i$  is a *support vector*. The optimal separating hyperplane  $(\mathbf{w}^0, b^0)$  can be written in terms of  $\alpha^0$  and the training data, specifically in terms of the support vectors:

$$\mathbf{w}^0 = \sum_{i=1}^l \alpha_i^0 y_i \mathbf{x}_i = \sum_{\text{support vectors}} \alpha_i^0 y_i \mathbf{x}_i. \quad (15)$$

$$b^0 = 1 - \mathbf{w}^0 \cdot \mathbf{x}_i \text{ for } \mathbf{x}_i \text{ with } y_i = 1 \text{ and } 0 < \alpha_i < C. \quad (16)$$

The optimal separating hyperplane classifies points according to the sign of  $f(\mathbf{x})$ ,

$$\begin{aligned} f(\mathbf{x}) &= \text{sign}(\mathbf{w}^0 \cdot \mathbf{x} + b^0) \\ &= \text{sign}\left(\sum_{\text{support vectors}} \alpha_i^0 y_i (\mathbf{x}_i \cdot \mathbf{x}) + b^0\right). \end{aligned} \quad (17)$$

Support vector  $\mathbf{x}_i$  with  $\alpha_i^0 = C$  may or may not be misclassified. All other  $\mathbf{x}_i$ 's are correctly classified.

In support vector methods one can control both parameters: in the separable case one obtains the unique solution which minimizes the empirical risk (down to zero) using a  $\Delta$ -margin separating hyperplane with the maximal margin (i.e.,

subset with the smallest VC dimension). In general case one obtains the unique solution when one chooses the value of the tradeoff parameter  $C$ .

### 2.1.2.2 Non-linear Support Vector Machines

The above algorithm is limited to linear separating hyperplanes. SVMs get around this problem by mapping the sample points into a higher dimensional space (sometimes called feature space) using a non-linear mapping chosen in advance. This is, we choose a map  $\Phi: \mathfrak{R}^n \mapsto H$  where the dimensionality of  $H$  is greater than  $n$ . We then seek a separating hyperplane in the higher dimensional space (see Figure 2.3); this is equivalent to a non-linear separating surface in  $\mathfrak{R}^n$ .

The data only ever appears in our training problem (14) in the form of dot products, so in the higher dimensional space we are only dealing with the data in the form  $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ . If the dimensionality of  $H$  is very large, then this could be difficult, or very computationally expensive. However, if we have a kernel function such that  $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ , then we can use this in place of  $\mathbf{x}_i \cdot \mathbf{x}_j$  everywhere in the optimization problem, and never need to know explicitly what  $\Phi$  is. Some widely used kernels are:

$$\text{Polynomial degree } d: k(\mathbf{x}, \mathbf{y}) = |\mathbf{x} \cdot \mathbf{y} + 1|^d \quad (18)$$

$$\text{Radial basis function: } k(\mathbf{x}, \mathbf{y}) = e^{-|\mathbf{x} - \mathbf{y}|^2 / c} \quad (19)$$

Implementation of a new set of decision functions can be done by changing only one function (kernel  $k(\mathbf{x}_i, \mathbf{x}_j)$ ), which defines the dot product in higher dimensional space.

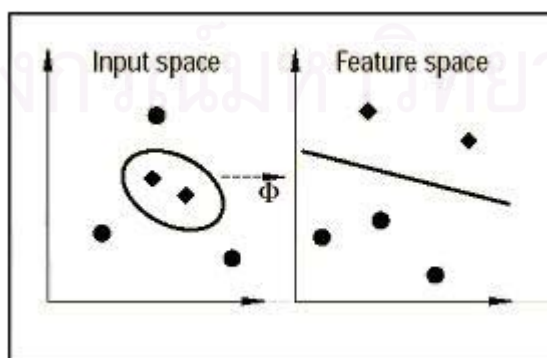


Figure 2.3: Feature space.

## 2.2 Literature Review

For multiclass SVMs, either several binary classifiers have to be constructed or a larger optimization problem is needed. In this thesis, we only deal with the combination of binary classifiers. In this section we discuss the conventional formulation to solve multiclass problems, for example, one-against-the-rest, one-against-one, the DDAG and the ADAG.

### 2.2.1 One-Against-the-Rest

The earliest implementation for multiclass SVMs is probably the one-against-the-rest method. It constructs  $k$  SVM classifiers where  $k$  is the number of classes. The  $i^{\text{th}}$  SVM is trained with all of the examples in the  $i^{\text{th}}$  class with positive labels, and all other examples with negative labels. Thus given  $l$  training data  $(x_1, y_1), \dots, (x_l, y_l)$ , where  $x_i \in R^n$ ,  $i = 1, \dots, l$  and  $y_i \in \{1, \dots, k\}$  is the class of  $x_i$ , the  $i^{\text{th}}$  SVM solves the following problem:

$$\begin{aligned} \min_{\mathbf{w}^i, b^i, \xi^i} \quad & \frac{1}{2}(\mathbf{w}^i)^T \mathbf{w}^i + C \sum_{j=1}^l \xi_j^i (\mathbf{w}^i)^T \\ & (\mathbf{w}^i)^T \Phi(\mathbf{x}_j) + b^i \geq 1 - \xi_j^i, \quad \text{if } y_j = i \\ & (\mathbf{w}^i)^T \Phi(\mathbf{x}_j) + b^i \leq -1 + \xi_j^i, \quad \text{if } y_j \neq i \\ & \xi_j^i \geq 0, \quad j = 1, \dots, l. \end{aligned} \quad (20)$$

Where the training data  $x_i$  are mapped to a higher dimensional space by the function  $\Phi$  and  $C$  is the penalty parameter.

Minimizing  $(1/2)(\mathbf{w}^i)^T \mathbf{w}^i$  means that we would like to maximize  $2/|\mathbf{w}^i|$ , the margin between two groups of data. When data are not linear separable, there is a penalty term  $C \sum_{j=1}^l \xi_j^i$  which can reduce the number of training errors. The basic concept behind SVMs is to search for a balance between the regularization term  $(1/2)(\mathbf{w}^i)^T \mathbf{w}^i$  and the training errors.

After solving (20), there are  $k$  decision functions

$$\begin{aligned} & (\mathbf{w}^1)^T \Phi(\mathbf{x}) + b^1 \\ & \vdots \\ & (\mathbf{w}^k)^T \Phi(\mathbf{x}) + b^k. \end{aligned}$$

We say  $x$  is in the class which has the largest value of the decision function



$$\text{class of } \mathbf{x} \equiv \arg \max_{i=1,\dots,k} ((\mathbf{w}^i)^T \Phi(\mathbf{x}) + b^i) . \quad (21)$$

This approach bears two main disadvantages: (i) separating one class from all others may be an unnecessarily hard problem that often requires us to apply very complex classification models, (ii) solving these subproblems can impose unacceptably high computational costs when kernel classifiers are applied to large-scale problems (Roth and Tsuda, 2001).

### 2.2.2 One-Against-One

This method constructs  $k(k-1)/2$  classifiers where each one is trained on data from two classes. The number of data needed for learning becomes smaller, so we can expect shorter training time. For training data from the  $i^{\text{th}}$  and the  $j^{\text{th}}$  classes, we solve the following binary classification problem:

$$\begin{aligned} \min_{\mathbf{w}^{ij}, b^{ij}, \xi_t^{ij}} \quad & \frac{1}{2} (\mathbf{w}^{ij})^T \mathbf{w}^{ij} + C \sum_t \xi_t^{ij} (\mathbf{w}^{ij})^T \\ & (\mathbf{w}^{ij})^T \Phi(\mathbf{x}_t) + b^{ij} \geq 1 - \xi_t^{ij}, \quad \text{if } y_t = i \\ & (\mathbf{w}^{ij})^T \Phi(\mathbf{x}_t) + b^{ij} \leq -1 + \xi_t^{ij}, \quad \text{if } y_t = j \\ & \xi_t^{ij} \geq 0. \end{aligned} \quad (22)$$

There are different methods for doing the future testing after all  $k(k-1)/2$  classifiers are constructed. If  $\text{sign}((\mathbf{w}^{ij})^T \Phi(\mathbf{x}) + b^{ij})$  says  $x$  is in the  $i^{\text{th}}$  class, then the vote for the  $i^{\text{th}}$  class is added by one. Otherwise, the vote for the  $j^{\text{th}}$  is increased by one. Then we predict  $x$  is in the class with the largest vote. The voting approach described above is also called the “Max Wins” strategy. In case that more than one class has the identical vote, a class will be randomly selected as the final output.

### 2.2.3 Decision Directed Acyclic Graph (DDAG)

Platt et al. (2000) presented a learning architecture, the Decision Directed Acyclic Graph (DDAG), which is used to combine many two-class classifiers into a multiclass classifier. For a  $k$ -class problem, its training phase is the same as the one-against-one method by solving  $k(k-1)/2$  binary SVMs, one for each pair of classes. However, in the testing phase, it uses a rooted binary directed acyclic graph which has

$k(k-1)/2$  internal nodes and  $k$  leaves (see Figure 2.4). Each node is a binary SVM of the  $i^{\text{th}}$  and  $j^{\text{th}}$  classes. Given a test sample  $x$ , starting at the root node, the binary decision function is evaluated. Then it moves to either left or right depending on the output value. Therefore, we go through a path before reaching a leaf node which indicates the predicted class.

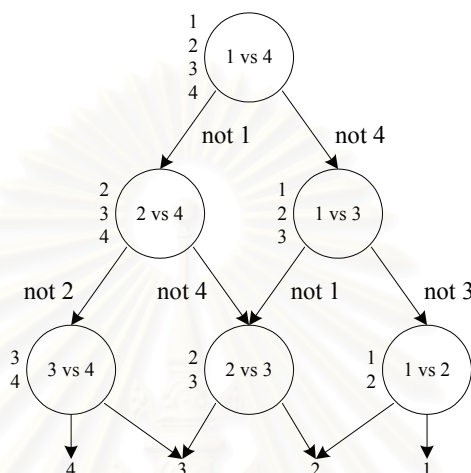


Figure 2.4: The DDAG finding the best class out of four classes.

There are some issues on the DDAG as pointed out by Ussivakul and Kijisirikul (2001, 2002). First, it gives outputs whose probabilities are not uniformly distributed, and thus its output depends on the sequence of binary classifiers in nodes, affecting reliability of the algorithm. In addition, the correct class placed in a node near the root node is clearly at disadvantage by comparison with the correct class near leaf nodes since it is exposed to higher risk of being incorrectly rejected. Second, the number of node evaluations for the correct class is unnecessary high. This results in higher cumulative error and lower the accuracy. The depth of the DDAG is  $k-1$  and this means that the number of times the correct class has to be tested against other classes, on average, scales linearly with  $k$ .

#### 2.2.4 Adaptive Directed Acyclic Graph (ADAG)

Ussivakul and Kijisirikul (2001, 2002) proposed an approach to alleviate the problem of the DDAG structure described above. An Adaptive DAG (ADAG) is a DAG with a reversed triangular structure. For a  $k$ -class problem, its training phase is the

same as the DDAG method by solving  $k(k-1)/2$  binary SVMs, one for each pair of classes. However, in the testing phase, the nodes are arranged in a reversed triangle with  $k/2$  nodes (rounded up) at the top,  $k/2^2$  nodes in the second layer and so on until the lowest layer of a final node. It has  $k-1$  internal nodes, each of which is labeled with an element of Boolean function (see Figure 2.5). Given a test example  $x$ , starting at the top level, the binary decision function is evaluated. The node is then exited via the outgoing edge with a message of the preferred class. In each round, the number of candidate classes is reduced by half. Based on the preferred classes from its parent nodes, the binary function of the next-level node is chosen. The reduction process continues until reaching the final node at the lowest level. The value of the decision function is the value associated with the message from the final leaf node. Like the DDAG, the ADAG requires only  $k-1$  decision nodes to be evaluated in order to derive an answer. Note that the correct class is evaluated against other classes for  $\log_2 k$  times or less, considerably lower than the number of evaluations required by the DDAG, which scales linearly with  $k$ .

Using the reversed triangular structure, the ADAG reduces the number of times the correct class is tested against other classes, and thus reduces the cumulative errors. However, there are still differences in accuracy between different sequences of nodes.

