THE HONG KONG POLYTECHNIC UNIVERSITY
Department of Computing

Localized Generalization Error Bound for Multiple Classifier System

Patrick, Pak Kei CHAN

A thesis submitted in partial fulfillment of requirements
for the degree of Doctor of Philosophy

June 2009

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____

Patrick, Pak Kei CHAN

ABSTRACT

The objective of this thesis is to study the Localized Generalization Error Model (L-GEM) for Multiple Classifier Systems (MCSs). L-GEM for a single classifier was proposed by Yeung (2007). It is based on the observation that it will not be reasonable to expect a classifier which is trained by using a set of learning samples to recognize unseen samples very different from the training set. The L-GEM provides an upper bound for the mean square error (MSE) of unseen samples in a neighborhood of each training sample.

One significant application of the L-GEM is that it could be used as an objective function for base classifier training. The assumption of the same width for all dimensions of a hidden neuron in the initial version of the L-GEM is now relaxed. The parameters of a RBF network are selected by minimizing its localized generalization error bound. The characteristics of the proposed objective function are compared with those for regularization methods. For the problem of weight selection, RBF networks trained by minimizing the proposed objective function consistently outperform RBF networks trained by techniques such as Training Error Minimization, Tikhonov Regularization, Weight Decay or Locality Regularization. The proposed objective function is equally effective in the selection of three parameters simultaneously: center, width and weight. RBF networks trained by minimizing the proposed objective function yield better testing accuracies when compared to those which minimize training error only.

A new dynamic fusion method for the construction of a MCS based on L-GEM is also proposed. This L-GEM based Fusion method (LFM) uses the L-GEM to estimate the local competence of the base classifiers in a MCS. Different from the current dynamic

classifier selection methods, the LFM estimates the performance of the base classifiers not only using the training samples but also points in local neighborhood regions. Experimental results show that a MCS using LFM has a good performance consistently in terms of testing classification accuracy and time complexity. The LFM is also compared with twenty one current dynamic fusion methods experimentally. The results show that the LFM yields better testing accuracies than other dynamic fusion methods.

L-GEM has been extended from a single classifier system to a MCS, named L-GEM^{MCS}. L-GEM^{MCS} is closely related to the existing error model "Bias Ambiguity Decomposition". L-GEM^{MCS} consists of four terms: base classifier training error, diversity of training error, base classifier sensitivity and diversity of sensitivity. The two terms, diversity of training error and diversity of sensitivity, are new concepts which could be used to characterize the interactions among the base classifiers in MCS. The meaning and relationship of these four terms are analyzed and discussed. L-GEM^{MCS} is shown to be useful in evaluating the generalization ability of a MCS. It can be used as a selection criterion for the best set of classifiers for the construction of a MCS from a pool of diverse base classifiers.

LIST OF PUBLICATIONS

Submitted Paper

- **Chan, P.P.K.**, Yeung, D.S., Liu, J.N.K., Lin, M.L. (2009). Dynamic Fusion Using Localized Generalization Error Model, Submitted to *IEEE Trans. on Pattern Analysis and Machine Intelligence*

Accepted Papers

- Chan, A.P.F., **Chan, P.P.K.**, Ng, W.W.Y., Tsang, E.C.C. and Yeung, D.S. (2008). A novel feature grouping method for ensemble neural network using localized Generalization Error Model. *Pattern Recognition and Artificial Intelligence, vol. 22(1), pp. 137-151.*

- **Chan, P.P.K.**, Chan, A.P.F., Tsang, E.C.C. and Yeung, D.S. (2006). Experimental Comparison Between Implicit and Explicit MCSs Construction Methods. In *Proc. 2006 Int'l Conf. on Machine Learning and Cybernetics, pp. 2218-2221.*

- **Chan, P.P.K.**, Wang, D.F., Tsang, E.C.C. and Yeung, D.S. (2006). Structured Large Margin Machine Ensemble. In *Proc. 2006 Int'l Conf. on Systems, Man and Cybernetics, vol. 1, pp. 840-844.*

- **Chan, P.P.K.**, Ng, W.W.Y. and Yeung, D.S. (2005). Active learning using localized generalization error of candidate sample as criterion. In *Proc. 2005 Int'l Conf. on Systems, Man and Cybernetics, vol. 4, pp. 3604-3609.*

- **Chan, P.P.K.**, Zeng, X.Q., Tsang, E.C.C., Yeung, D.S. and Lee, J.W.T. (2007). Neural network ensemble pruning using sensitivity measure in web applications. In *Proc. 2007 Int'l Conf. on Systems, Man and Cybernetics, pp. 3051-3056.*

- Tsang, E.C.C., Yeung, D.S. and **Chan, P.P.K.** (2003). Fuzzy support vector machines for solving two-class problems, In *Proc. 2003 Int'l Conf. on Machine Learning and Cybernetics, vol. 2, pp. 1080-1083.*

- Tsang, G.C.Y., **Chan, P.P.K.**, Yeung, D.S. and Tsang, E.C.C. (2004). Denial of service detection by support vector machines and radial-basis function neural network. In *Proc. 2004 Int'l Conf. on Machine Learning and Cybernetics, vol. 7, pp. 4263-4268.*

- Wang, D.F., **Chan, P.P.K.**, Yeung, D.S. and Tsang, E.C.C. (2004). Feature subset selection for support vector machines through sensitivity analysis. In *Proc. 2004 Int'l Conf. on Machine Learning and Cybernetics, vol. 7, pp. 4257-4262.*

- Yeung, D.S. and **Chan, P.P.K.** (2009). A Novel Dynamic Fusion Method Using Localized Generalization Error Model, To Appear in *Proc. 2009 Int'l Conf. on Systems, Man, and Cybernetics.*

- Yeung, D.S. and **Chan, P.P.K.** (2009). A Novel Learning Objective Function Using Localized Generalization Error Bound for RBF Network, In *Proc. 2009 Int'l Conf. on Machine Learning and Cybernetics, vol. 2, pp. 936-942.*

- Yeung, D.S., **Chan, P.P.K.** and Ng, W.W.Y. (2009), Radial Basis Function Network Learning using Localized Generalization Error Bound. *Information Science, vol. 179 (19), 3199-3217.*

- Yeung, D.S., Ng, W.W.Y., Chan, A.P.F., **Chan, P.P.K.**, Firth, M. and Tsang, E.C.C. (2007). Bankruptcy Prediction Using Multiple Intelligent Agent System via a Localized Generalization Error Approach. In *Proc. 2007 Int'l Conf. on Service Systems and Service Management, pp. 1-6.*

ACKNOWLEDGEMENTS

It has been a long, tough but exciting and rewarding life journey.

During this journey I encountered many wonderful people. Some offered me their helping hands. Some delivered words of encourgament when I was most discouraged. Some became my trusted friends. But every single memory of this journey adds blessing to my life.

If I must name the most treasures lesson learned during my Ph.D. study, I would say, "*I learn to be humble.*"

First, I would like to express my most sincere appreciation to my original chief supervisor and current co-supervisor, **Chair Professor Daniel So Yeung**, for introducing me to the exciting world of research. I will never forget how he encouraged and supported me kindly and patiently when I failed to do my part. He enlightens me not only academically, but also values and goals in life. Prof. Yeung is the greatest teacher I ever met.

My chief supervisor, **Dr. James Nga Kwok Liu**, and co-supervisor, **Dr. Eric C. C. Tsang**, have always supported me with valuable guidance and constructive advice. Without their timely help, my work will be much delayed.

It has been my great fortune to meet many scholars in my Ph.D. journey: **Professor Xizhao Wang**, **Professor Degeng Chen**, **Dr. Ludmila Kuncheva**, **Dr. Xiaoqin Zeng** and **Dr. John W. T. Lee**. They were always willing to impart their valuable knowledge and engaged me in countless inspired discussions.

I have also been benefited from precious interaction with my research teammates: **Dr. Wing Ng**, **Dr. Defeng Wang**, **Mr. Robert P. Woon**, **Miss Aki Po Fong Chan**, **Miss**

Jessica Bin Bin Sun and **Miss Candy Huili Li**. Thank you for all the stimulating debates, challenging questions, mathematics tutorials, English editing and, of course, coffee breaks.

Special thank to all my friends and my teachers: **Miss Foo Lau So, Dr. David Wan**, **Dr. Anthony Yeung**, **Dr. W. S. Chan**, **Miss Ling Yi Kam** and **Mr. K. C. Cheung**. They are more than generous in showing their support and love to me.

This thesis is dedicated to **my Dad**, **Mum**, and **Grandma**. They spoil me with unconditional support, absolute trust and unlimited freedom to enable me to pursue my dream. I can not ask anyone for more than what they have offered me. They deserve the most credit.

Jesus says, "*I am the WAY*".