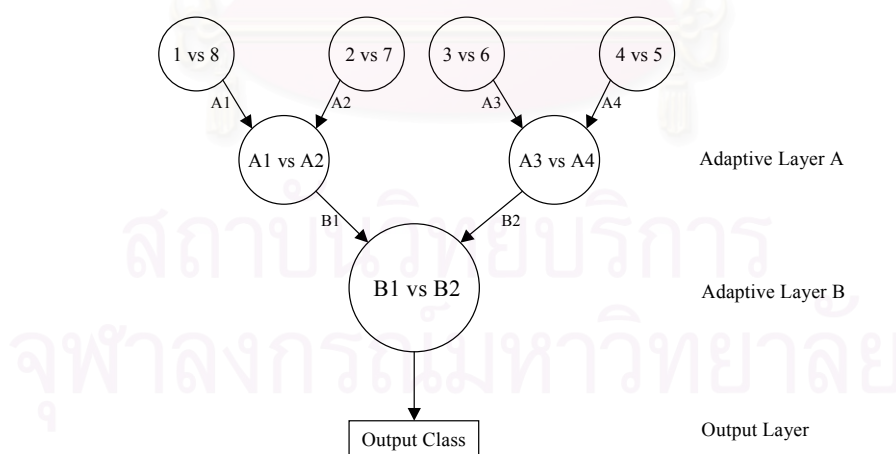


Figure 2.5: The structure of an ADAG for an 8-class problem.

## 2.2.5 Binary Tree SVMs

Schwenker (2000) proposed a hierarchical SVMs for multiclass classification using the binary tree structure. The major idea of hierarchical classification is first to make a coarse discrimination between confusion classes and then a finer discrimination within the confusion classes. In figure 2.6 an example of a hierarchical classifier is depicted. Each node within the graph represents a binary classifier which discriminates feature vectors of a confusion class into one of two smaller confusion classes or possibly into individual classes.

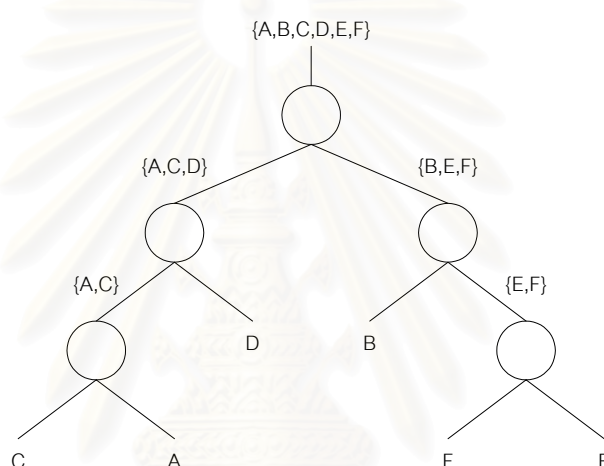


Figure 2.6: The structure of binary tree classifier.

The classification subtask is defined through the annotations of the incoming and outgoing edges of the node. For example, at the root of the tree the label of the incoming edge is  $\{A, \dots, F\}$ , so for this (sub-)tree a 6-class classification task is given. The edges to the children are annotated with  $\{A, C, D\}$  and  $\{B, E, F\}$ . This means that this SVM has to classify feature vectors into confusion class  $\{A, C, D\}$  or  $\{B, E, F\}$ . To achieve this, all members of the six classes  $\{A, \dots, F\}$  have to be re-labeled: feature vectors with class labels  $A, C$ , or  $D$  get the new label  $-1$  and those with class label  $B, E, F$  get the new label  $1$ . After this re-labeling procedure the SVM is trained. Note that re-labeling has to be done for training each classifier in each node of the tree.

However, the question how to construct the hierarchy of subsets of classes is not answered. In some cases it may be a priori defined.

## 2.2.6 Decision-Tree-Based Multiclass SVMs

Takahashi and Abe (2002) presented decision-tree-based multiclass SVMs to overcome the unclassifiable regions. Figure 2.7 shows an example of the unclassifiable regions for a 3-class problem. The data in the shaded regions cannot be classified.

Figure 2.8(a) shows an example of the division of the feature space, and figure 2.8(b) expresses this by a decision tree. In the training phase, the hyperplane  $f_1(x)$  which separates Class 1 from Classes 2 and 3 is calculated. For remaining classes (Class 2 and Class 3), the hyperplane  $f_2(x)$  which separates these two classes is calculated. In the testing phase, given an input  $x$ , we first calculate a value of  $f_1(x)$ . If it is positive,  $x$  is classified into Class 1, and if negative, calculate a value of  $f_2(x)$ . If it is positive,  $x$  is classified into Class 2, but if negative, classified into Class 3.

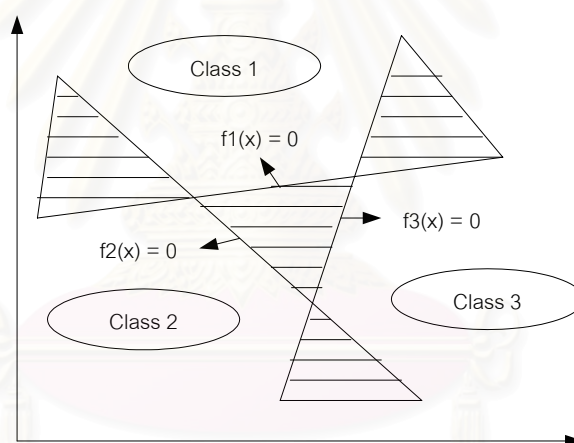


Figure 2.7: The existence of unclassifiable regions.

But a problem is how to determine the structure of the decision tree. The division of the feature space depends on the structure of a decision tree. To maintain high generalization ability, the most separable classes should be separated at the upper nodes of a decision tree. The Euclidean distance or Mahalanobis distance can be used as a separation measure.

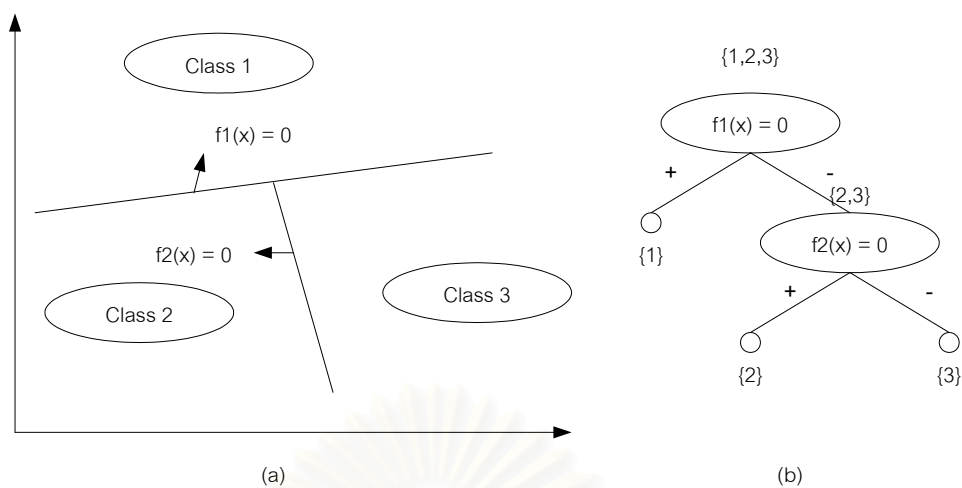


Figure 2.8: (a) An example of the division of feature space.

(b) Expression by a decision tree.

สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

## CHAPTER III

### REORDERING ADAPTIVE DIRECTED ACYCLIC GRAPH

This chapter gives details about the proposed method, Reordering Adaptive Directed Acyclic Graph (RADAG), which is the modification of the ADAG (described in the previous chapter). Our approach determines a best sequence of binary classifiers in nodes in the ADAG by dynamically reordering the sequence during classification process according to each test data. Only binary classifiers which have small generalization errors will be used in data classification.

#### 3.1 Reordering Adaptive Directed Acyclic Graph (RADAG)

From the structure of the ADAG (see Figure 2.2), the sequence of binary classifiers 1vs8, 2vs7, 3vs6, and 4vs5 may give different accuracy from the accuracy of the sequence 3vs4, 2vs8, 6vs7, and 1vs5. For example, suppose that a test example whose correct class is class 7, for the first sequence, the classifier 2vs7 may incorrectly eliminate the correct class while for the second sequence, the classifier 6vs7 may correctly classify the correct class. The first sequence causes wrong classification for the test example. This shows the dependency of the sequence of binary classifiers in nodes in the ADAG structure.

We propose a method, called *Reordering Adaptive Directed Acyclic Graph (RADAG)*, to improve the accuracy of the original ADAG. For a  $k$ -class problem, the RADAG's training phase is the same as the ADAG method by solving  $k(k-1)/2$  binary SVMs. However, the testing phase is organized as follows. The differences are the initialization of the binary classifiers in the top level and the order of sequences in lower levels (see Figure 3.1). In the first step, we use a reordering algorithm with minimum-weight perfect matching described in the next section to choose the optimal sequence to be the initial sequence. We use the sequence to evaluate every test example. In the second step, as in the ADAG, test points of the RADAG are evaluated against the decision nodes. In the third step, unlike the ADAG, the RADAG will reorder the sequence before processing in the next level by using the reordering algorithm with minimum-weight perfect matching. This sequence differs for each test example, and it

depends on the results of nodes from the previous level. The second and the third steps are repeated until there is only one class remains.

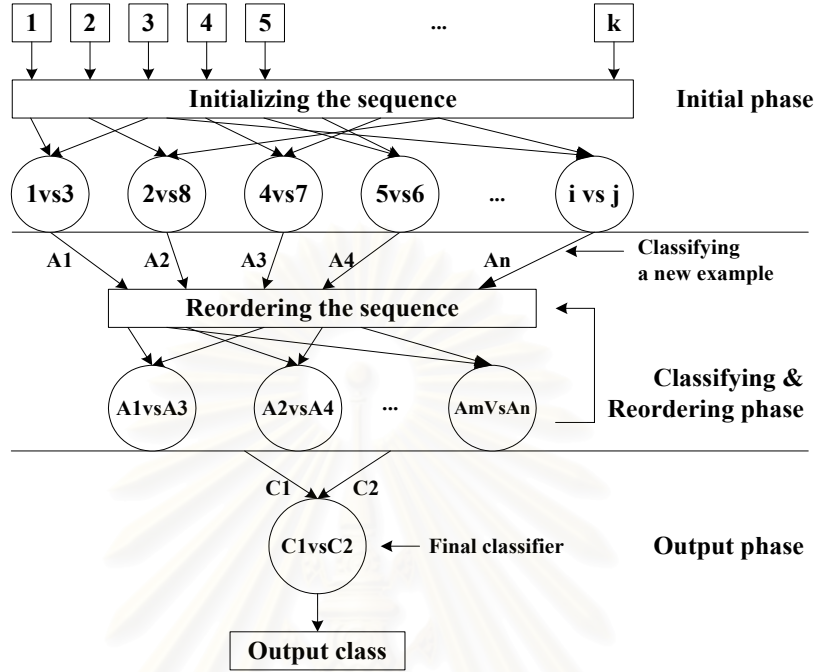


Figure 3.1: Classifying process of the RADAG.

## 3.2 Reordering Algorithm

### 3.2.1 Generalization Performance of Support Vector Machines

The ability of a hypothesis to correctly classify data not in the training set is known as its generalization (Cristianini and Shawe-Taylor, 2000). Generalization analysis of pattern classifiers is concerned with determining the factors that affect the accuracy of a pattern classifier. Generalization performance of SVMs can be approximated by bounding on the generalization error (Barlett and Shawe-Taylor, 1999).

Define the class  $F$  of real-valued functions on the ball of radius  $R$  in  $\mathfrak{R}^n$  as  $F = \{\mathbf{x} \mapsto \mathbf{w} \cdot \mathbf{x} : \|\mathbf{w}\| \leq 1, \|\mathbf{x}\| \leq R\}$ . There is a constant  $c$  such that, for all probability distributions, with probability at least  $1-\delta$  over  $l$  independently generated examples  $\mathbf{z}$ , if a classifier  $h = \text{sgn}(f) \in \text{sgn}(F)$  has margin at least  $\Delta$  on all the examples in  $\mathbf{z}$ , then the error of  $h$  is no more than

$$\frac{c}{l} \left( \frac{R^2}{\Delta^2} \log^2 l + \log \left( \frac{1}{\delta} \right) \right). \quad (23)$$



Furthermore, in case that the training data cannot be separated by the hyperplane without error, with probability at least  $1-\delta$ , every classifier  $h \in \text{sgn}(F)$  has error no more than

$$\frac{r}{l} + \sqrt{\frac{c}{l} \left( \frac{R^2}{\Delta^2} \log^2 l + \log \left( \frac{1}{\delta} \right) \right)} \quad (24)$$

where  $r$  is the number of labeled examples in  $\mathbf{z}$  with margin less than  $\Delta$ .

Below we show an example of the generalization error of classifiers. The experiment is based on the English letter image recognition dataset from the UCI Machine Learning Repository (Blake et al., 1998), which has 26 classes. Hence there are 325 classifiers. The classifiers are trained by using the Polynomial kernel of degree 3. In Figure 3.2, the generalization errors of all classifiers expressed by Equation (24) are depicted. The generalization errors of all classifiers are varying. The standard deviation of the generalization errors is 25.36 and the average of all of them is 28.82.

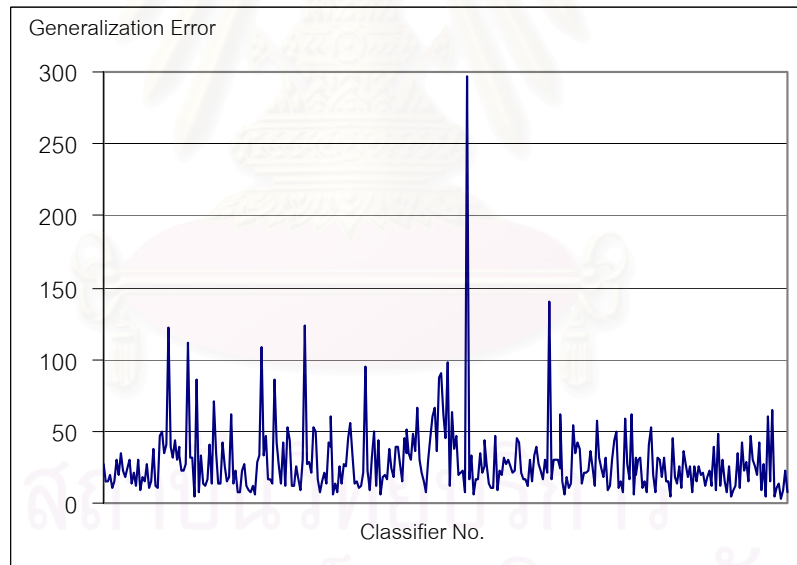


Figure 3.2: The generalization errors of 325 classifiers.

### 3.2.2 Reordering Algorithm with Minimum-Weight Perfect Matching

For the reason described above, we consider the generalization errors in order to choose the optimal sequence from all possible

$$\frac{k!}{2^{\lfloor \frac{k}{2} \rfloor} \left\lfloor \frac{k}{2} \right\rfloor !}$$

chance to predict the wrong class. Among  $k(k-1)/2$  classifiers,  $k/2$  classifiers which have small generalization errors will be considered to be used in data classification.

Let  $G$  be a graph with node set  $V$  and edge set  $E$ . Let  $c_e$  be the weight of  $e \in E$ . A perfect matching in a graph  $G$  is a subset of edges such that each node in  $G$  is met by exactly one edge in the subset. Given a real weight  $c_e$  for each edge  $e$  of  $G$ , the minimum weight perfect matching problem is to find a perfect matching  $M$  of minimum weight  $\sum(c_e : e \in M)$ .

For  $U \subseteq V$ , let  $E(U) = \{(i,j):(i,j) \in E, i \in U, j \in U\}$ .  $E(U)$  is the set of edges with both endpoints in  $U$ . The set of edges incident to node  $i$  in the node-edge incidence matrix is denoted by  $\delta(i)$ . The convex hull of perfect matchings on a graph  $G = (V, E)$  with  $|V|$  even is given by

$$\begin{aligned}
 & \text{a) } x \in R_+^m \\
 & \text{b) } \sum_{e \in \delta(v)} x_e = 1 \text{ for } v \in V \\
 & \text{c) } \sum_{e \in E(U)} x_e \leq \left\lfloor \frac{|U|}{2} \right\rfloor \text{ for all odd sets } U \subseteq V \text{ with } |U| \geq 3 \\
 & \text{or by (a),(b) and} \\
 & \text{d) } \sum_{e \in \delta(U)} x_e \geq 1 \text{ for all odd sets } U \subseteq V \text{ with } |U| \geq 3
 \end{aligned} \tag{25}$$

where  $|E| = m$  and  $x_e = 1$  means that  $e$  is in the matching. So the minimum weight of a perfect matching is at least as large as the value of

$$\min \sum_{e \in E} c_e x_e \tag{26}$$

where  $x$  satisfies “(a),(b) and (c)” or “(a),(b) and (d)” (Nemhauser and Wolsey, 1999).

Let  $G = (V, E)$  be a graph with node set  $V$  and edge set  $E$ . Each node in  $G$  denotes one class and each edge denotes one binary classifier which has a generalization error from equation (24) (see Figure 3.3(a)). Given a real weight  $c_e$  being generalization error for each edge  $e$  of  $G$ , the output of the reordering algorithm for graph  $G$  is a subset of edges with the minimum sum of generalization errors of all edges and each node in  $G$  is met by exactly one edge in the subset (see Figure 3.3(b)). Therefore, the reordering problem is to solve the linear program in Equation (26).

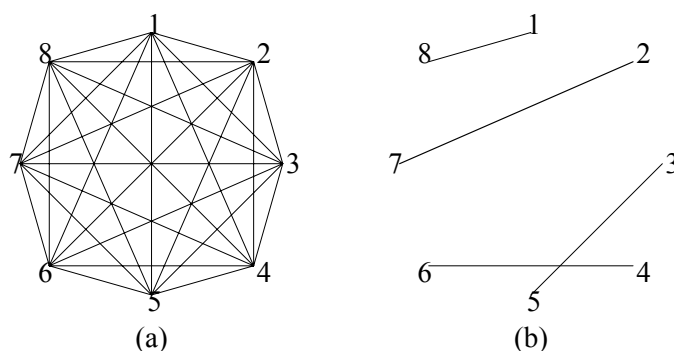


Figure 3.3: (a) A graph for an 8-class problem.  
 (b) An example of the output of the reordering algorithm.

One of the fundamental results in combinatorial optimization is the polynomial time *blossom* algorithm for computing minimum-weight perfect matchings by Edmonds. The straightforward implementation of Edmonds' original description of the algorithm can easily be seen to run in time bounded by  $O(n^2m)$ , where  $n$  is the number of nodes in the graph and  $m$  is the number of edges. This was improved by Lawler and by Gabow to  $O(n^3)$ , and later to  $O(nm \log n)$  by Galil, Micali and Gabow. A further improvement was made by Gabow, Galil and Spencer, lowering the bound to  $O(n(m \log \log \log \max\{m/n, 2\} + n + n \log n))$ . The  $\log \log \log$  term was then removed by Gabow, resulting in a bound of  $O(n(m+n \log n))$ . Gabow's bound is currently the best known result in terms of  $n$  and  $m$  [see (Cook and Rohe, 1997) for more details].

### 3.3 Time Complexity

The Max Wins needs  $O(n^2)$  number of comparisons for the problem with  $n$  classes. The DDAG reduces the number of comparisons down to  $O(n)$ . By reducing the depth of the path, the ADAG requires  $O(n)$  comparisons of binary classifiers with accuracy higher than that of the DDAG. Our approach needs a little time more than the ADAG for reordering the sequence. Note that, currently, the minimum-weight perfect matching algorithm, which is used in the reordering algorithm, runs in time bounded by  $O(n(m+n \log n))$  (Cook and Rohe, 1997), where  $n$  is the number of nodes (classes) in the graph and  $m = n(n-1)/2$  is the number of edges (binary classifiers). The RADAG will reorder the sequence in every level, except for the last level. The sequence of classes in the top level is reordered only one time and we use the sequence to evaluate every test

example. Hence for classifying each test data, we need  $\log_2 n - 2$  times of reordering, where each time the number of classes is reduced by half. Therefore, the running time of the RADAG is bounded by  $O(c_1 n) + O(c_2 n^3 \log_2 n)$ , where  $c_1$  is much larger than  $c_2$  (see section 4.2.2 for empirical results).

### 3.4 Estimating the Difference between Means of Generalization Errors

In this section we will explain test concerning means of generalization errors of classifiers of interest (the binary classifier of the correct class and other classes in the sequence), which are selected by the ADAG and the RADAG. For the ADAG, we randomly selected the classifiers to be used in data classification, whereas for the RADAG the classifiers are selected by using the minimum-weight perfect matching algorithm. This may be analyzed using a method called the *one tailed paired t test*.

To estimate the difference between two means, we wish to test

$$H_0 : \mu_1 - \mu_2 = 0,$$

$$H_1 : \mu_1 - \mu_2 > 0.$$

The point estimate of the difference  $\mu_1 - \mu_2$  of two population means is given by  $\overline{X}_1 - \overline{X}_2$ . The form of the paired t test is

$$t = \frac{\overline{X}_1 - \overline{X}_2}{S_d / \sqrt{n}} \quad (27)$$

where  $S_d$  is the standard deviation of the sample of differences  $\overline{X}_1 - \overline{X}_2$ .

Assume that the distribution of generalization errors is normal distribution (see figure 3.4). The simulation is based on the dataset which has 26 classes, and hence there are 325 binary classifiers. In the experiment, a generalization error is randomly selected to each classifier with equal probability, uniformly distributed. We examine 5,200 sequences of classifiers, where for the first 200 sequences we assume that the correct class is class 1, for the second 200 sequences we assume that the correct class is class 2 and so on. For the ADAG, we randomly selected 5,200 sequences of classifiers. For the RADAG, 5,200 sequences are selected by using the minimum-weight perfect matching algorithm so all sequences are the same. Then we compute sum of generalization errors of classifiers of interest that correspond to the binary classifier of the correct class and other classes in the sequence. The experiments

are repeated 100 times. Then we calculate the sample mean  $\bar{X}_1$  ( $\bar{X}_{ADAG}$ ) and  $\bar{X}_2$  ( $\bar{X}_{RADAG}$ ). Equation 27 gives

$$t = \frac{237132.1 - 95780.0}{5348.98 / \sqrt{100}} = 264.26.$$

Using  $\alpha = 0.001$  with a one-tailed test and degree of freedom = 99, we determine from the t table that the critical value is 3.175. Then the null hypothesis is rejected.

The formula for a 99% confidence interval for the difference between two means is

$$\left( (\bar{X}_1 - \bar{X}_2) - 2.365 \frac{S_d}{\sqrt{n}}, (\bar{X}_1 - \bar{X}_2) + 2.365 \frac{S_d}{\sqrt{n}} \right). \quad (28)$$

Equation 28 gives (140087.1, 142617.1). The interval does not include 0, and thus the null hypothesis is rejected at the 1% significance level. These mean that our method obtains classifiers whose generalization errors are statistically significantly lower than the ADAG.

In addition, according to the assumption mentioned above, the empirical results show that the minimum-weight perfect matching always selects classifiers with generalization errors less than half of maximum generalization error.

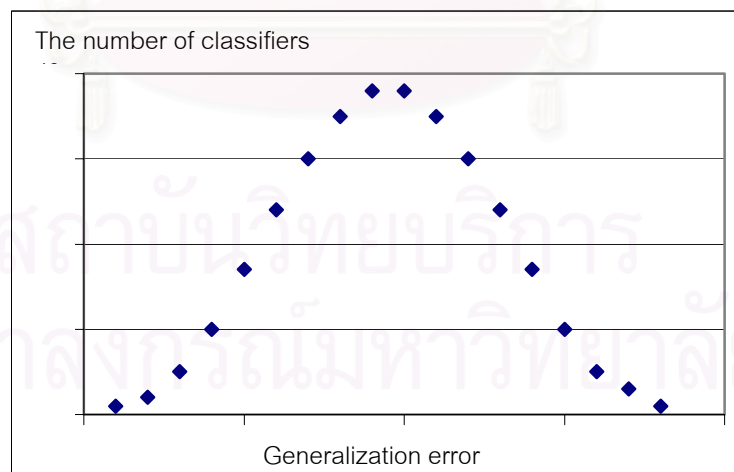


Figure 3.4: The distribution of the generalization errors.

## CHAPTER IV

### EXPERIMENTAL SETTING AND RESULTS

This chapter presents the experimental setting and results. We compare the classification accuracy of the RADAG with those of the original ADAG and the Max Wins algorithm. We also compare the running time of our approach with that of Max Wins which is probably the currently most accurate method for multiclass SVMs. Moreover, we use a one-tailed paired t-test to estimate the difference between accuracies of the RADAG and the DDAG, the ADAG and the Max Wins.

#### 4.1 Dataset and Experimental Setting

The experiments are based on several datasets from the UCI Machine Learning Repository (Blake et al., 1998) including glass, satimage, segment, shuttle, vowel, soybean, letter and isolet (see Table 4.1). For the glass and segment problems, there is no provided test data so we used 5-fold cross validation. For the soybean problem, we discarded the last four classes because of missing values. In addition, we examined our method with Thai printed character recognition dataset (Pichitdej, 2001), which has 68 classes including 44 consonants, and 26 vowels and tonal masks. These datasets are different in the number of classes, the number of dimensions, and sizes. 68 classes is the most classes and 617 dimensions is the most dimensions which are used in the experiment.