I am glad that through my Ph.D. journey, I have found this WAY. I never regret it.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND NOTATIONS

MCS	Multiple Classifier System (MCS)
L-GEM	Local Generalization Error Model
L-GEM^{MCS}	Local Generalization Error Model for MCSs
MSE	Mean Square Error

Ω	the input space
D	the training set
x_i	the i^{th} training sample
x_{ir}	the r^{th} feature of the i^{th} training sample
y_i	the class ID of the i^{th} training sample
C	the number of classes
N	the number of samples in training set
n	the number of features of a sample
L	the number of base classifiers in a MCS
f^{RBF}	a RBF Network
f^{MLP}	a MLP Neural Network

f^{mcs}	a MCS
f_i	the i^{th} base classifier in a MCS

CHAPTER 1
INTRODUCTION

1.1 Pattern Classification

Pattern classification is the organization of samples into different categories having similar properties. A number of samples are given as a training set. The objective of pattern classification is to construct a classifier, which learns the knowledge from the given training set, to recognize the unseen samples accurately. For example, we want to filter out spam mails in our email box. A number of spam mails and non-spam mails should be collected and given to the classifier for training purpose. The classifier will learn the difference between spam and non-spam mails from the training set. After training, the classifier can judge if an unseen email is a spam mail or not based on what the classifier has learnt. Obviously, the unseen samples and the training samples should have a certain level of similarity. Otherwise, the knowledge gained from the training set cannot help to identify the unseen samples correctly.

1.2 Localized Generalization Error Model (L-GEM)

A given training set is denoted $D=\{(x_i, y_i)\}_{i=1}^{N}$, where N is the number of training samples. x_i is a n dimension vector denoting the i^{th} training sample, $[x_{i1}, x_{i2},\ldots, x_{in}]^T$, n is the number of features and the superscript T is the vector transpose. y_i represents the true class ID of x_i and $y_i \in \{\omega_c\}_{c=1}^{C}$, where C is the number of classes. The ultimate goal of the

classifier is to recognize unseen samples correctly. The ability can be represented by the generalization error. In this thesis it is defined, by using the mean square error, as:

$$\int_{\Omega} (f(x) - F(x))^2 p_{\Omega}(x) dx \tag{1.1}$$

where Ω denotes the entire input space, $f(x)$ is the trained function, $F(x)$ is the target output and $p_{\Omega}(x)$ denotes the true unknown probability density function of x. As a result, the objective of classification is to create a classifier which has a small value of the loss function defined in (1.1).

However, the exact generalization error of a classifier cannot be computed. Many methods have been proposed to obtain an estimation on it. These methods can be categorized into analytical (e.g. Akaike information criterion (AIC) and VC-dimension), and empirical (e.g. cross-validation and bootstrap). However, the estimated values obtained by analytical methods are usually very loose or restrictive to a certain classifier [Cherkassky et al 1998, 1999, Watanabe 2001]. On the other hand, empirical methods rely on repetitive experiments using different training sets randomly selected from the original one. If the variance of the classifier is large, the resulting performance is usually not good [Hastie et al 2001].

Yeung et al [Ng et al 2007, Yeung et al 2007] proposed a new model called the Localized Generalization Error Model (L-GEM). As there is no information about unseen samples which are very different from the training set, a classifier cannot learn this part of the input space and subsequently the error of the classifier in that region is expected to be high. Therefore, it may be counterproductive to assess the generalization performance of the classifier on unseen samples very different from the training set. Hence it will be more sensible to develop a generalization error model for unseen samples located within a neighborhood of each of the training samples. The localized generalization error bound (R^*_Q)

is an upper bound for the mean square error (MSE) of unseen samples in a Q neighborhood of the training samples.

The Q neighborhood is simply defined as the union of all Q_i neighborhoods of the training sample x_i.

$$Q = \bigcup_{i=1}^{N} Q_i \ , \tag{1.2}$$

where $Q_i = \{x \mid x = x_i + \Delta x; |\Delta x| \le q\}$, i=1..N, N is the number of training samples, $x_i \in D$. When q=0, Q_i={x_i} and Q=D. When $q \to \infty$, $Q \to \Omega$. Therefore, q is the parameter to control the size of the region being considered.

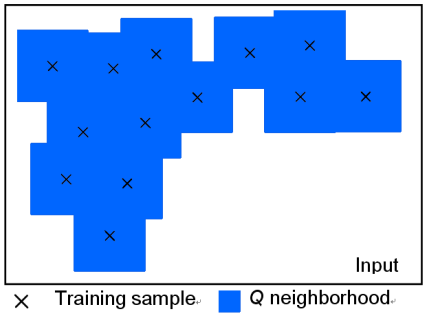


Figure 1.1 Q neighborhood of an artificial training set in the input space

Figure 1.1 illustrates the Q neighborhood of a two-dimensional training set. The localized generalization error in the Q neighborhood is defined as:

$$R_Q = \int_Q (f(x) - F(x))^2 \, p_Q(x) dx \ . \tag{1.3}$$

$p_Q(x)$ denotes the probability density function of x in Q neighborhood. Since of $Q \subseteq \Omega$, the relation between $p_\Omega(x)$ and $p_Q(x)$ is as follows:

$$p_\Omega(x) = c \cdot p_Q(x) \ , \tag{1.4}$$

where the probability of x appearing in Q-neighborhood is s, such as $c = \int_Q p_\Omega(x)dx$ and $x \in Q$.

The derivation of the upper bound for (1.3) is presented as follows:

$$R_Q = \int_Q (f(x) - F(x))^2 \cdot p_Q(x)dx$$

$$\leq \sum_{i=1}^{N} \left(\int_{Q_i} (f(x) - F(x))^2 \cdot p_Q(x)dx \right)$$

$$= \sum_{i=1}^{N} \left(\int_{Q_i} (f(x) - f(x_i) + f(x_i) - F(x_i) + F(x_i) - F(x))^2 \cdot p_Q(x)dx \right). \quad (1.5)$$

Let $\Delta Y_i = f(x) - f(x_i)$, $err_i = f(x_i) - F(x_i)$ and $A_i = F(x_i) - F(x)$. Equation (1.5) can be expressed as:

$$\sum_{i=1}^{N} \left(\int_{Q_i} (f(x) - f(x_i) + f(x_i) - F(x_i) + F(x_i) - F(x))^2 \cdot p_Q(x)dx \right)$$

$$= \sum_{i=1}^{N} \int_{Q_i} (\Delta Y_i + err_i + A_i)^2 \cdot p_Q(x)dx$$

$$= \sum_{i=1}^{N} \int_{Q_i} \begin{pmatrix} (\Delta Y_i)^2 + (err_i)^2 + (A_i)^2 \\ + 2(\Delta Y_i)(err_i) + 2(\Delta Y_i)(A_i) + 2(err_i)(A_i) \end{pmatrix} \cdot p_Q(x)dx$$

$$= \sum_{i=1}^{N} \left(\int_{Q_i} (\Delta Y_i)^2 \cdot p_Q(x)dx \right) + \sum_{i=1}^{N} \left(\int_{Q_i} (err_i)^2 \cdot p_Q(x)dx \right)$$
$$+ \sum_{i=1}^{N} \left(\int_{Q_i} (A_i)^2 \cdot p_Q(x)dx \right) + 2\sum_{i=1}^{N} \left(\int_{Q_i} (\Delta Y_i)(err_i) \cdot p_Q(x)dx \right)$$
$$+ 2\sum_{i=1}^{N} \left(\int_{Q_i} (\Delta Y_i)(A_i) \cdot p_Q(x)dx \right) + 2\sum_{i=1}^{N} \left(\int_{Q_i} (err_i)(A_i) \cdot p_Q(x)dx \right)$$

$$\leq \sum_{i=1}^{N}\left(\int_{Q_i}(\Delta Y_i)^2 \cdot p_Q(x)dx\right)+\sum_{i=1}^{N}\left(\int_{Q_i}(err_i)^2 \cdot p_Q(x)dx\right)+\sum_{i=1}^{N}\left(\int_{Q_i}(A_i)^2 \cdot p_Q(x)dx\right)$$
$$+2\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(\Delta Y_i)^2 \cdot p_Q(x)dx\right)\cdot\sum_{i=1}^{N}\left(\int_{Q_i}(err_i)^2 \cdot p_Q(x)dx\right)}$$
$$+2\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(\Delta Y_i)^2 \cdot p_Q(x)dx\right)\cdot\sum_{i=1}^{N}\left(\int_{Q_i}(A_i)^2 \cdot p_Q(x)dx\right)}$$
$$+2\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(err_i)^2 \cdot p_Q(x)dx\right)\cdot\sum_{i=1}^{N}\left(\int_{Q_i}(A_i)^2 \cdot p_Q(x)dx\right)}$$

$$=\left(\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(err_i)^2 \cdot p_Q(x)dx\right)}+\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(\Delta Y_i)^2 \cdot p_Q(x)dx\right)}+\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}(A_i)^2 \cdot p_Q(x)dx\right)}\right)^2$$

$$=R_Q^* \tag{1.6}$$

The derivation of the error bound (1.3, 1.5-1.6) is slightly different from the one given in [Ng et al 2008, Yeung et al 2007]. Here ε is removed since it is not necessary to be included. Assume that x is uniformly distributed in the Q neighborhood, we have:

$$R_Q^* = \left(\sqrt{R_{emp}}+\sqrt{\underset{D}{E}\underset{Q_i}{E}\left((\Delta Y_i)^2\right)}+A\right)^2 \tag{1.7}$$

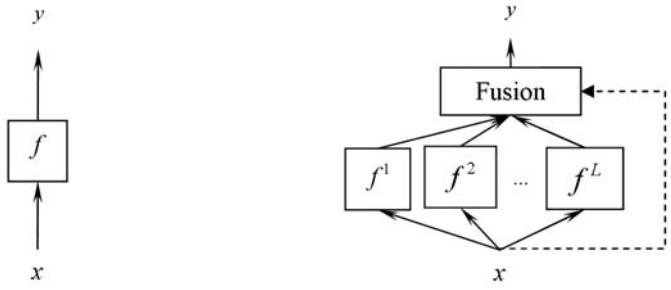
where $R_{emp}=(1/N)\sum_{i=1}^{N}\left(f(x_i)-F(x_i)\right)^2$, $\Delta Y_i = f(x)-f(x_i)$ and A is the square root value of the difference between the maximum and minimum value of the target outputs. The value of A is fixed for a given dataset.