In these experiments we scaled both training data and test data to be in  $[-1,1]$  and employed Polynomial and RBF kernels. In the training phase, the  $k(k-1)/2$  binary classifiers were constructed by using the software package called SVM<sup>light</sup> version 5.0 (Joachims, 1999). In the experiments, we compared the accuracy of four algorithms, i.e., the DDAG, the original ADAG, the Max Wins, and the RADAG. For the DDAG and the ADAG, we examined all possible sequences for datasets having not more than 8 classes, whereas we randomly selected 50,000 sequences for datasets having more than 8 classes. We then calculate the average of accuracy of these sequences. We also compare the computational time between the RADAG and the Max Wins.

Table 4.1: Description of the datasets used in the experiments.

Dataset	#training data	#test data	#class	#dimension
Glass	214	5-fold	6	9
Satimage	4,435	2,000	6	36
Segment	2,310	5-fold	7	18
Shuttle	43,500	14,500	7	9
Vowel	528	462	11	10
Soybean	290	340	15	35
Letter	15,963	4,037	26	16
Isolet	6,238	1,559	26	617
ThaiPrintedCharacter1	3,264	3,264	68	128
ThaiPrintedCharacter2	3,264	3,264	68	128

## 4.2 Experimental Results

Table 4.2 to Table 4.11 present the results of comparing three methods including the ADAG, the Max Wins and the RADAG. The best accuracy among three methods for each value of the parameter of the kernel is illustrated in bold-face. The highest accuracy of each kernel is depicted by asterisk.

### 4.2.1 The Accuracy of Classification

- **The glass dataset**

This dataset is about glass identification. Table 4.2 shows the accuracy of classification. As shown in the table, the RADAG gives higher accuracy in RBF kernel whereas in Polynomial kernel the ADAG gives higher accuracy.

- **The satimage dataset**

This dataset consists of the multi-spectral values of pixels in 3x3 neighborhoods in a satellite image, and the classification associated with the central pixel in each neighborhood. Table 4.3 shows the accuracy of classification. As shown in

the table, the RADAG gives higher accuracy in Polynomial kernel whereas in RBF kernel the Max Wins gives higher accuracy.

Table 4.2: A comparison of the accuracy of classification of the glass dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	<b>71.135*</b>	71.078	71.063	0.01	71.953	71.871	<b>71.971</b>
3	68.801	<b>68.989</b>	68.716	0.02	71.064	70.918	<b>71.506</b>
4	69.400	<b>69.598</b>	68.239	0.03	71.054	70.918	<b>71.506</b>
5	69.947	<b>70.173</b>	69.158	0.04	70.784	70.616	<b>71.041</b>
6	69.318	<b>69.553</b>	69.158	0.05	70.144	69.943	<b>71.052</b>
7	67.572	<b>67.902</b>	67.298	0.06	72.314	72.088	<b>72.912</b>
8	68.243	<b>68.463</b>	68.217	0.07	72.604	72.385	<b>73.377</b>
				0.08	72.759	72.506	<b>73.377</b>
				0.09	71.943	73.238	<b>74.319*</b>
				0.10	71.684	71.367	<b>72.447</b>

Table 4.3: A comparison of the accuracy of classification of the satimage dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	86.986	87.152	<b>87.450</b>	0.5	89.083	89.123	<b>89.150</b>
3	87.219	87.248	<b>87.350</b>	1.0	90.329	90.340	<b>90.350</b>
4	87.562	87.545	<b>87.750</b>	1.5	90.819	90.834	<b>90.950</b>
5	88.232	88.198	<b>88.500</b>	2.0	91.530	91.542	<b>91.600</b>
6	88.430	88.453	<b>88.900*</b>	3.0	91.968	<b>91.984*</b>	91.950
7	87.941	87.957	<b>88.150</b>	4.0	91.804	<b>91.820</b>	91.800
8	87.720	87.757	<b>87.850</b>	5.0	91.483	91.496	<b>91.500</b>

- **The segment dataset**

This dataset is about image segmentation data. The instances were drawn randomly from a database of 7 outdoor images. The images were hand



segmented to create a classification for every pixel. Table 4.4 shows the accuracy of classification. In this dataset, the RADAG gives higher accuracy in Polynomial kernel whereas in RBF kernel the Max Wins gives higher accuracy.

Table 4.4: A comparison of the accuracy of classification of the segment dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	96.736	96.743	<b>96.970</b>	0.5	97.052	<b>97.071</b>	97.013
3	96.400	96.428	<b>96.623</b>	0.6	97.080	<b>97.089</b>	97.056
4	96.632	96.658	<b>96.840</b>	0.7	97.282	<b>97.298*</b>	97.273
5	96.932	96.942	<b>97.056</b>	0.8	97.249	97.241	<b>97.273</b>
6	97.084	97.080	<b>97.143</b>	0.9	97.182	97.171	<b>97.186</b>
7	97.095	97.083	<b>97.230</b>	1.0	<b>97.090</b>	97.077	97.056
8	97.407	97.379	<b>97.489*</b>	1.5	97.030	97.046	<b>97.056</b>

- The shuttle dataset

Table 4.5 shows the accuracy of classification. These three methods give comparable accuracy in both Polynomial kernel and RBF kernel.

Table 4.5: A comparison of the accuracy of classification of the shuttle dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	99.833	<b>99.834</b>	<b>99.834</b>	0.5	99.836	<b>99.834</b>	<b>99.834</b>
3	99.866	<b>99.869</b>	<b>99.869</b>	1.0	<b>99.862</b>	<b>99.862</b>	<b>99.862</b>
4	99.865	<b>99.869</b>	<b>99.869</b>	1.5	<b>99.879</b>	<b>99.879</b>	<b>99.876</b>
5	<b>99.890</b>	<b>99.890</b>	<b>99.890</b>	2.0	<b>99.883</b>	<b>99.883</b>	<b>99.883</b>
6	<b>99.903</b>	<b>99.903</b>	<b>99.903</b>	3.0	<b>99.897*</b>	<b>99.897*</b>	<b>99.897*</b>
7	<b>99.917</b>	<b>99.917</b>	<b>99.917</b>	4.0	99.893	<b>99.897*</b>	99.890
8	<b>99.924*</b>	<b>99.924*</b>	<b>99.924*</b>	5.0	99.889	<b>99.892</b>	99.883

- The vowel dataset

This dataset is about speaker independent recognition of the eleven steady state vowels of British English. Table 4.6 shows the accuracy of classification. As shown in the table, the RADAG also gives higher accuracy in both Polynomial kernel and RBF kernel.

Table 4.6: A comparison of the accuracy of classification of the vowel dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	63.862	<b>63.918</b>	63.420	0.1	61.344	61.169	<b>62.771</b>
3	64.293	64.329	<b>64.502*</b>	0.2	65.589	65.340	<b>67.100*</b>
4	62.663	<b>62.773</b>	61.905	0.3	65.400	65.203	<b>66.450</b>
5	60.401	60.346	<b>60.606</b>	0.4	63.928	64.108	<b>65.368</b>
6	58.556	58.649	<b>58.874</b>	0.5	63.676	63.677	<b>64.286</b>
7	56.720	<b>56.725</b>	56.710	1.0	61.066	61.214	<b>61.255</b>
8	<b>55.652</b>	55.647	55.411	1.5	60.905	60.833	<b>61.472</b>

Table 4.7: A comparison of the accuracy of classification of the soybean dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	89.643	89.768	<b>90.588</b>	0.04	89.666	89.738	<b>90.294</b>
3	90.402	90.471	<b>91.176*</b>	0.05	90.340	90.379	<b>90.882*</b>
4	90.116	89.968	<b>90.588</b>	0.06	90.094	90.068	<b>90.588</b>
5	90.446	90.353	<b>90.588</b>	0.07	90.388	90.362	<b>90.882*</b>
6	89.584	89.506	<b>89.706</b>	0.08	90.412	90.468	<b>90.588</b>
7	<b>88.686</b>	88.668	88.529	0.09	90.399	90.468	<b>90.588</b>
8	<b>87.804</b>	87.785	87.647	0.1	90.097	<b>90.174</b>	90.000
				0.2	86.653	86.682	<b>86.765</b>

- **The soybean dataset**

This dataset is about soybean disease. Table 4.7 shows the accuracy of classification. As shown in the table, the RADAG also gives higher accuracy in both Polynomial kernel and RBF kernel.

- **The letter dataset**

This dataset is about English letter image recognition. Table 4.8 shows the accuracy of classification. In this dataset, the RADAG gives higher accuracy in RBF kernel whereas in Polynomial kernel the Max Wins gives higher accuracy.

Table 4.8: A comparison of the accuracy of classification of the letter dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	95.110	<b>95.522</b>	95.467	0.5	96.651	96.674	<b>96.854</b>
3	95.984	<b>96.125*</b>	95.987	1.0	97.381	97.427	<b>97.449</b>
4	95.892	<b>96.120</b>	96.111	1.5	97.589	<b>97.629</b>	97.548
5	95.520	94.622	<b>95.888</b>	2.0	97.632	<b>97.661</b>	97.622
6	95.349	<b>95.551</b>	95.417	3.0	97.909	97.918	<b>97.969*</b>
7	94.783	94.975	<b>95.046</b>	4.0	97.797	<b>97.815</b>	97.795
8	93.829	94.072	<b>94.204</b>	5.0	97.675	<b>97.689</b>	97.672

- **The isolet dataset**

This dataset is about isolated English letter speech recognition. Table 4.9 shows the accuracy of classification. The RADAG gives higher accuracy in both Polynomial kernel and RBF kernel.

- **The Thai printed character 1 dataset**

This dataset is about Thai printed character recognition. For the training set, the characters were printed by laser printer with 600 dpi resolution, and then they were copied by a copier machine with saturated ink. For the test set, the characters were printed by laser printer with 600 dpi resolution. The characters in both training set and test sets were scanned by scanner with 200 dpi resolution. Table 4.10 shows the

accuracy of classification. The RADAG gives higher accuracy in Polynomial kernel whereas in RBF kernel the ADAG gives higher accuracy.

Table 4.9: A comparison of the accuracy of classification of the isolet dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	96.553	96.586	<b>96.600</b>	0.001	96.551	96.554	<b>96.665</b>
3	97.030	97.040	<b>97.049*</b>	0.002	96.606	96.619	<b>96.793</b>
4	96.990	<b>97.024</b>	96.985	0.003	96.898	96.889	<b>97.115*</b>
5	96.653	<b>96.695</b>	96.665	0.004	96.742	96.726	<b>96.857</b>
6	96.667	96.666	<b>96.729</b>	0.005	96.699	96.681	<b>96.857</b>
7	96.132	96.133	<b>96.151</b>	0.01	96.932	96.916	<b>96.985</b>
8	95.492	95.488	<b>95.510</b>	0.02	<b>96.731</b>	<b>96.731</b>	96.729
				0.03	95.684	95.680	<b>95.767</b>

Table 4.10: A comparison of the accuracy of classification of the Thai printed character 1 dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	93.316	93.280	<b>93.444*</b>	0.0007	96.607	<b>96.616</b>	96.538
3	81.938	81.927	<b>82.077</b>	0.0008	96.587	96.595	<b>96.599</b>
4	63.857	<b>63.858</b>	63.817	0.0009	96.608	<b>96.626</b>	96.599
5	48.759	48.760	<b>48.775</b>	0.001	96.677	<b>96.688</b>	96.661
				0.002	<b>97.078</b>	97.018	97.028
				0.003	<b>97.084*</b>	97.080	97.028
				0.004	96.782	96.782	<b>96.783</b>

- The Thai printed character 2 dataset

This dataset is about Thai printed character recognition. For the training set, the characters were printed by laser printer with 600 dpi resolution, and then they were copied by a copier machine with saturated ink. For the test set, the characters

were copied by a copier machine with pale ink. The characters in both training set and test sets were scanned by scanner with 200 dpi resolution. Table 4.11 shows the accuracy of classification. The RADAG gives higher accuracy in both Polynomial kernel and RBF kernel.

Table 4.11: A comparison of the accuracy of classification of the Thai printed character 2 dataset.

Polynomial kernel				RBF kernel			
d	ADAG	Max Wins	RADAG	c	ADAG	Max Wins	RADAG
2	98.994	98.989	<b>99.112*</b>	0.0007	99.061	99.058	<b>99.173</b>
3	98.129	98.123	<b>98.254</b>	0.0008	99.193	99.192	<b>99.265</b>
4	94.710	94.709	<b>94.761</b>	0.0009	99.193	99.192	<b>99.234</b>
5	88.592	88.584	<b>88.634</b>	0.001	99.224	99.223	<b>99.265</b>
				0.002	99.265	99.263	<b>99.357</b>
				0.003	99.306	99.304	<b>99.357</b>
				0.004	99.387	99.385	<b>99.418*</b>

- **Summary**

Tables 4.12 and 4.13 present the summary of the comparison between the ADAG, the Max Wins and the RADAG for Polynomial and RBF kernels, respectively. We present the optimal parameters ( $d$  and  $c$  in Equations (18) and (19)) of the kernels and the corresponding accuracies for each method. The results show that our method yields highest accuracy in almost all of datasets. This shows the effectiveness of the RADAG.