In the L-GEM framework, a classifier with a smaller R^*_Q is preferred. It means that we would like to have a classifier with a small combined value of training error and sensitivity value. If classifier f yields a smaller R^*_Q than classifier g, then f is expected to a better generalization performance than g does. R^*_Q is used as selection criterion in many applications, such as model selection [Ng et al 2007, Yeung et al 2007], feature selection [Ng et al 2008, Yeung et al 2005] and sample selection [Chan et al 2005]. All experimental results have shown consistently good performances.

1.3 Multiple Classifier System (MCS)

Instead of using a single classifier, one may combine the decisions of a number of classifiers to solve a classification problem. The goal of a MCS is to improve the performance of base classifiers by taking advantage of their collective wisdom. MCS is as the same as an ensemble of classifiers. The area of combining classifiers went through parallel routes in different research areas, for example, pattern recognition and data fusion. Therefore, there are various terms for the same notion. Kuncheva [Kuncheva 2001] collected sixteen different aliases, for example, multiple classifier system and ensemble of classifiers. The term MCS is used for the International Workshops on Multiple Classifier Systems [MCS Workshop] held since 2000.

Figure (1.2) shows the architecture for a single classifier and a MCS. f represents a classifier and f^i represents a base classifier in a MCS, where $i = 1..L$ and L is the total number of base classifiers in a MCS. y is the final decision.



a) Single classifiers b) MCSs

Figure 1.2 Architectures of single classifiers and MCSs

For the case of a single classifier (Figure (1.2a)), one classifier makes the final decision. However, in a MCS (Figure (1.2b)), the final decision is made by combining the decisions of a number of base classifiers using a fusion method. Some fusion methods make use of the information of the input x to determine their parameter settings (for instance, the weights). This relationship is represented by a dotted line.

The study of MCSs is motivated by a number of appealing reasons. For example, the concept of a MCS is a very familiar human experience. Many decisions, especially the important ones, are made by a group of people rather than by one person. It would have the advantage of easier acceptance by a majority of the people. Moreover, a complex problem can be broken down into smaller ones by a MCS. This makes the problems easier to be solved and understood. MCSs are also natural extensions of single classifiers.

It is a view shared by many researchers that MCSs can outperform single classifiers. For example, Dietterich suggests three reasons why MCS might be better than a single classifier [Dietterich 2000a]. These three reasons are shown in Figure (1.3). f^* is the optimal classifier for the problem. The outer curve (H) represents the boundary curve for the region of all classifiers. f_1, f_2, f_3 and f_4 are classifiers.

a) Statistical View b) Computational View c) Representational View

Figure 1.3 Three views by Dietterich

Statistical View: A problem arises when the amount of training data available is too small compared to the size of the hypotheses space. Without sufficient data the learning algorithm can find many different hypotheses in H that all give the same accuracy on the training data. If one of these hypotheses is chosen based on a certain criteria, some potentially valuable information may be lost. There is also the risk of choosing the wrong one. Therefore, by constructing a MCS out of all these estimators, the algorithm can average their outputs and may avoid the above problem. Figure (1.3a) illustrates this situation. The

inner curve denotes the set of hypotheses that all give good accuracy on the training data. It is hoped that, by combining the base classifiers, the resultant classifier will be close to f^* than a classifier randomly chosen from the "good classifier space".

Computational View: Many learning algorithms work by performing local search, e.g. back propagation in neural network and greedy splitting rule in decision tree. The local search may get stuck in local optima. This problem occurs even when there is sufficient training data. A MCS constructed by running the local search from many different starting points may provide a better approximation to the true unknown function than any of the individual classifiers, as shown in Figure (1.3b).

Representational View: The complexity of the optimal classifier f^* cannot be known in most situations. As a result, the hypotheses in H cannot represent f^*. By constructing MCSs, it may be possible to expand the space of classifiers. Figure (1.3c) illustrates this situation.

1.4 Problem Statement

MCS has shown to be a successful technique empirically [Bertolini et al 2009, Chen et al 2009, Dietterich 2000b, Ji et al 1997, Tumer et al 2003, Wolpert 1992], but it lacks the support of a theoretical justification for its claim on improved performance over single classifiers. A few models have been proposed to evaluate the generalization error of a MCS [Freund et al 1996, Krogh et al 1995, Tumer et al 1996a, 1996b, Ueda et al 1996]. However, they are mainly attempts to describe the generalization error of a MCS in terms of those for its base classifiers. Moreover, these models are difficult to use since some components in their error estimation model are not computable [Krogh et al 1995, Tumer et al 1996a, Ueda et al 1996]. In other cases the error bound could be very loose [Freund et al 1996].

The major goal of this study is to develop a generalization error model for a MCS with the following characteristics:

- To provide a measure of the generalization capability of the MCS
- To enable a qualitative as well as a quantitative analysis of the relationship between the generalization capability of the MCS and its base classifiers
- To possess ease of computability
- To be applicable to a wide range of problems relevant to the study of MCSs such as dimensionality reduction, base classifier selection, base classifier training and fusion parameter selection

1.5 Contributions of the Thesis

The main contribution of this thesis can be separated into two categories: theoretical contribution and application:

Theoretical Contribution

- A comparison on the motivation, the interpretation and the settings of parameters between the L-GEM and the Regularization method (Chapter 2) for neural network training.
- A new definition of the sensitivity measure for L-GEM (Chapter 3). Computational complexity is reduced and the flexibility increases with this new definition.
- The derivation of the sensitivity measure for the MLP Neural Network and the RBF Network (Chapter 3 and Chapter 4).

- The derivation of the Localized Generalization Error Model for Multiple Classifiers System (L-GEM^{MCS}) (Chapter 4). The properties of the L-GEM^{MCS} are discussed.

- A discussion on the four components of the L-GEM^{MCS}: their meanings and relations (Chapter 4).

- A comparison between the Base Classifier Training Error (Err^{base}) and the Diversity of the Base Classifier Training Errors (Err^{div}) in the L-GEM^{MCS} on one hand, with the weighted error and ambiguity term in the Bias Ambiguity Decomposition method on the other hand (Chapter 4). A direct relationship between Err^{div} and ambiguity was proved.

- A new definition of the sensitivity terms for L-GEM^{MCS} (Chapter 4). The estimation of two sensitivity terms in L-GEM^{MCS} is also proposed.

Application of L-GEM and L-GEM^{MCS}

- Proposing a new training method of the RBF network using the L-GEM as an objective function (Chapter 2). Well known training methods including the regularization method are compared with the L-GEM experimentally.

- Using L-GEM to calculate the weights in a Dynamic Fusion method for MCSs (Chapter 3). The most suitable number of nearest neighborhoods is suggested by an empirical evidence. Twenty one current dynamic fusion methods are compared with L-GEM in the experiments.

- MCS comparison using L-GEM^{MCS} (Chapter 4). The best MCS is selected from a pool of diversely trained MCSs.

- Proposing a new base classifier selection for MCSs using L-GEM^{MCS} (Chapter 5). A subset of the base classifiers is selected to construct a better MCS comparing with the original one.

1.6 Structure of the Thesis

This thesis is organized as follows:

Chapter 2 RBF Network Training

A novel training objective function using L-GEM is proposed for a RBF network. The localized generalization error bound of the network is minimized with respect to its weight parameters. The proposed training objective function is compared with three well-known training methods: Minimizing Training Error, Tikhonov Regularization and Weight Decay. Experimental results show that RBF networks trained by minimizing the proposed objective function consistently outperform other methods.

Chapter 3 Dynamic Fusion Method for MCSs

A new dynamic classifier fusion method named L-GEM Fusion Method (LFM) for MCSs is proposed. The localized generalization error upper bound for the neighborhood of a testing sample is used to estimate the local competence of the base classifiers in a MCS. Different from the recent dynamic classifier fusion methods, the proposed method considers not only the training error but also the sensitivities of the base classifiers. Experimental results show that the MCSs using LFM outperform other dynamic fusion methods.

Chapter 4 L-GEM for MCSs (L-GEM^{MCS})

The L-GEM for single classifiers is extended to the L-GEM^{MCS} for the case of MCSs. The L-GEM^{MCS} has a constant term plus four other terms: base classifier training error,

diversity of base classifier training error, base classifier sensitivity, and diversity of base classifier sensitivity. These terms are analyzed and discussed. Experimental result shows that MCSs selected using the L-GEM^{MCS} as a selection criteria is not only more accurate but also less complex (smaller number of base classifiers) than other methods.

Chapter 5 Base Classifier Selection for MCSs

Current methods for MCS construction may require the creation of a large number of base classifiers. However, some of the base classifiers may have no contribution to the improvement of the performance of the MCS. Pruning has been found to be a useful technique for performance improvement. L-GEM^{MCS} is used as a pruning criterion for better base classifier selection for a MCS, which is supported by experimental results.

Chapter 6 Conclusion and Future Work

The thesis is summarized and concluded. In addition several problems are suggested for future research.

CHAPTER 2
RBF NETWORK TRAINING

Training a classifier with good generalization capability is a major issue for pattern classification problems. However, the generalization error of a classifier cannot be computed. A common method to train a classifier $f(x)$ is by minimizing the training error (empirical risk):

$$\min R_{emp} = \frac{1}{N}\sum_{i=1}^{N}\left(f\left(x_i\right) - F\left(x_i\right)\right)^2 \tag{2.1}$$

It is well known that an over-fitting problem may occur if a classifier focuses too much on minimizing the training error which may lead to poor generalization capability. Regularization is a technique to address this problem. Essentially a regularization term is added to the training objective function to reduce the complexity of a classifier. It is based on the belief that a more "smooth' classifier will lead to better generalization capability. However, the regularization technique suffers from the fact that, if the training set is not a good representation of the feature space, then it will be unreasonable to expect the trained classifier in recognizing unseen samples which are very different from the training ones, whether the classifier being smooth or not. A more meaningful approach is to select a learning objective function related to the generalization error bound. In this chapter, we propose a new learning objective function based on the Localized Generalization Error Model (L-GEM) developed by Yeung et al [Yeung et al 2007].

A brief review of RBF training methods and regularization techniques is presented in Section 2.1. The new learning objective function is presented and applied to the Two- and

Three-Phase Learning Method in Section 2.2. Section 2.3 compares our proposed technique with the regularization one and the experimental results are presented in Section 2.4. Section 2.5 concludes this chapter.

2.1 Network Training Method

RBF networks were first introduced in 1988 [Broomhead et al 1988]. They are motivated by the locally tuned responses observed in biologic neurons, e.g. the cells in visual cortex respond selectively to simulation which is both local in retinal position and local in angle of object orientation. This local behavior inspires the design of the RBF network. RBF network is a standard three-layer feed-forward network: the first layer consisting of input units, a hidden layer containing the Gaussian function units and the last one being the output layer. The RBF network is defined as:

$$f(x) = \sum_{m=1}^{M} \alpha_m \varphi_m(x), \tag{2.2}$$

where M is the number of hidden neurons and $\varphi_m(x) = \exp\left(-\left((x-u_m)/(\sqrt{2}v_m)\right)^2\right)$ is the Gaussian function. α_m is the m^{th} weight which indicates the importance of the m^{th} Gaussian output function. The mapping between the input layer and the hidden layer is nonlinear while the transformation from the hidden layer to the output layer is linear. If the weight connected with a certain hidden neuron is close to zero, it means that particular hidden neuron has no significant effect on the final output. u_m is the center position of the Gaussian function, v_m is the width of the m^{th} center which affects the generalization capability of that neuron. A hidden neuron with a larger width has more influence on the final output.

In the training of a RBF network, after the number of neurons is determined, parameters such as center, width and weight will be tuned by using the training samples. F.

Schwenker et al [Schwenker et al 2001] categorize the RBF network training into two types: Two- and Three-Phase Learning.

2.1.1 Two Phase Learning

In the Two-Phase learning, the center and width are decided first. It takes advantage of the well-defined meanings of the RBF network parameters. The centers are highly related to the density of data points. Without using the class label, K-means [Kiernan et al 1996, Mendoza et al 2009, Moody 1991, Moody et al 1989] is the most typical method to divide the samples into different clusters. It minimizes the distance between the center and the samples in that cluster. Since class label is available in a classification problem, supervised method could be used. Learning Vector Quantization (LVQ) algorithm [Vogt 1993] was proposed by Kohonen [Kohonen 1990] for vector quantization and classification tasks. Different from the unsupervised clustering, each cluster center belongs to a class. A center is moved closer to samples in the same class and away from samples belonging to a different class. Decision tree [Kubat 1998, Yoo et al 1995] can be used to separate the feature space into different regions. Each region represents a center of the RBF network.

The next step is the selection of the width for each center. The width can be determined by computing the variance of all samples in a cluster [Brizzotti et al 1999, De Castro et al 1999]. K-nearest-neighbor algorithm [Mak et al 1998, Musavi et al 1992, West et al 2009] is sometimes applied and the width is calculated as the mean of distances among the centers belonging to other k-nearest hidden neurons. The next phase is to find the weights after the centers and widths are decided. They can be easily found by linear optimization using any linear least-square method. Gradient Descent [Kiernan et al 1996] and Singular Value Decomposition (SVD) [Bruzzone et al 1999, Mak et al 1998] are two popular methods. The performance of a RBF network highly depends on the selections of

the centers and the widths. However, current methods of tuning these parameters can not guarantee good classification result.

2.1.2 Three Phase Learning

A classifier trained by the Two-Phase Learning is adjusted again through a further optimization procedure in the Three-Phase Learning. The trained classifier is refined by an adaptation of all parameters simultaneously by the gradient descent method:

$$\frac{\partial(R_{emp})}{\partial \alpha_k} = \frac{1}{N}\sum_{i=1}^{N}\left(2(f(x_i)-F(x_i))\exp\left(-\sum_{r=1}^{R}\left(\frac{x_{ir}-u_{kr}}{\sqrt{2}v_{kr}}\right)^2\right)\right), \tag{2.3}$$

$$\frac{\partial(R_{emp})}{\partial u_{kt}} = \frac{1}{N}\sum_{i=1}^{N}\left(2(f(x_i)-F(x_i))\alpha_k \exp\left(-\sum_{r=1}^{R}\left(\frac{x_{ir}-u_{kr}}{\sqrt{2}v_{kr}}\right)^2\right)\left(\frac{x_{it}-u_{kt}}{v_{kt}^2}\right)\right), \tag{2.4}$$

$$\frac{\partial(R_{emp})}{\partial v_{kt}} = \frac{1}{N}\sum_{i=1}^{N}\left(2(f(x_i)-F(x_i))\alpha_k \exp\left(-\sum_{r=1}^{R}\left(\frac{x_{ir}-u_{kr}}{\sqrt{2}v_{kr}}\right)^2\right)\left(\frac{(x_{it}-u_{kt})^2}{v_{kt}^3}\right)\right). \tag{2.5}$$

In [Meghabghab et al 2004, Montazer et al 2009, Schwenker et al 2001], the experimental results showed that the Three-Phase Learning method is better than the Two-Phase Learning method. After a RBF network is trained, the gradient descent acts as a tuning method to refine the classifier. Hence, the number of learning iterations as well as the training time can be reduced.

The objective of the training methods mentioned above is to minimize the mean square error (MSE) of the training set. However, the exclusive consideration of the training accuracy may lead to an ill-posed problem [Caruana et al 2000, Kon et al 2001, Krzyiak et al 1996]. Regularization methods are proposed to address this ill-posed problem.

2.1.3 Regularization

A problem is ill-posed if it does not satisfy all three conditions: existence, uniqueness and continuity [Poggio et al 1990]. In most pattern classification problems, the information contained in a training set is not sufficient to determine uniquely the classifier in regions without samples [Poggio et al 1989]. For a given training set and a desirable performance level, many classifiers could be constructed to meet the given requirements with obviously different generalization capabilities, hence an ill-posed problem. To solve it, some prior knowledge is needed. Regularization assumes a smooth objective function. Smoothness means that two similar inputs correspond to two similar outputs. The main idea of regularization theory is to have a new training objective function which depends not only on the training error but also on the smoothness of the output function.

$$\min R_{emp+reg} = R_{emp} + \lambda\phi(f), \tag{2.6}$$

where λ is the regularization parameter which controls the trade-off between the training error and the smoothness of the classifier and $\phi(f)$ is the regularization term. The regularization term for the well-known Tikhonov regularization method [Chen et al 2000] is defined as $\phi(f) = \|Df\|^2 / 2$, where D is a linear differential operator which is the Frechet differential of the Tikhonov functional [Haykin 1994, Tikhonov et al 1977], and it can be interpreted as the local linear approximation of the curve. This method makes the solution smooth and satisfying the property of continuity. Based on Tikhonov regularization theory, Poggio et al. proposed a regularization network in [Poggio et al 1990]. In a regularization network, the number of hidden neurons of the RBF network is exactly the same as the number of samples. However, the complexity of the RBF network is large when the training set is big. To overcome this problem, a generalized RBF network was proposed [Poggio et al 1990]. The generalized RBF network is an approximated solution of the regularization network. The number of hidden neurons of a generalized RBF network can be less than the

number of samples. Another well known regularization method is called weight decay or ridge regression [Ng 2004, Young et al 2007]. The regularization term in a weight decay method is the norm of weight, which is $\phi(f) = \|\alpha\|_2^2$. Large α values cause large variations in the outputs of the classifier and may not achieve high predictive performance. Xue et al. [Xue et al 2008] proposed Locality Regularization (LR) which reduces the output differences of *k*-nearest training samples according to the L-GEM. In LR, the *k* value of each sample is the number of nearest samples which are in the same class. The regularization term is defined as the sensitivity measure, $\phi(f) = \sqrt{\alpha^T \widetilde{K}' \alpha}$, where $\widetilde{K}' = \frac{1}{N}\sum_{b=1}^{N}\sum_{j=1}^{k} K_b'(\Delta x) K_b'(\Delta x)^T p(x)$, $K_b'(\Delta x) = \left[K(x_1, x_j) - K(x_1, x^{(b)}), \ldots, K(x_N, x_j) - K(x_N, x^{(b)})\right]^T$ and $K(x_i,x_j)$ is the Gaussian kernel. However, their sensitivity term is different from those defined in the L-GEM in the sense that their training objective function only considers the training samples, while the L-GEM takes into consideration unseen samples in a neighborhood of each training sample. Hence the sensitivity term in LR is not related to any generalization capability. Moreover, since the number of centers of RBF network is equal to the number of training samples, a large training set could result in a complex classifier. The figure Figure (2.1) summarize these three regularization methods.