Tables 4.14 and 4.15 present the comparison between the RADAG and the maximum and the minimum accuracy of the ADAG for Polynomial and RBF kernels, respectively. We also show the corresponding standard deviation for the accuracy of classification of the ADAG. The optimal parameters ( $d$  and  $c$  in Equations (18) and (19)) of the kernels are the same as in Tables 4.12 and 4.13. The results show that the RADAG not only gives higher accuracy than the average accuracy of the ADAG (see Tables 4.12 and 4.13), but also provides accuracy equal to the maximum accuracy of

the ADAG in the satimage, soybean and Thai printed character 2 datasets for the Polynomial kernel and in the glass, soybean, isolet and Thai printed character 2 datasets for the RBF kernel (see Tables 4.14 and 4.15). For the shuttle dataset, the standard deviation for the accuracy of classification equals zero, so the accuracy of these three methods are the same.

Table 4.12: A comparison of the accuracy of classification using the Polynomial kernel.

Dataset	d	ADAG	d	Max Wins	D	RADAG
Glass	2	<b>71.135</b>	2	71.078	2	71.063
Satimage	6	88.430	6	88.453	6	<b>88.900</b>
Segment	8	97.408	8	97.379	8	<b>97.489</b>
Shuttle	8	<b>99.924</b>	8	<b>99.924</b>	8	<b>99.924</b>
Vowel	3	64.293	3	64.329	2	<b>64.502</b>
Soybean	5	90.446	3	90.471	3	<b>91.176</b>
Letter	3	95.984	3	<b>96.125</b>	4	96.111
Isolet	3	97.030	3	97.040	3	<b>97.049</b>
ThaiPrintedCharacter1	2	93.316	2	93.280	2	<b>93.444</b>
ThaiPrintedCharacter2	2	98.994	2	98.989	2	<b>99.112</b>

Table 4.13: A comparison of the accuracy of classification using the RBF kernel.

Dataset	c	ADAG	c	Max Wins	C	RADAG
Glass	0.08	72.759	0.09	73.238	0.09	<b>74.319</b>
Satimage	3.0	91.968	3.0	<b>91.984</b>	3.0	91.950
Segment	0.7	97.282	0.7	<b>97.298</b>	0.7	97.273
Shuttle	3.0	<b>99.897</b>	3.0	<b>99.897</b>	3.0	<b>99.897</b>
Vowel	0.2	65.589	0.2	65.340	0.2	<b>67.100</b>
Soybean	0.08	90.412	0.08	90.468	0.07	<b>90.882</b>
Letter	3.0	97.909	3.0	97.918	3.0	<b>97.969</b>
Isolet	0.01	96.932	0.01	96.916	0.003	<b>97.115</b>
ThaiPrintedCharacter1	0.003	<b>97.084</b>	0.003	97.080	0.003	97.028
ThaiPrintedCharacter2	0.004	99.387	0.004	99.385	0.004	<b>99.418</b>

Table 4.14: A comparison between the RADAG and the minimum and maximum accuracies of the ADAG using the Polynomial kernel.

Dataset	ADAG			RADAG
	Max	Min	Std.	
Glass	73.400	69.656	1.314	71.063
Satimage	<u>88.900</u>	87.950	0.265	<u>88.900</u>
Segment	97.792	97.100	0.237	97.489
Shuttle	<u>99.924</u>	99.924	0.000	<u>99.924</u>
Vowel	65.801	62.987	0.424	64.502
Soybean	<u>91.176</u>	90.000	0.280	<u>91.176</u>
Letter	96.532	95.417	0.141	96.111
Isolet	97.308	96.923	0.072	97.049
ThaiPrintedCharacter1	93.687	93.043	0.080	93.444
ThaiPrintedCharacter2	<u>99.112</u>	98.897	0.058	<u>99.112</u>

Table 4.15: A comparison between the RADAG and the minimum and maximum accuracies of the ADAG using the RBF kernel.

Dataset	ADAG			RADAG
	Max	Min	Std.	
Glass	<u>74.308</u>	71.971	0.928	<u>74.319</u>
Satimage	92.100	91.850	0.086	91.950
Segment	97.446	97.143	0.125	97.273
Shuttle	<u>99.897</u>	99.897	0.000	<u>99.897</u>
Vowel	69.264	63.203	0.816	67.100
Soybean	<u>90.882</u>	90.000	0.175	<u>90.882</u>
Letter	98.043	97.845	0.030	97.969
Isolet	<u>97.115</u>	96.859	0.057	<u>97.115</u>
ThaiPrintedCharacter1	97.242	96.997	0.047	97.028
ThaiPrintedCharacter2	<u>99.418</u>	99.357	0.025	<u>99.418</u>

Another advantage of our method compared to the DDAG and the ADAG is that our method always provides one best accuracy for each dataset using the reordering algorithm, whereas, depending on the sequence of classes, the DDAG and the ADAG may give low accuracies. This shows the effectiveness of the RADAG.

#### 4.2.2 The Computational Time

Tables 4.16 and 4.17 present the comparison of the running time between the RADAG and the Max Wins for Polynomial and RBF kernels by using a 400 MHz Pentium II processor. There is no running time of the glass dataset because it has too few test examples to measure the time. The running time of the RADAG consists of the running time for classifying and reordering. The classifying time of the RADAG equals the running time of the ADAG. The results show that our method requires low running time in all datasets, especially when the number of classes and/or the number of dimensions are relatively large. For a  $k$  class problem, the Max Wins requires  $k(k-1)/2$  classifiers for the classification whereas the RADAG requires only  $k-1$  classifiers. Hence the larger the number of classes the more running time the Max Wins requires than the RADAG. Moreover, the number of dimensions affects the running time of each classifier. Hence the larger the number of dimensions the more running time the Max Wins requires than the RADAG. For the RADAG, the number of classes affects the running time for reordering. However, it takes a little time even when there are many classes.



Table 4.16: A comparison of the computational time using the Polynomial kernel.

Dataset	#test data	#class	#dimension	d	RADAG			Max Wins
					Classifying (seconds)	Reordering (seconds)	Total (seconds)	Classifying (seconds)
Satimage	2,000	6	36	6	1.90	0.50	2.40	9.47
Segment	462	7	18	8	0.11	0.08	0.19	0.41
Shuttle	14,500	7	9	8	1.75	3.38	5.13	5.15
Vowel	462	11	10	2	0.12	0.25	0.37	0.61
Soybean	340	15	35	3	0.30	0.25	0.55	1.86
Letter	4,037	26	16	4	8.48	4.20	12.68	125.58
Isolet	1,559	26	617	3	116.02	1.48	117.50	1,671.98
ThaiPrintedCharacter1	3,264	68	128	3	94.63	13.83	108.46	2,996.64
ThaiPrintedCharacter2	3,264	68	128	3	96.41	13.19	109.60	3,042.98

Table 4.17: A comparison of the computational time using the RBF kernel.

Dataset	#test data	#class	#dimension	c	RADAG			Max Wins
					Classifying (seconds)	Reordering (seconds)	Total (seconds)	Classifying (seconds)
Satimage	2,000	6	36	3.0	11.75	0.51	12.26	37.13
Segment	462	7	18	0.7	0.24	0.10	0.34	0.82
Shuttle	14,500	7	9	3.0	3.36	0.63	3.99	9.27
Vowel	462	11	10	0.2	0.10	0.27	0.37	0.61
Soybean	340	15	35	0.07	0.32	0.45	0.77	2.20
Letter	4,037	26	16	3.0	62.27	3.69	65.96	802.85
Isolet	1,559	26	617	0.01	100.42	1.60	102.02	1,369.11
ThaiPrintedCharacter1	3,264	68	128	0.002	86.25	16.46	102.71	2,772.18
ThaiPrintedCharacter2	3,264	68	128	0.001	55.75	14.37	70.12	1,877.77

### 4.3 Evaluate the RADAG

#### 4.3.1 Estimating the Difference between Accuracies

To estimate the difference between accuracies, we added up the training set and test set into one set. Then we use a  $k$ -fold cross-validation method in which the set is partitioned into  $k$  disjoint, equal-sized subsets. In this  $k$ -fold cross-validation approach, each example from the set is used exactly once in a test set, and  $k-1$  times in a training set (Mitchell, 1997). In our experiment, we use 5-fold crsss-validation for the glass dataset and 10-fold crsss-validation for all others. Tables 4.18 and 4.19 present the comparison between the ADAG, the Max Wins and the RADAG for Polynomial and RBF kernels, respectively. We also compare the accuracy of classification with that of the DDAG. The optimal parameters ( $d$  and  $c$  in Equations (18) and (19)) of the kernels are the same as in Tables 4.12 and 4.13. The best accuracy among these methods is illustrated in bold-face. \*\*\*\*, \*\*\*, \*\* and \* in the tables mean 99.99%, 99%, 95%, and 90% confidence interval for estimating the difference between accuracies of three algorithms and the RADAG using a one-tailed paired t-test.

For estimating the difference between errors of two learning methods, the mean difference  $\bar{Y}$  in errors from all disjoint subsets is returned as an estimate of the difference between the two learning algorithms (Mitchell, 1997). The approximate  $N\%$  confidence interval for estimating the difference using  $\bar{Y}$  is given by

$$\left(\bar{Y} - t_{N,k-1} S_{\bar{Y}}, \bar{Y} + t_{N,k-1} S_{\bar{Y}}\right) \quad (24)$$

where  $\bar{Y}$  is the sample mean defined as

$$\bar{Y} = \frac{1}{k} \sum_{i=1}^k Y_i$$

$Y_i$  is the difference in error between two learning methods from the  $i^{\text{th}}$  subset, and  $S_{\bar{Y}}$  is the estimated standard deviation of the sample mean defined as

$$S_{\bar{Y}} \equiv \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^k (Y_i - \bar{Y})^2}.$$

Table 4.18: A comparison of estimating the difference between accuracies of classification using the Polynomial kernel.

Dataset	DDAG	ADAG	Max Wins	RADAG
Glass	71.069	<b>71.135</b>	71.078	71.063
Satimage	89.599	89.622	89.615	<b>89.681</b>
Segment	97.360**	97.383*	97.351**	<b>97.533</b>
Shuttle	99.919**	99.922**	99.923*	<b>99.930</b>
Vowel	98.872*	98.894*	98.901*	<b>98.990</b>
Soybean	92.202	92.281	92.470	<b>92.698</b>
Letter	95.994****	96.379***	96.512*	<b>96.674</b>
Isolet	97.484	97.485	97.488	<b>97.499</b>
ThaiPrintedCharacter1	99.239	99.242	<b>99.246</b>	99.234
ThaiPrintedCharacter2	99.657	99.670	<b>99.677</b>	99.617

Table 4.19: A comparison of estimating the difference between accuracies of classification using the RBF kernel.

Dataset	DDAG	ADAG	Max Wins	RADAG
Glass	72.850*	72.759*	73.238*	<b>74.319</b>
Satimage	92.129	92.141	92.148	<b>92.152</b>
Segment	97.652	97.650	<b>97.656</b>	97.576
Shuttle	99.926**	99.927*	99.928	<b>99.931</b>
Vowel	98.965*	98.975*	98.980*	<b>99.091</b>
Soybean	91.739**	92.570*	92.533*	<b>93.016</b>
Letter	95.994****	96.379****	96.512*	<b>96.634</b>
Isolet	97.517	97.523	97.527	<b>97.589</b>
ThaiPrintedCharacter1	<b>99.387</b>	<b>99.387</b>	<b>99.387</b>	<b>99.387</b>
ThaiPrintedCharacter2	<b>99.663</b>	<b>99.663</b>	<b>99.663</b>	<b>99.633</b>

The results show that our method performs statistically significantly better than the other methods in the segment, shuttle, vowel and letter problems in case of the Polynomial kernel. In case of the RBF kernel our method performs statistically significantly better than the other methods in the glass, shuttle, vowel, soybean and letter problems.