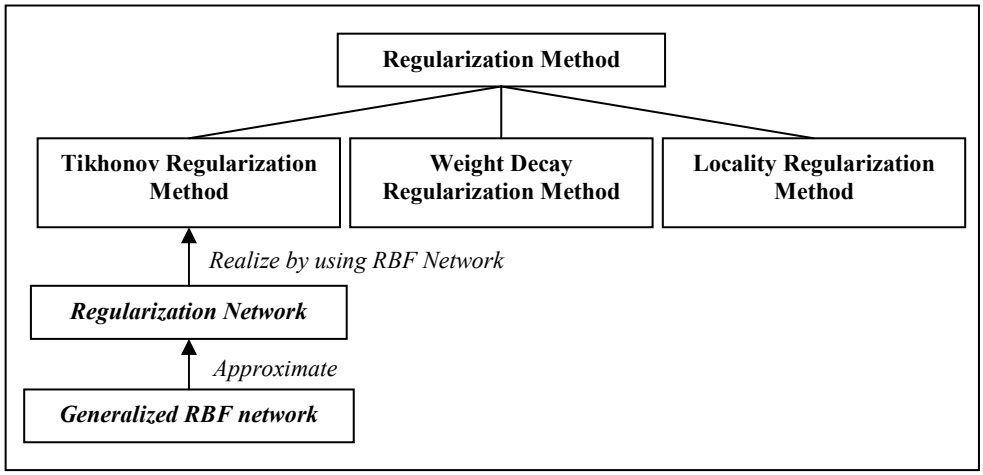


Figure 2.1 Regularization methods

Conclusion

In conclusion, only minimizing the training error may cause too complex classifiers. These complex classifiers will over-fit the training sample and their generalization ability is not expected to be good. The regularization technique was proposed to overcome the over-fitting problem. However, in its attempt to achieve a smoother function, it may need to sacrifice the training accuracy. In other words, the classifier's failure in recognizing some local points (certain training points and their neighboring points) may not be offset by any success in achieving a globally more smooth function, i.e., it may still fail to recognize unseen points very different from the training ones, be it smooth or not. This prompts a search for a more meaningful learning objective function which is more closely related to the generalization capability of the classifier.

2.2 L-GEM as an Objective Function for Learning

The regularization methods attempt to achieve a smoother classifier. However, the relation between smoothness and the generalization ability of a classifier is not clear. L-GEM, which is the upper error bound of the unseen samples in the Q neighborhood, is tied to the generalization capability and it may be a more meaningful training objective function. In this section, a new learning model using L-GEM as an objective function is proposed and the properities of this model are discussed. The section is organized as follows: The proposed learning method with L-GEM is introduced in Section 2.2.1. Section 2.2.2 and 2.2.3 discuss the properties of the new learning method. Finally, this new learning method is applied to Two- and Three-Phase Learning Method in Section 2.2.4 and 2.2.5.

2.2.1 Learning Model with L-GEM

In this proposed training method, R^*_Q is used as the objective function. As A is fixed for a given dataset, they will not affect R^*_Q in training. Hence, these two parameters are ignored and the objective function is now defined as:

$$\min\ R'_Q = \sqrt{R_{emp}} + \sqrt{E_Q\left((\Delta Y_i)^2\right)}. \tag{2.7}$$

This objective function is used to adjust the weights. It could also be used to train the centers and widths for RBF network. Gradient descent is used to minimize this objective function. The parameter set $P = (\alpha_k, u_{kt}, v_{kt})$ is adjusted by a small distance in which the objective function decreases most rapidly.

$$P^{(\tau+1)} = P^{(\tau)} - \gamma \Delta P^{(\tau)}, \tag{2.8}$$

where γ is the training parameter which controls the size of the change in each update. The adaptation rule is defined by

$$\Delta P^{(\tau)} = \frac{\partial}{\partial P^{(\tau)}} R'_Q = \frac{\partial}{\partial P^{(\tau)}} \sqrt{R_{emp}} + \frac{\partial}{\partial P^{(\tau)}} \sqrt{E_Q\left((\Delta Y_i)^2\right)}, \tag{2.9}$$

where $\dfrac{\partial \sqrt{R_{emp}}}{\partial P^{(\tau)}} = \dfrac{1}{2\sqrt{R_{emp}}} \dfrac{\partial}{\partial P^{(\tau)}} R_{emp}$ and

$$\frac{\partial \sqrt{E_Q\left((\Delta Y_i)^2\right)}}{\partial P^{(\tau)}} = \begin{cases} \dfrac{1}{2\sqrt{E_Q\left((\Delta Y_i)^2\right)}} \dfrac{\partial}{\partial P^{(\tau)}} E_Q\left((\Delta Y_i)^2\right) & q > 0 \\ 0 & q = 0 \end{cases}.$$

For the weight adaptation rule, $\dfrac{\partial}{\partial \alpha_k} R_{emp}$ is defined as in Equation (2.3) and $\dfrac{\partial}{\partial \alpha_k} E_Q\left((\Delta Y_i)^2\right)$ is given by

$$\frac{\partial E_Q\left((\Delta Y_i)^2\right)}{\partial \alpha_k} = \frac{2q^2}{3}\left[\alpha_k \exp\left(\frac{\mathrm{var}(s_k)}{2} - E(s_k)\right)\left(\begin{array}{l}\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + \left(\mu_{x_i} - u_{ji}\right)^2}{v_{ji}^4} \\ + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ki}^4}\end{array}\right)\right], \tag{2.10}$$

where $E(s_k) = \sum_{i=1}^{n}\left(\left(\sigma_{x_i}^2 + \left(\mu_{x_i} - u_{ki}\right)^2\right)/v_{ki}^2\right)$,

$\mathrm{var}(s_k) = \sum_{i=1}^{n}\left(\left(E_D\left(\left(x_i - \mu_{x_i}\right)^4\right) - \sigma_{x_i}^4 + 4\left(\mu_{x_i} - u_{ki}\right)E_D\left(\left(x_i - \mu_{x_i}\right)^3\right) + 4\sigma_{x_i}^2\left(\mu_{x_i} - u_{ki}\right)^2\right)/v_{ki}^4\right)$ and μ_{x_i} is the mean of i^{th} feature of the sample in training set.

Similarly, the adaptation rule for the center ($\frac{\partial}{\partial u_{kt}}R'_Q$) and the width ($\frac{\partial}{\partial v_{kt}}R'_Q$) can be derived. $\frac{\partial}{\partial u_{kt}}R_{emp}$ and $\frac{\partial}{\partial v_{kt}}R_{emp}$ are Equations (2.4) and (2.5). $\frac{\partial}{\partial u_{kt}}E_Q\left((\Delta Y_i)^2\right)$ and $\frac{\partial}{\partial v_{kt}}E_Q\left((\Delta Y_i)^2\right)$ are as follows:

$$\frac{\partial E_Q\left((\Delta Y_i)^2\right)}{\partial u_{kt}} = -\frac{2q^2(\alpha_k)^2}{3v_{kt}^4}\exp\left(\begin{array}{l}\frac{\mathrm{var}(s_k)}{2} \\ -E(s_k)\end{array}\right)\left[\left(\mu_{x_t} - u_{kt}\right)\left(1 - v_{kt}^2\left(\begin{array}{l}\sum_{i=1}^{n}\frac{\left(\mu_{x_i} - u_{ki}\right)^2}{v_{ki}^4} \\ + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ki}^4}\end{array}\right)\right)\right], \tag{2.11}$$

$$\frac{\partial E_Q\left((\Delta Y_i)^2\right)}{\partial v_{kt}} = \frac{2q^2(\alpha_k)^2}{-3v_{kt}^5}\exp\left(\begin{array}{l}\frac{\mathrm{var}(s_k)}{2} \\ -E(s_k)\end{array}\right)\left[\begin{array}{l}\left(2\left(\mu_{x_t} - u_{kt}\right)^2 + \frac{0.4Q^2}{3}\right) \\ -\left(\begin{array}{l}\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + \left(\mu_{x_i} - u_{ji}\right)^2}{v_{ji}^4} \\ + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ki}^4}\end{array}\right)\left(v_{kt}^2\left(\mu_{x_t} - u_{kt}\right)^2\right)\end{array}\right]. \tag{2.12}$$

2.2.2 Stopping Criteria

The iterative learning process stops when a pre-selected number of iterations or a desired value of R'_Q is achieved. Obviously, prior knowledge is needed to set these threshold

values. Other more practical criteria will depend on the changes of R'_Q and $|R'_Q|$. It is reasonable to stop the training if the performance change of the classifier is not significant.

2.2.3 Time Complexity

Let N denote the number of training samples, n be the number of features, M be the number of hidden neurons and e be the number of training epochs. The width, center and weight (α_k, u_{kt}, v_{kt}) are updated once in each epoch by using all training samples. The time complexity of using R'_Q as the training objective is $O(eNMn^2)$, in comparison with $O(eNMn)$, which is the time required for training by minimizing the training error.

2.2.4 Two-Phase Learning Method using R'_Q ($2PLR_Q$)

When R'_Q is used in the Two-Phase Learning, only weight adaptation rule ($\Delta\alpha$) will be used. The center and the width of a Gaussian function of the RBF network are determined by using clustering methods. Finally, the weight will be updated by $\Delta\alpha$ using gradient descent to minimize the R'_Q. This model will be used to compare R'_Q with regularization method which only considers weight update.

2.2.5 Three-Phase Learning Method using R'_Q ($3PLR_Q$)

Considering that fact that the time complexity of R'_Q is larger than other methods, and the performance of the Three-Phase Learning is better than the Two-Phase Learning with a reduced number of learning iterations, the Three-Phase Learning method is expected to be a good choice for R'_Q.

At first, the centers and widths of RBF network are determined by any method, e.g. the ones mentioned in Section 2.1. The weights are calculated by the Singular Value Decomposition (SVD) technique. Then the parameters of the trained network will be fine-tuned by minimizing the objective function R'_Q using the gradient descent method.

2.3 Comparison between R^*_Q and Regularization

2.3.1 Motivation

L-GEM was developed with the observation that predictions of unseen samples far away from training samples are not reliable and could be misleading. It provides a generalization error upper bound for unseen points within a neighborhood containing all training points. In L-GEM, a classifier with a small error bound (R^*_Q) is preferred. A small R^*_Q indicates that a classifier recognizes well the training samples and its outputs are stable in a neighborhood that contains the training samples. On the other hand, regularization is motived to address the ill-posed problems in pattern classification. Only minimizing the training error in learning may result in many solutions. Additional prior information is assumed to make the problem well-posed. Smoothness is a reasonable assumption in approximation [Poggio et al 1990]. Smoothness means that a small change in input will cause a small change in output. A smooth classifier with small training error is preferred for regularization.

2.3.2 Sensitivity and Regularization Term

The formula of R^*_Q consists of four terms: training error, sensitivity measure, A and ε. Since A and ε are fixed for a given training set, only training error and sensitivity measure will affect the classifier's performance during the training. This framework is similar to the regularization method which minimizes the training error and the regularization term. The only thing different is between the sensitivity measure and the regularization term. However, it is noted that the relation between training error and sensitivity measure in the L-GEM is nonlinear.

Sensitivity measure is defined as the mean value of the squared output differences between training and unseen samples within its Q neighborhood. The value of the sensitivity

measure is large when the output differences between unseen samples in the Q neighborhood and training samples are large. It means that the outputs of the trained classifier change substantially when the differences in inputs are relatively small. Classifiers with relatively large sensitivity values are not preferred in the L-GEM framework. Regularization assumes a smooth target function. Smoothness means small changes in some input parameters yield a correspondingly small change in the outputs. Different regularization methods have different regularization terms. Tikhonov Regularization considers the differentiation of the function and Weight Decay focuses on the values of the weights. As a result, a smoother classifier is trained. Conceptually, minimization of sensitivity measure and regularization term is the same and both try to stabilize the outputs. On the other hand, the sensitivity measure can be shown to be a special case of the weight decay. Let

$$V_j = \frac{q^2}{3}\exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\left(\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + (\mu_{x_i} - u_{ji})^2}{v_{ji}^4} + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ji}^4}\right). \tag{2.13}$$

The sensitivity term can be written as:

$$E_Q\left((\Delta Y_i)^2\right) \approx \frac{q^2}{3}\sum_{j=1}^{M}\left[(\alpha_j)^2\exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\left(\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + (\mu_{x_i} - u_{ji})^2}{v_{ji}^4} + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ji}^4}\right)\right] = \sum_{j=1}^{M}V_j(\alpha_j)^2 . \tag{2.14}$$

This expression is similar to the one obtained in [Poggio et al 1990] which shows that the sensitivity measure is a weighted norm of weights. Here the sensitivity measure appears as a special form of the regularization term in Weight Decay which is $\sum_{j=1}^{M}\lambda_j(\alpha_j)^2$. Comparison of Equation (2.14) and $\sum_{j=1}^{M}\lambda_j(\alpha_j)^2$ indicates that V plays a role similar to the trade-off parameter (λ) in the regularization method, balancing the training error and the

smoothness minimization. In summary, the regularization method minimizes the smoothness of a classifier in the whole input space while the sensitivity measure only concerns the outputs of the classifier in the Q neighborhood. In this sense, the sensitivity measure is a special case of the regularization term. If some parts of the input space do not contain any training sample, a classifier cannot learn in those parts since no information is provided. As a result, considering the smoothness of outputs in the whole space may not be necessary.

Although the appearance of the sensitivity measure and the regularization term is similar, their interpretations and derivations are very different. Sensitivity measure is closely tied to the generalization error and it affects the value of R'_Q. Minimizing the training error and the sensitivity measure together can reduce the error bound of the unseen samples in the Q neighborhood. On the other hand, it is not clear how the regularization term is related to the generalization ability of a classifier. It is intended to measure the smoothness of a classifier. As a result, a classifier which is smooth with good training error is constructed. Therefore, learning by using the localized generalization error is intuitively more relevant to the ultimate objective of training a classifier.

2.3.3 Parameter Determination

The regularization parameter (λ) in the regularization term controls the trade-off between the training accuracy and the smoothness of the classifier. When $\lambda = 0$, the regularization term is ignored. When $\lambda \rightarrow \infty$, the regularization term dominates the training and the training samples are neglected. The selection of the value of λ is critical and it affects significantly the performance of the trained classifier. Prior information about the classification is needed or cross validation method is used for determining λ. The role of q in R'_Q is similar to λ. In L-GEM, q controls the size of the Q neighborhood which contains the training set.

$$E_Q\left((\Delta Y_i)^2\right) \approx \frac{q^2}{3}\sum_{j=1}^{M}\left[(\alpha_j)^2 \exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\left(\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + (\mu_{x_i} - u_{ji})^2}{v_{ji}^4} + \frac{0.2}{3}q^2\sum_{i=1}^{n}\frac{1}{v_{ji}^4}\right)\right]$$

$$= q^2\sum_{j=1}^{M}\left(\frac{1}{3}(\alpha_j)^2 \exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + (\mu_{x_i} - u_{ji})^2}{v_{ji}^4}\right) + q^4\sum_{j=1}^{M}\left(\frac{0.2}{9}(\alpha_j)^2 \exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\sum_{i=1}^{n}\frac{1}{v_{ji}^4}\right)$$

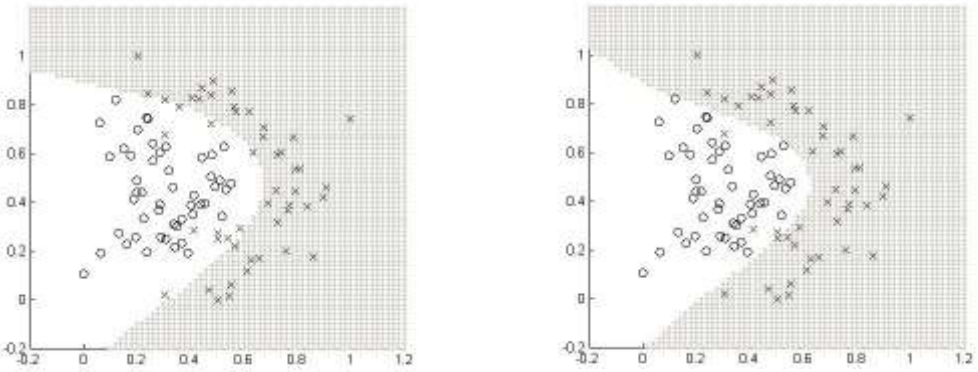
$$= q^2 t + q^4 s, \tag{2.15}$$

where $t = \sum_{j=1}^{M}\left(\frac{1}{3}(\alpha_j)^2 \exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\sum_{i=1}^{n}\frac{\sigma_{x_i}^2 + (\mu_{x_i} - u_{ji})^2}{v_{ji}^4}\right)$ and

$$s = q^4\sum_{j=1}^{M}\left(\frac{0.2}{9}(\alpha_j)^2 \exp\left(\frac{\mathrm{var}(s_j)}{2} - E(s_j)\right)\sum_{i=1}^{n}\frac{1}{v_{ji}^4}\right).$$

Given a set of training samples and predefined center, width and weight of a RBF network, the sensitivity measure could be shown as an increasing function of q. A larger q means that a bigger Q neighborhood is considered. Hence the sensitivity measure is the dominating factor in R'_Q and the effect of the training error will be reduced. On the other hand, when q is small, the importance of the training error increases since the neighborhood will become smaller. The special case when $q = 0$ reduces the problem to the minimization of the training error only. This is similar to the situation when $\lambda = 0$. When $q \to \infty$, the sensitivity measure also tends to infinity and the effect of the training error becomes insignificant. Therefore, q may be viewed as a trade-off parameter to regulate the importance between the training error and the sensitive measure. Cross validation can also be used to determine q if no prior information is available. However, there is a significant difference between λ and q. While λ is considered as a trade-off parameter between the training error and the regularization function, q offers a geometric interpretation in the input space. Any available information on the input data distribution may be useful for

determining the size of q. For example, in Figure (2.2), some of the samples from the two classes are quite close to each other. In this case, q is not expected to be too large. The decision boundary trained by R'_Q with q = 0.01 (Figure (2.2b)) performs better than the one trained with q = 0.1 (Figure (2.2a)).



a) q=0.1 b) q=0.01

Figure 2.2 The decision plane of RBF network using R'_Q with different q values on an artificial dataset. "O" and "X" represent the samples in different classes. The white and the gray areas represent the decision regions of the RBF network.