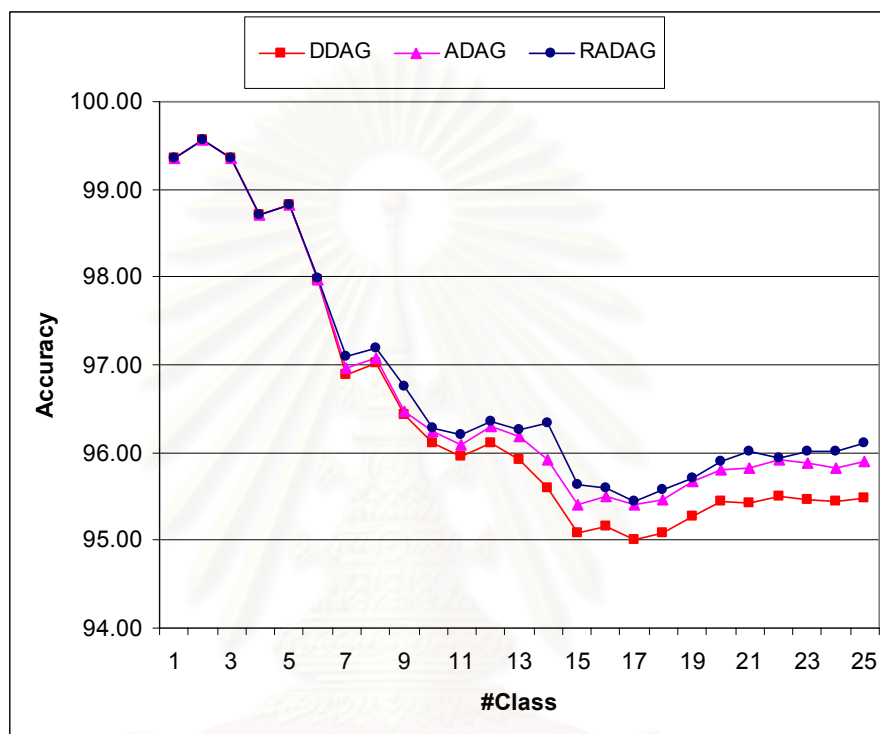
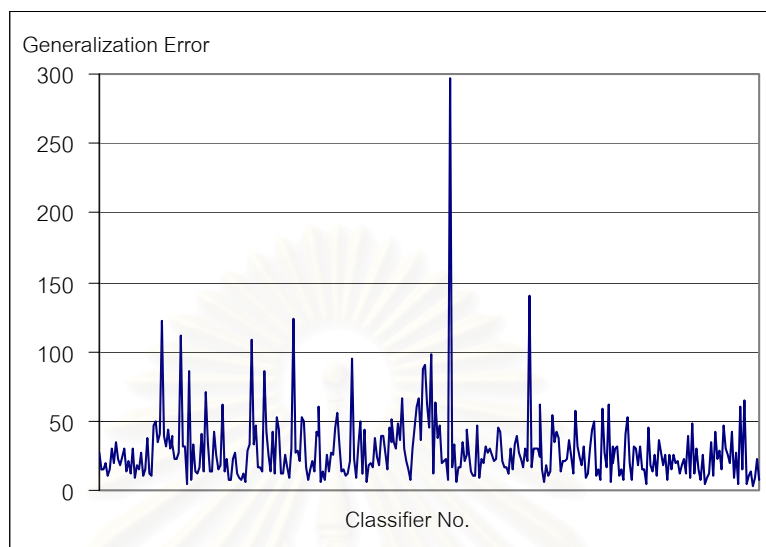


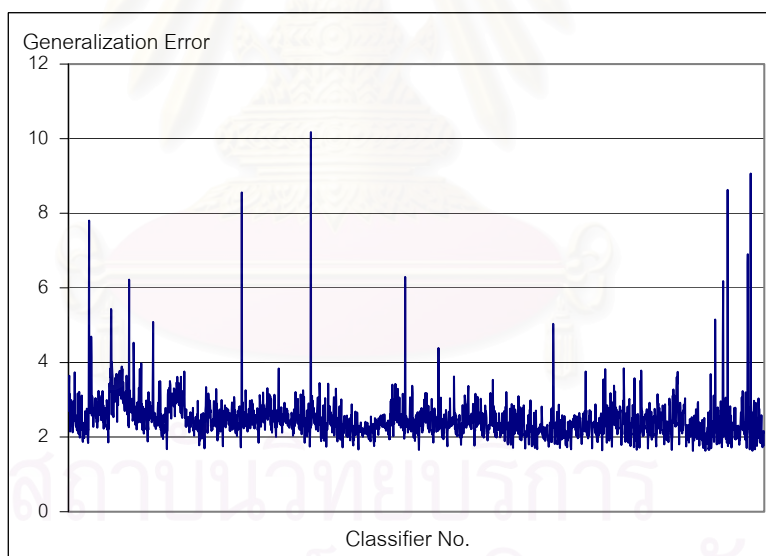
Figure 4.1: The trends of the accuracy of classification when the number of classes is varying.

Figure 4.1 shows the trends of the accuracy of classification of the DDAG, the ADAG and the RADAG when the number of classes is varying. The experiment is based on the English letter image recognition which has 26 classes. The dataset is trained by using Polynomial kernel degree 4. For the DDAG, the number of evaluations for the correct class is unnecessary high. This results in higher cumulative error and lower the accuracy. Using the reversed triangular structure, the ADAG reduces the number of times the correct class is tested against other classes, and thus reduces the cumulative errors. By using the minimum-weighted perfect matching, the RADAG can select the appropriate sequence of binary classifiers which have small generalization errors to be used in data classification. This greatly improves the

performance of the original ADAG, especially when the number of classes is relatively large.



(a) The generalization errors of classifiers of the letter dataset.



(b) The generalization errors of classifiers of the Thai printed character dataset.

Figure 4.2 The variation of the generalization errors.

In addition, the performance of the RADAG is better when the variance of the generalization errors is high. In that case, when using the reordering algorithm the RADAG can select the best sequence whose generalization error much lower than that of the arbitrary sequence of the DDAG and the ADAG. Figure 4.2 shows the variation of

the generalization errors of two datasets, the letter and the Thai printed character. The variances of the generalization errors of the letter dataset and the Thai printed character dataset are 643.13 and 0.30, respectively. In this case, the variance of the generalization errors of the letter dataset is higher, and thus the performance of the RADAG when using with the letter dataset is better (see Table 4.18).



สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

## CHAPTER V

### PERFORMANCE ENHANCEMENT

The experimental results from the previous chapter show that the RADAG requires much less computational time, especially when the number of classes and/or the number of dimensions are large. So we have much more time to do more things to improve the accuracy. In this chapter we will discuss how to improve the performance of the RADAG. We also suggest an alternative way to increase the accuracy of the DDAG method.

#### 5.1 Enhancing Performance of the DDAG

In this section, we suggest an improved algorithm of the DDAG, called Selected DDAG (SDDAG). Figure 3.2 shows that the generalization errors of all classifiers are varying. Among  $k(k-1)/2$  classifiers, only  $k-1$  classifiers are used in data classification. The binary classifiers in nodes in the DDAG structure should dynamically change during classification process rather than being fixed in advance. The classifiers with the least generalization error will be considered to be in nodes in each level.

The SDDAG algorithm is illustrated in Figure 5.1.  $k$  is the number of classes. *Set-of-the-discarded-classes* keeps classes which were discarded after classifying the test data.

1. Sort all  $k(k-1)/2$  classifiers by ascending the generalization error.
2. Initialize *set-of-the-discarded-classes* to be empty.
3. For  $i = 1$  to  $k-1$  do;
  - 3.1 Select a classifier with the least generalization error, of which the corresponding two classes are not in the *set-of-the-discarded-classes*;
  - 3.2 Classify the test data;
  - 3.3 Put the discarded class into the *set-of-the-discarded-classes*.
4. Print the remaining class as the output class.

Figure 5.1: The SDDAG algorithm.



To evaluate the SDDAG algorithm, we examined the SDDAG with the isolated English letter speech recognition dataset. Table 5.1 shows the comparison of the accuracy of classification between the DDAG and the improved algorithm, SDDAG. The highest accuracy of each parameter of the kernel is depicted in bold-face. The results show that the SDDAG gives higher accuracy in both polynomial and RBF kernels.

Table 5.1: A comparison of the accuracy of classification of the DDAG and the SDDAG.

Polynomial kernel			RBF kernel		
d	DDAG	SDDAG	c	DDAG	SDDAG
2	96.556	<b>96.600</b>	0.001	96.509	<b>96.729</b>
3	97.032	<b>97.049</b>	0.002	96.594	<b>96.729</b>
4	96.985	<b>97.049</b>	0.003	96.900	<b>96.985</b>
5	96.631	<b>96.665</b>	0.004	96.753	<b>96.857</b>
6	96.672	<b>96.729</b>	0.005	96.710	<b>96.857</b>
7	96.133	<b>96.151</b>	0.01	96.939	<b>96.985</b>
8	95.492	<b>95.574</b>	0.03	96.683	<b>96.702</b>

## 5.2 Enhancing Performance of the RADAG

In this section we reveal a problem that may occur when using the RADAG. Then we introduce enhancement versions to overcome the problem.

### 5.2.1 The Problem

In some dataset, there are several similar classes. The classifiers which are trained to separate these classes may have high generalization errors. For example, for the English letter image recognition dataset, classes 'B', 'H', 'K' and 'R' are very similar. Consequently the binary classifiers of these classes have high generalization error.

The structure of the RADAG may cause the classifiers with high generalization errors in low level. The method continually discards the incorrect classes. As the result, there are more similar classes in the lower level. Especially in the last level, there is no choice to select the best classifier. The last classifier comes from the output

of two nodes in the previous level. It may have high generalization error so it may cause wrong classification.

First we want to compare the wrong classification of the ADAG and the RADAG, and thus we use the reordering algorithm with minimum-weight perfect matching to choose the optimal sequence to be the initial sequence for both methods. The experiment is based on the English letter image recognition. The dataset is trained by using the Polynomial kernel of degree 4.

Table 5.2 shows the number of misclassified examples after evaluating 4,037 test examples. The result shows that the RADAG causes wrong classification lower than the ADAG in higher levels (e.g. 1,2), but it causes mistakes higher than the ADAG in lower levels (e.g. 4,5). From the table, the RADAG causes 10 misclassified examples less than the ADAG. Actually, the RADAG can reduce 27 misclassified examples of the ADAG, but it introduces 17 new misclassified examples. Table 5.3 illustrates that almost all of the new mistakes occur in low levels, especially in the last level.

Table 5.2: A comparison of the number of misclassified examples of the ADAG and the RADAG.

Level	#class (#classifier)	#misclassify	
		ADAG	RADAG
1	26 (13)	4	4
2	13 (6)	30	10
3	7 (3)	17	15
4	4 (2)	47	54
5	2 (1)	69	74
Total		167	157

Table 5.3: The number of new misclassified examples introduced by the RADAG.

Level	#new misclassify
1	0
2	1
3	1
4	7
5	8
Total	17

### 5.2.2 Enhancement Versions

According to the problem of the RADAG discussed in the previous subsection, we proposed five alternative methods to overcome this problem.

*Method 1* (See figure 5.2): First, we try the RADAG in the higher levels until there are only similar classes. Then we use Max Wins to predict the correct class. The output class is the class with the most votes.

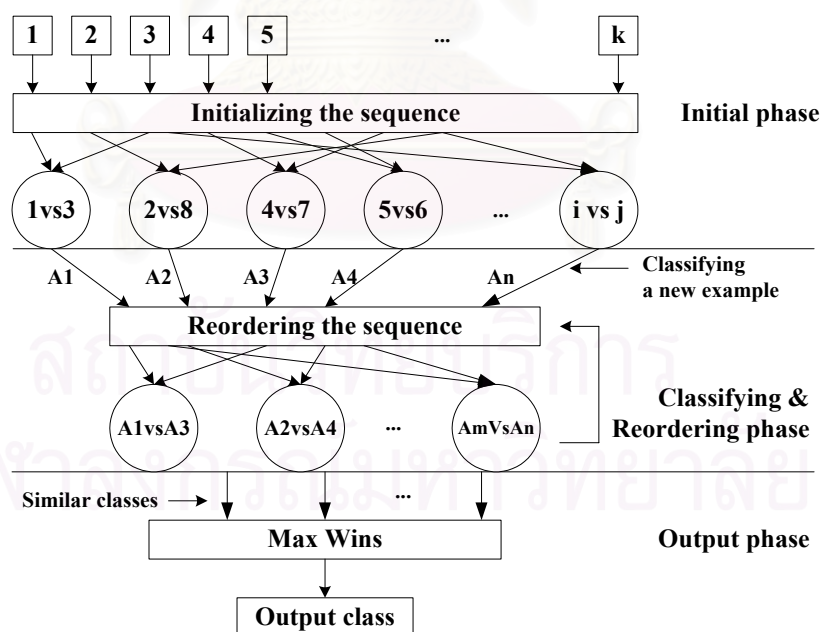


Figure 5.2: Classifying process of the RADAG &amp; Method1.

*Method 2* (See figure 5.3): First, we try the RADAG in the higher levels until there are only two or more similar classes. Then we vote by using

only SVMs constructed from one of these classes and the other classes. For example, for a 5-class problem, assuming that there are class 2 and class 5 in the last level. Then we vote by using 2vs1, 2vs3, 2vs4, 2vs5, 5vs1, 5vs3 and 5vs4. The predicted class is the class with the most votes.