Let P be the pairwise distance from any sample of class 1 to any sample of class 2. Three intuitive determination methods of q using the geometric information from the training samples are proposed. 1) Average of all P values (*Average*), 2) The minimum of al P values (*Shortest*) and 3) The average of the smallest 1% among all P values (*1% Average*).

The shape of the Q_i neighborhood is a square and the q is the half of the width of the square. The Euclidean Distance is not a suitable distance measurement in this situation. The Chebyshev Distance, also named "Maximum Value Distance" is applied and defined as:

$$d(x_1, x_2) = \max_{1 \le k \le n} \left(| x_{1k} - x_{2k} | \right), \tag{2.16}$$

where x_i, x_{ik} and n denotes the training sample, one of its feature despondingly and number of features.

Four datatsets, which are Connectionist, Credit Approval, Dermatology and Heart, of UCI Machine Learning repository [MLR] are used. Each dataset is equally divided into two parts randomly: training and testing. Only samples in the training set are used during training. The samples in the testing set are reserved to evaluate the performances of the trained classifiers. The experiment generates ten independent runs for each pair of dataset. The inputs of all samples are normalized to [0, 1]. RBF network is applied in this experiment. The center and width of the RBF network are determined by K-mean and K-nearest-neighbor algorithm respectively. Scatter-Based Clustering (SBC) [Sohn et al 1997] is used to find the number of centers. $R_Q^{'}$ is applied as the training objective function to decide the weight. q is determined by the three intuitive methods (*Average*, *Shortest* and *1% Average*) and Cross Validation (CV). In CV, q = {0.05, 0.075, 0.1, 0.125, 0.15, 0.175, 0.2, 0.225, 0.25} are tested.

Table 2.1 CV VS Other q Determination Methods using Geomatric Information
Average Classification Accuracy of Testing Set over Ten Independent Runs

		CV	Average	Shortest	1% Average
Connectionist	Acc	79.20	69.64	75.85	79.31
	q	0.140	0.381	0.122	0.154
Credit Approval	Acc	87.62	80.36	83.01	86.98
	q	0.160	0.487	0.033	0.124
Dermatology	Acc	98.41	92.89	95.77	98.32
	q	0.250	0.481	0.167	0.223
Heart	Acc	80.61	71.74	81.20	78.93
	q	0.120	0.493	0.104	0.165
Average	Acc	86.46	78.66	83.96	85.89

Table (2.1) shows that the average classification accuracy of testing set. Although the CV has the best result, it is a trail and error technique. The three geometric based methods are more time efficient and some of them (i.e., the 1% Average) could perform almost as well as the CV method.

2.3.4 Flexibility

Generally speaking, the regularization technique can be applied to any learning method and it is more flexible than the L-GEM. However, there are constraints on

regularization technique as well. For example, Tikhonov Regularization can only be applied to a differentiable function, while L-GEM has no such restriction. On the other hand, L-GEM is a general concept and can be applied in many different applications. However, it cannot be applied to those classifiers in which the R^*_Q is not avaliable, e.g. rule-based systems.

2.3.5 Conclusion

In conclusion, sensitivity measure and regularization are similar concepts. Both describe the complexity of a classifier. The shapes of decision boundary of classifiers trained by $2PLR_Q$ and the regularization method are also smoother than the boundaries trained by non-regularization methods. However, the classifier trained by the $2PLR_Q$ is more stable in neighborhoods near the training samples. Using R'_Q as a training objective function is more suitable since it is derived from error bounds of unseen samples in the Q neighborhood. The experimental performances of the classifiers trained by using L-GEM (R'_Q) and the regularization ($R_{emp+reg}$) methods will be analyzed and discussed in the next session.

2.4 Comparing the Experimental Results of the $2PLR_Q$ and $3PLR_Q$ with Others

In this section, the performance of the $2PLR_Q$ will be analyzed and compared with several well-known methods experimentally. Eighteen datasets shown in Table (2.2) from the UCI Machine Learning repository [MLR] and Intelligent Data Analysis Group [DAG] have been used. They cover a wide range of applications involving two-class and multi-class problems. Each dataset is equally divided into two parts randomly: training and testing. The experiment generates twenty independent runs for each pair of dataset. Only samples in the

training set are used during training. The samples in the testing set are reserved to evaluate the performances of the trained classifiers. The inputs of all samples are normalized to [0, 1].

Table 2.2 Eighteen Datasets

Dataset	# Class	# Sample	# Feature
Breast Cancer Wisconsin	2	569	32
Car Evaluation	4	1728	6
Connectionist	2	208	60
Credit Approval	2	690	15
Dermatology	6	366	34
Pima Indians Diabetes	2	768	8
Solar Flare	2	1389	10
German Credit Data	2	1000	20
Glass Identification	7	214	10
Heart	2	270	13
Hepatitis	2	155	19
Ionosphere	2	351	34
Iris	3	150	4
Thyroid	2	215	5
Tic-Tac-Toe Endgame	2	958	9
Titanic	2	2201	3
Waveform	3	5000	21
Wine	3	173	13

The experimental results of the $2PLR_Q$ and regularization methods are compared and analyzed in Section 2.4.1. $3PLR_Q$ is then compared with Three-Phase Learning using the training error as an objective function in Section 2.4.2. Finally, a special situation when the training set does not represent the testing set is discussed in Section 2.4.3.

2.4.1 Comparing the Experimental Results of the $2PLR_Q$ with Regularization Methods

The performances of the $2PLR_Q$ and the well known regularization methods of a RBF networks are compared. Generalized RBF network (GRBF), Weight Decay (WD), Locality Regularization (LR) and two training methods with MSE training (Singular Value Decomposition (SVD) and Gradient Descent with training error (GD)) are considered. As these regularization methods are only concerned with weight learning, it is only appropriate

to consider the adaptation rule of weights in R'_Q and the $2PLR_Q$ is used in experiments in this section.

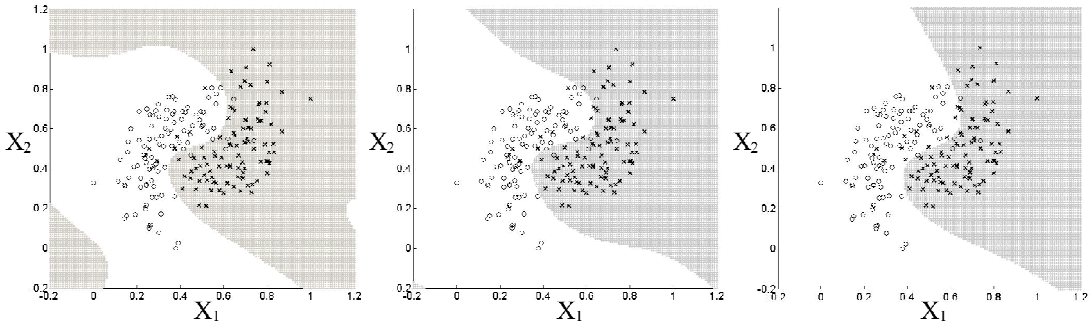
Examples of the decision boundaries of RBF networks trained by these methods are illustrated in 2.4.1.1 and their generalization abilities are compared in 2.4.1.2.

2.4.1.1 Visualization of Decision Boundary of a RBF network

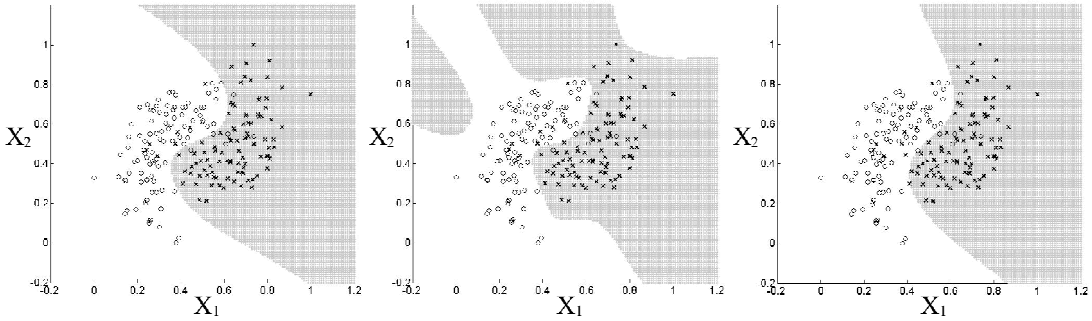
The shapes of the decision boundary of RBF network trained by different weight learning methods are discussed in this section. The Banana dataset is used for this illustration. It is a 2-class dataset with 2 features and it can be plotted on a 2-dimenssional graph. Figure (2.3) shows that the most complex decision plane is the one trained by LR by using all training points as centers. Although the regularization term helps to reduce the complexity, the decision plane is still "not smooth". As SVD and GD only focus on the training error, the shapes of the resulting decision planes are more complex comparing with those obtained by GRBF, WD and $2PLR_Q$. The decision planes for GRBF, WD and $2PLR_Q$ are similar. They are smooth in general and also separate the samples from different classes.

To further investigate the results obtained by GRBF, WD and $2PLR_Q$, three dimensional graphs are plotted. For a two-class problem, two functions f_1 and f_2 are trained to represent the two classes. A sample point x is class 1 if $f_1(x) > f_2(x)$; otherwise it is class 2. The two functions f_1 and f_2 for GRBF, WD and $2PLR_Q$ are shown in Figures (2.4a1), (2.4b1) and (2.4c1) respectively. Figures (2.4a2), (2.4b2) and (2.4c2) show the function output (f) against feature 2 (X_2) while the function output (f) against feature 1 (X_1) is shown in Figures (2.4a3), (2.4b3) and (2.4c3). The normalized samples are located between 0 and 1 and our discussion below makes reference to the region bounded by the two dotted lines. For ease of discussion "the top" shall refer to outputs above 0, and "the bottom" for outputs below 0.