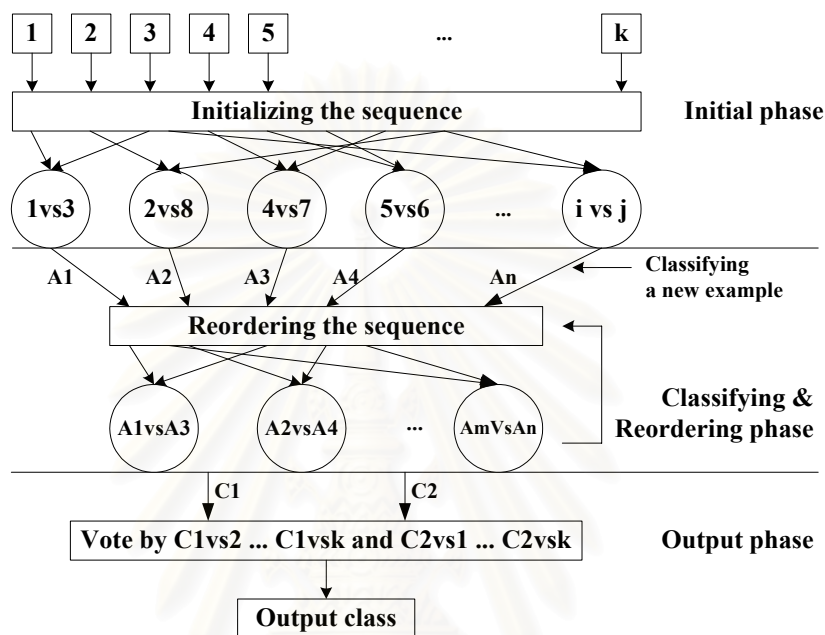


Figure 5.3: Classifying process of the RADAG & Method2.

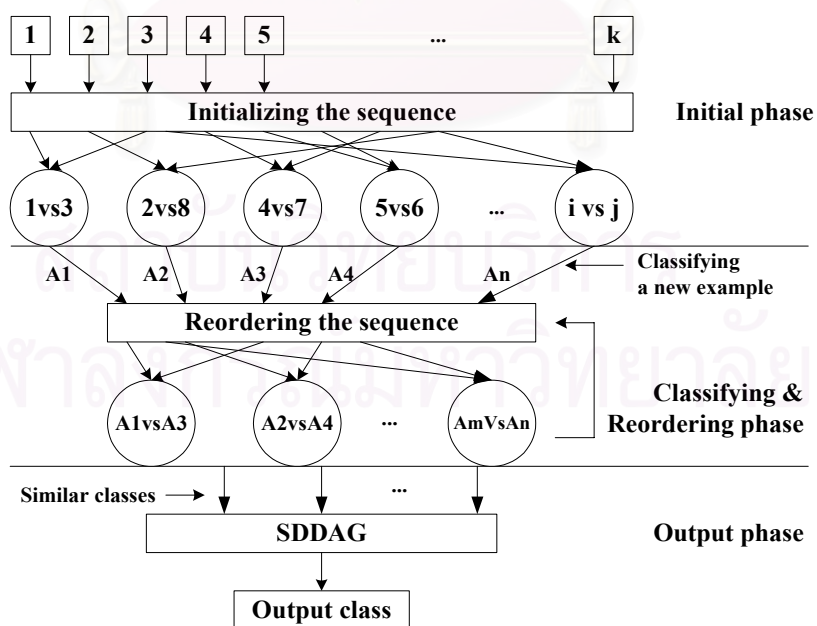


Figure 5.4: Classifying process of the RADAG & Method3.

*Method 3* (See figure 5.4): First, we try the RADAG in the higher levels until there are only similar classes. Then we use SDDAG to eliminate the incorrect class. The output class is the remaining class.

*Method 4* (See figure 5.5): Sort all classifiers by ascending the generalization error. Then we collect a set of classifiers with small generalization errors. Make sure that the set covers all classes. For example, we collect a set of classifiers where each class is combined with at least three other classes (see Figure 5.5(b)). These three classifiers have small generalization error. We then run the RADAG by using only the collected classifiers. These steps are used in every level, except for the last level. Figure 5.5(c) is an example of the output of the RADAG. It consists of 4 binary classifiers, which are used in classification process.

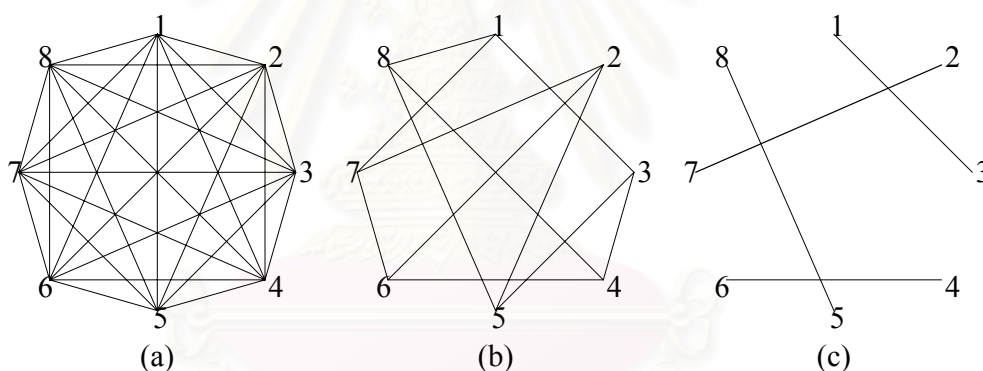


Figure 5.5: Classifying process of the RADAG & Method4.

(a) A graph for an 8-class problem.

(b) A graph of collected classifiers which have small generalization errors.

(c) An example of the output of the reordering algorithm when using collected classifiers.

*Method 5* (See figure 5.6): First, we try the RADAG in the higher levels until there are only two remaining classes. Then we vote by using the classifiers of the two classes, which are trained by using various parameters of kernels. The output class is the class with the most votes.

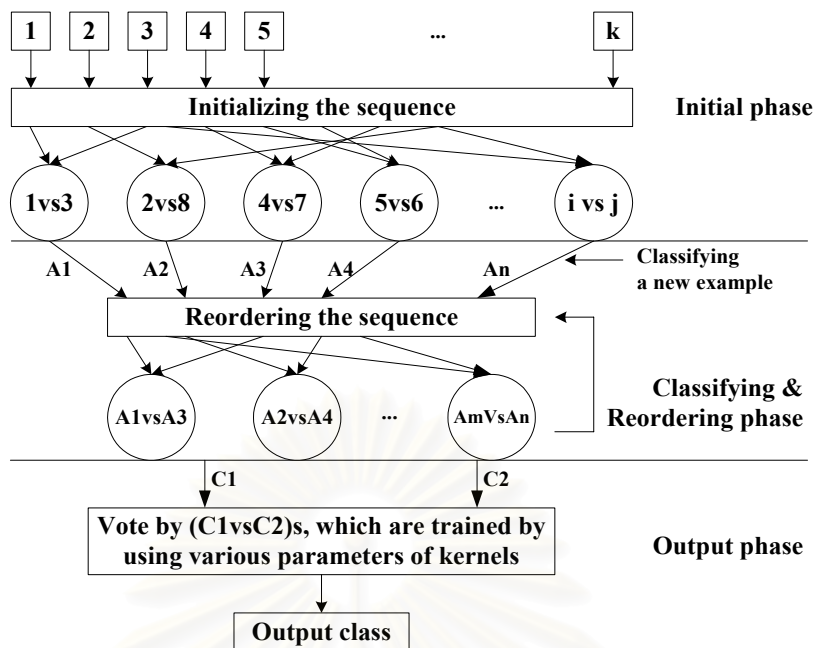


Figure 5.6: Classifying process of the RADAG & Method5.

### 5.2.3 Results

Methods 3 and 4 take the running time for classification which equal to that of the RADAG. In case of small number of classes, methods 1, 2 and 5 may require more running time for classification than Max Wins. But it is negligible time if the number of classes is large.

Tables 5.4 and 5.5 present the results of the comparison between the RADAG and these methods for Polynomial and RBF kernels, respectively. We present the optimal parameters ( $d$  and  $c$  in Equations (18) and (19)) of the kernels and the corresponding accuracies. For method 5, the  $d$  or  $c$  means the parameter of the kernel of the classifiers which are used in all levels excluding the last level. The best accuracy among these methods is illustrated in bold-face. The results show that method 1 and method 5 may be more suitable for practical use. Most of the results of method 4 are comparable to the results of the RADAG. Most of the results of method 2 are less than the results of the RADAG.

For the polynomial kernel, Method 3 gives accuracy equal to the maximum accuracy of the ADAG in the satimage dataset (compared with Table 4.14). Method 4 provides accuracy equal to the maximum accuracy of the ADAG in the

satimage, soybean and Thai printed character 2 datasets. Method 5 gives accuracy higher than the maximum accuracy of the ADAG in the satimage, letter and and Thai printed character 2 datasets.

For the RBF kernel, Method 1 provides accuracy higher than the maximum accuracy of the ADAG in the soybean dataset (compared with Table 4.15). Method 2 gives accuracy equal to the maximum accuracy of the ADAG in the soybean and Thai printed character 2 datasets. Method 3 provides accuracy equal to the maximum accuracy of the ADAG in the Thai printed character 2 dataset. Method 4 gives accuracy equal to the maximum accuracy of the ADAG in the soybean dataset. Method 5 provides accuracy higher than the maximum accuracy of the ADAG in the segment dataset and gives accuracy equal to the maximum accuracy of the ADAG in the isolet datasets.



สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

Table 5.4: A comparison of the accuracy of classification of the RADAG and the Enhancing methods using the Polynomial kernel

Dataset	RADAG		RADAG & Method1		RADAG & Method2		RADAG & Method3		RADAG & Method4		RADAG & Method5	
	d	Accuracy	d	Accuracy	d	Accuracy	d	Accuracy	d	Accuracy	d	Accuracy
Glass	2	71.063	2	71.517	2	71.052	2	71.063	2	<b>71.528</b>	8	71.041
Satimage	6	88.900	6	88.550	6	88.450	6	88.900	6	88.900	2	<b>90.000</b>
Segment	8	97.489	8	97.402	8	97.359	8	97.576	8	97.489	7	<b>97.749</b>
Shuttle	8	<b>99.924</b>	8	<b>99.924</b>	8	<b>99.924</b>	8	<b>99.924</b>	8	<b>99.924</b>	2	99.890
Vowel	3	64.502	2	64.286	3	63.853	3	64.935	3	64.502	2	<b>65.152</b>
Soybean	3	<b>91.176</b>	5	90.882	3	90.588	3	90.882	3	<b>91.176</b>	3	90.588
Letter	4	96.111	3	96.185	4	96.086	3	96.260	4	96.012	3	<b>96.730</b>
Isolet	3	<b>97.049</b>	3	<b>97.049</b>	3	<b>97.049</b>	3	<b>97.049</b>	3	<b>97.049</b>	3	96.985
ThaiPrintedCharacter1	2	93.444	2	93.382	2	93.076	2	93.382	2	93.444	2	<b>93.627</b>
ThaiPrintedCharacter2	2	99.112	2	99.020	2	98.958	2	99.081	2	99.112	2	<b>99.142</b>



Table 5.5: A comparison of the accuracy of classification of the RADAG and the Enhancing methods using the RBF kernel

Dataset	c	RADAG	c	RADAG & Method1	c	RADAG & Method2	c	RADAG & Method3	c	RADAG & Method4	c	RADAG & Method5
Glass	0.09	<b>74.319</b>	0.09	73.843	0.09	72.912	0.09	<b>74.319</b>	0.09	73.843	0.09	71.982
Satimage	3.0	91.950	3.0	<b>92.000</b>	3.0	<b>92.000</b>	3.0	91.950	3.0	91.950	3.0	90.750
Segment	0.7	97.273	0.7	97.403	0.7	97.403	0.8	97.230	0.8	97.273	0.7	<b>97.576</b>
Shuttle	3.0	<b>99.897</b>	3.0	<b>99.897</b>	3.0	<b>99.897</b>	3.0	<b>99.897</b>	3.0	<b>99.897</b>	1.0	<b>99.897</b>
Vowel	0.2	67.100	0.3	66.017	0.2	64.069	0.2	67.965	0.2	67.100	0.2	<b>68.182</b>
Soybean	0.07	90.882	0.07	<b>91.176</b>	0.07	90.882	0.07	90.588	0.07	90.882	0.07	90.588
Letter	3.0	<b>97.969</b>	3.0	97.944	3.0	97.919	3.0	97.919	3.0	97.944	3.0	97.795
Isolet	0.003	<b>97.115</b>	0.003	96.985	0.003	96.985	0.003	96.985	0.01	96.985	0.003	<b>97.115</b>
ThaiPrintedCharacter1	0.003	97.028	0.003	<b>97.120</b>	0.003	97.089	0.003	97.028	0.001	96.140	0.0007	96.691
ThaiPrintedCharacter2	0.004	<b>99.418</b>	0.004	99.387	0.004	<b>99.418</b>	0.004	<b>99.418</b>	0.0009	98.192	0.002	99.326

## CHAPTER VI

### SUMMARY AND FUTURE WORKS

In this chapter, we conclude our research work and present some directions for the future work.

#### 6.1 Summary

In this thesis, we have presented a new approach for multiclass SVMs, called Reordering Adaptive Directed Acyclic Graph (RADAG), which is the modification of the original ADAG. Our approach eliminates the dependency of the sequence of binary classifiers in nodes in the original ADAG by selecting an appropriate sequence

from all possible  $\frac{k!}{2^{\lfloor \frac{k}{2} \rfloor} \lfloor \frac{k}{2} \rfloor!}$  sequences, where  $k$  is the number of classes, which consists

of classifiers with small generalization error. By the use of minimum-weight perfect matching, only binary classifiers which have small generalization errors will be used in data classification and the RADAG can reorder the sequence of binary classifiers in polynomial time.

Assume that the distribution of generalization errors is normal distribution. The generalization errors of classifiers which are selected by the reordering algorithm with minimum-weight perfect matching are statistically significantly lower than those randomly selected by the ADAG at the 1% significant level. Besides, the empirical results show that the minimum-weight perfect matching always selects classifiers with generalization errors less than half of maximum generalization error.

The experimental results show that our new approach yields higher accuracy than the DDAG, the original ADAG and even Max Wins which is probably the currently most accurate method for multiclass SVMs. The RADAG always provides one best accuracy for each dataset using the reordering algorithm, whereas, depending on the sequence of classes, the DDAG and the ADAG may give low accuracies. Moreover, the running time used by the RADAG is much less than Max Wins, especially when the

number of classes and/or the number of dimensions are relatively large. So our approach is suitable for large-scale problems.

In this thesis, we also presented alternative methods to enhance the performance of the RADAG. Although some of them require more computational time than the RADAG, they can raise the accuracies. In addition, we proposed a method to improve the performance of the DDAG.

## 6.2 Future Works

Reducing the number of comparison of binary classifiers is still an open issue, which needs a through investigation. A hierarchical SVMs for multiclass classification using the binary tree structure requires only  $O(\log_2 n)$  applications of binary classifiers for the problem with  $n$  classes. But a problem is how to determine the binary classifiers in nodes in the binary tree.

One can replace the SVM classifiers in nodes in the RADAG structure with other kinds of binary classifiers, to implement a new multiclass classifier.

To increase the accuracy of the RADAG, it is worthwhile to investigate a good method to tackle the problem when the classes in the lower levels are very similar. In addition, one may choose an effective algorithm to be used with reordering algorithm to select an optimal sequence of classifiers. Beside the bound on the generalization error used in this research, one may consider other factors that affect the accuracy of classification. Another measure that may place a bound on the generalization error of a consistent hyperplane with high confidence is the number of support vectors.

The knowledge learned by the RADAG (or other multiclass SVMs such as DDAG and ADAG) is generally difficult to understand by humans compared to the decision tree which easily generates symbolic rules. The provision of a mechanism that can interpret the structure of the RADAG in the form of rules would be very useful. Knowledge extraction by forming symbolic rules from the internal nodes of the RADAG may be more effective when using with mission critical applications such as those engaged in aerospace, military, and medical systems.

## REFERENCES

- Abe, S. and Inoue, T. Fuzzy Support Vector Machines for Multiclass Problems.  
The European Symposium on Artificial Neural Networks (ESANN-2002),  
2002: 113-118.
- Allwein, E., Schapire, R., Singer, Y. Reducing Multiclass to Binary: A Unifying Approach  
for Margin Classifiers. Journal of Machine Learning Research, vol. 1, September,  
2001.
- Ambwani, T. Multi class Support Vector Machine implementation to intrusion detection.  
International Joint Conference on Neural Networks, Vol. 3, 2003: 2300-2305.
- Bartlett, P. L. and Shawe-Taylor, J. Generalization performance of Support Vector  
Machines and other pattern classifiers. in Advances in Kernel Methods - Support  
Vector Learning, USA: MIT Press, Cambridge, 1999: 43-54.
- Blake, C., Keogh, E., and Merz, C. UCI Repository of Machine Learning Databases.  
Department of Information and Computer Science, University of California, Irvine,  
1998. Available from: <http://www.ics.uci.edu/~mlearn/MLRepository.html>
- Cristianini N. and Shawe-Taylor, J. An Introduction to Support Vector Machines and  
other kernel-based learning methods. Cambridge University Press, 2000.
- Cook, W. and Rohe, A. Computing Minimum-weight Perfect Matchings. Technical Report  
97863, Forschungsinstitut für Diskrete Mathematik, Universität Bonn, 1997.
- Crammer, K. and Singer, Y. On the Algorithmic Implementation of Multiclass Kernel-  
based Vector Machines. Journal of Machine Learning Research, vol. 2,  
December, 2001: 265-292.

- Dong, X., Zhaohui, W. and Yunhe P. A New Multi-class Support Vector Machines. IEEE International Conference on System, Man, and Cybernatics, vol. 3, 2001: 1673-1676.
- Franc, V. and Hlavac, V. Multi-class Support Vector Machine. The 16<sup>th</sup> International Conference on Pattern Recognition. Vol. 2, 2002: 236-239.
- Friedman, J. Another Approach to Polychotomous Classification. Technical report, Department of Statistics, Stanford University, 1996.
- Fürnkranz, J. Round robin classification. Journal of Machine Learning Research, vol. 2, March, 2002.
- Godbole, S., Sarawagi, S. and Chakrabarti, S. Scaling multi-class Support Vector Machines using inter-class confusion. The 8<sup>th</sup> ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.
- Goh, K., Chang, E. and Cheng, K. SVM binary classifier ensembles for image classification. The 10<sup>th</sup> International Conference on Information and Knowledge Management, 2001.
- Goh, K., Chang, E. and Cheng, K. Support Vector Machines pairwise classifiers with error reduction for image classification. The 2001 ACM Workshops on Multimedia: multimedia information retrieval, 2001.
- Hsu, C. and Lin, C. A Comparison of Methods for Multiclass Support Vector Machines. IEEE Transactions on Neural Networks, vol. 13, March, 2002: 415-425.
- Joachims, T. Making large-scale SVM learning practical. Advances in Kernel Methods – Support Vector Learning, MIT Press, 1999.

- Kindermann, J., Leopold, E. and Paass, G. Multi-class Classification with Error Correcting Codes. Treffen der GI-Fachgruppe 1.1.3, Maschinelles Lernen, GMD Re. 114, 2000.
- Klautau, A., Jevtić N. and Orlitsky, A. On nearest-neighbor error-correcting output codes with application to all-pairs multiclass support vector machines. Journal of Machine Learning Research, vol. 4, December, 2003.
- Li, K., Huang, H. and Tian, S. A Novel Multi-class SVM Classifier based on DDAG. The 1<sup>st</sup> International Conference on Machine Learning and Cybernetics, 2002: 1203-1207.
- Liu, X., Xing, H. and Wang, X. A Multistage Support Vector Machine. International Conference on Machine Learning and Cybernetics, Vol. 2, 2003: 1305-1308.
- Mayoraz, E., Alpaydm, E. Support Vector Machines for Multi-class Classification. IDIAP-RR 6, IDIAP, 1998.
- Mitchell, T. Machine Learning, McGraw Hill, 1997.
- Nemhauser, G. and Wolsey, L. Integer and Combinatorial Optimization. New York: Wiley, 1999.
- Pichitdej, S. Thai Printed Character Recognition Using a Neural Network Ensemble. Degree of Master Computer Engineering in Department of Computer Engineering Faculty of Engineering Chulalongkorn University, 2001.
- Platt, J., Cristianini, N. and Shawe-Taylor, J. Large Margin DAGs for Multiclass Classification. Advances in Neural Information Processing Systems, MIT Press, vol. 12, 2000: 547-553.

Rifkin, R. and Klautau, A. In Defense of One-Vs-All Classification. Journal of Machine Learning Research, vol. 5, June, 2004.

Roth, V. and Tsuda, K. Pairwise Coupling for Machine Recognition of Hand-Printed Japanese Characters. IEEE International Conference on Computer Society, vol. 1, 2001: 1120-1125.

Schwenker, F. Hierarchical Support Vector Machines for Multi-class Pattern Recognition. The 4<sup>th</sup> International Conference on knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000: 561-565.

Sekhar, C., Takeda, K. and Itakura, F. Close-class-set Discrimination Method for Large-class-set Pattern Recognition using Support Vector Machines. IEEE/INNS' International Joint Conference on Neural Networks (IJCNN-2002), vol. 1, 2002: 577-582.

Takahashi, F. and Abe, S. Decision-Tree-Based Multiclass Support Vector Machines. The 9<sup>th</sup> International Conference on Neural Information Processing (ICONIP-2002), vol.3, 2002: 1418-1422.

Thubthong, N. and Kijsirikul, B. Support Vector Machines for Thai Phoneme Recognition. International Conference on Intelligent Technologies, 2000: 229-234.

Ussivakul, N. and Kijsirikul, B. Adaptive DAG: Another Approach for Multiclass Classification. International Conference on Intelligent Technologies, 2001.

Ussivakul, N. and Kijsirikul, B. Multiclass Support Vector Machines using Adaptive Directed Acyclic Graph. IEEE/INNS' International Joint Conference on Neural Networks (IJCNN-2002), 2002.

Vapnik, V. Statistical Learning Theory. New York: Wiley, 1998.

Vapnik, V. An Overview of Statistical Learning Theory. IEEE Transactions on Neural Networks, vol. 10, September, 1999: 988-999.

Weston, J. and Watkins, C. Multi-Class Support Vector Machines. Technical Report CSD-TR-98-04, Department of Computer science, Royal Holloway, University of London, May, 1998.

Yu, H., Han, J. and Chang, K. PEBL: positive example based learning for Web page classification using SVM. The 8<sup>th</sup> ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.



สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย





APPENDICES

สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย

## APPENDIX A

### PUBLICATIONS

#### A.1 International Journal

1. Phetkaew, T., Rivepiboon W., and Kijirikul, B. Reordering Adaptive Directed Acyclic Graphs for Multiclass Support Vector Machines. Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.7, No. 3, 2003: 315-321.

#### A.2 International Conference

1. Phetkaew, T., Kijirikul, B., and Rivepiboon, W. Reordering Adaptive Directed Acyclic Graphs: An Improved Algorithm for Multiclass Support Vector Machines. The proceedings of the INNS/IEEE International Joint Conference on Neural Network (IJCNN-2003), 2003: 1605-1610.
2. Phetkaew, T., Kijirikul, B., and Rivepiboon, W. Multiclass Classification of Support Vector Machines by Reordering Adaptive Directed Acyclic Graph. The proceedings of the SANKEN International Workshop on Intelligent Systems (SIWIS-2003), 2003.
3. Phetkaew, T., Kijirikul, B., and Rivepiboon, W. Reordering Adaptive Directed Acyclic Graphs for Multiclass Support Vector Machines. The proceedings of the 3<sup>rd</sup> International joint conference on Intelligence Technologies and the 3rd Vietnam-Japan Symposium on Fuzzy Systems and Applications (InTech/VJFuzzy-2002), 2002: 276-284.

#### A.3 National Conference

1. Phetkaew, T., Rivepiboon, W., and Kijirikul, B. Learning Multiclass Support Vector Machines by Reordering Adaptive Directed Acyclic Graph. The proceedings of the 7<sup>th</sup> National Computer Science and Engineering Conference (NCSEC-2003), 2003: 450-455.

## BIOGRAPHY

**Name** Thimaporn Phetkaew  
**Sex** Female  
**Marital Status** Married  
**Date of Birth** July 3, 1975  
**Place of Birth** Surat Thani  
**Permanent Address** 110 Moo. 3, Khaowong, Bantakhun, Surat Thani, 84230

### Education:

2004 **Ph.D.** in Computer Engineering, Chulalongkorn University  
*Funding source:*  
- The Thailand Research Fund (the Royal Golden Jubilee Ph.D. Program)  
- Commission on Higher Education Ministry of Education

2000 **M.Sc.** in Computer Science, Prince of Songkla University  
*Funding source:*  
- National Science and Technology Development Agency

1997 **B.Sc.** in Applied Mathematics, Prince of Songkla University



สถาบันวิทยบริการ  
จุฬาลงกรณ์มหาวิทยาลัย