The outputs of the classifiers trained by $2PLR_Q$, WD and GRBF against the feature X_2 are shown in Figures (2.4a2), (2.4b2) and (2.4c2). Although f_2 of WD is somewhat stable at the top, the outputs of f_1 change significantly and form two crests. Moreover, f_1 increases while f_2 decreases over [0,1] at the bottom. For the GRBF, f_1 (f_2) are decreasing (increasing) within [0,1], both in the top and the bottom. On the contrary, the outputs of f_1 trained by $2PLR_Q$ are quite stable at the top and change only slightly at the bottom. Although f_2 forms a crest at the top, its outputs are stable at the bottom. From Figures (2.4a2), (2.4b2) and (2.3c2), the classifier trained by the PLR_Q seems to be smoother in general than those trained by the GRBF or the WD. From Figures (2.4a3), (2.4b3) and (2.4c3), f_1 and f_2 of the WD and the GRBF form two or three crests with different heights at the top, while at the bottom, f_1 increases and f_2 decreases in [0,1]. On the other hand, f_1 and f_2 trained by the PLR_Q are simpler in shape. At the top, f_2 has only one crest and it is near 0 at the bottom. The heights of the two crests of f_1 are very similar, so its outputs will be more stable comparing with the classifiers trained by the WD and the GRBF.

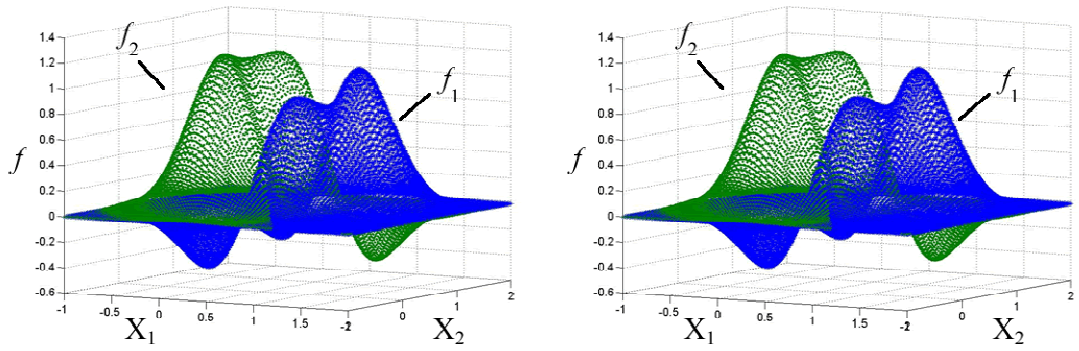


a) Singular Value Decomposition b) Gradient Descent c) Weight Decay

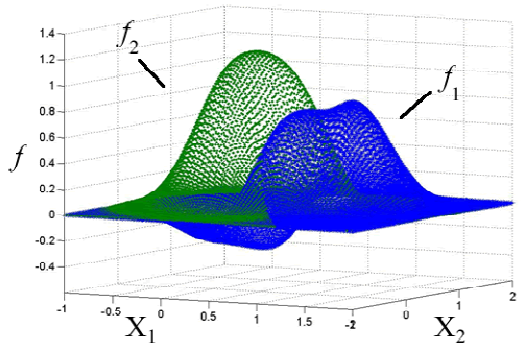


d) Generalized RBF network e) Locality Regularization f) 2PLR$_Q$

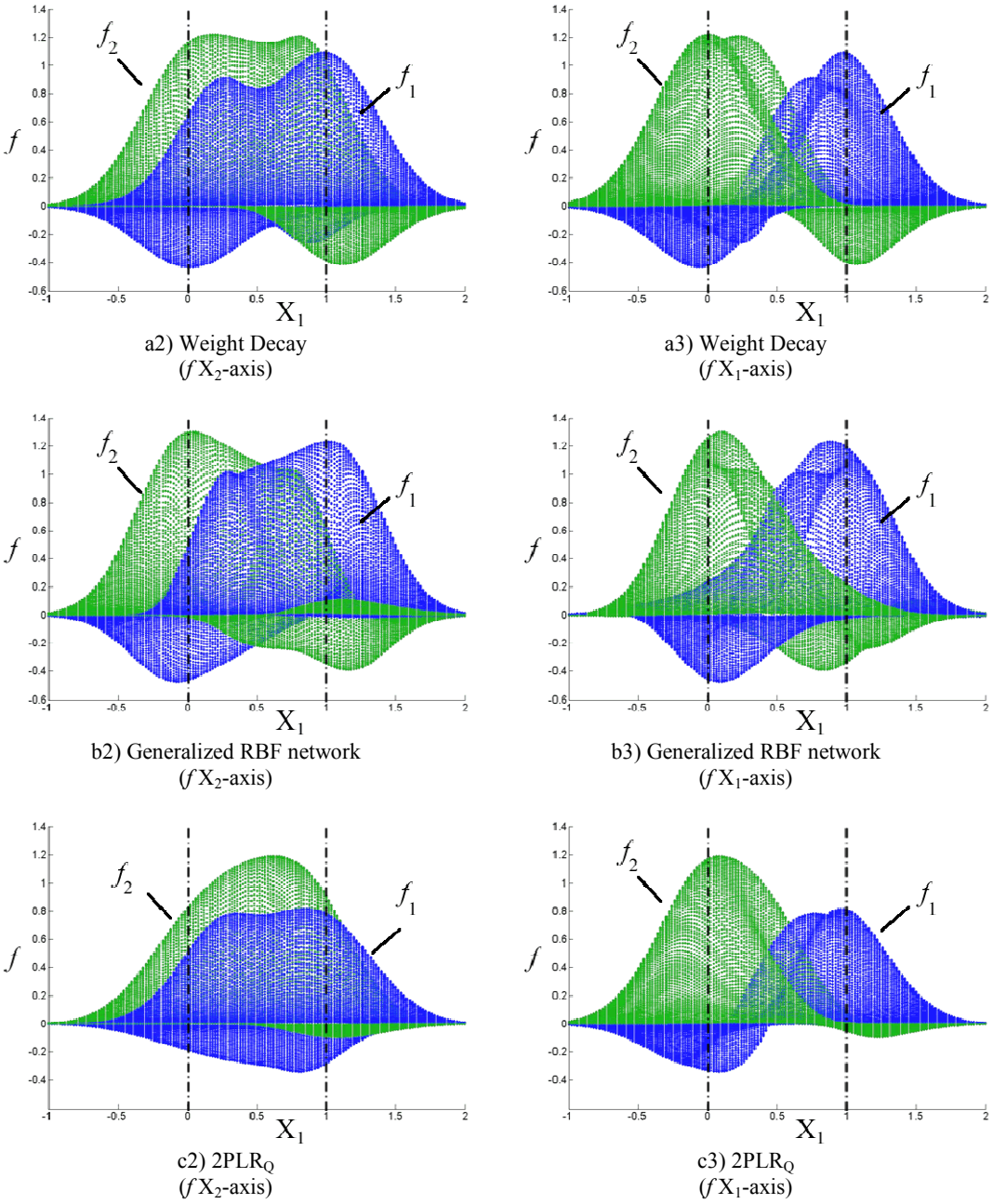
Figure 2.3 The decision planes of the RBF network using different weight learning methods on the Banana Dataset. "O" and "X" represent the samples in different classes.
The white and gray areas represent the decision regions of the RBF network.



a1) Weight Decay b1) Generalized RBF network



c1) 2PLR$_Q$



a2) Weight Decay (f X_2-axis)

a3) Weight Decay (f X_1-axis)

b2) Generalized RBF network (f X_2-axis)

b3) Generalized RBF network (f X_1-axis)

c2) $2PLR_Q$ (f X_2-axis)

c3) $2PLR_Q$ (f X_1-axis)

Figure 2.4 The functions of RBF network
using weight decay regularization, Generalized RBF network and $2PLR_Q$.

2.4.1.2 Comparison of Generalization Capability

The objective of this experiment is to compare the generalization capability of the RBF network with different weight learning methods. The center and width of the RBF classifier are determined first. Then the weights are computed by different learning methods.

K-mean (Km), Learning Vector Quantization (LVQ) and Decision tree (DT) are used to find the center, and Scatter-Based Clustering (SBC) [Sohn et al 1997] is used to find the number of centers for K-mean and Learning Vector Quantization. SBC method was chosen since it is independent of any training method. Since the number of centers for the LR is equal to the number of training samples, LR is not affected by these clustering methods. The width is calculated using the K-nearest-neighbor algorithm (Knn) and the variance method (Var). Assuming that no prior information is available, a suitable value for λ (q) is determined by cross validation [Haykin 1994] for the regularization method (R'_Q).

The average percentage of classification accuracy and its variances of the testing sets over twenty independent runs are shown in Table (2.3). Each dataset contains seven rows. For each row, an experiment using different center and width selection methods is performed. For example, the first row shows results using the Km, and the K-nn. The seventh row is the average value of the six experiments with a total of 120 independent runs. Each column represents a different weight learning method. The Student's t-test is applied to examine the statistical significance of the performance made by $2PLR_Q$ against the other methods. The value of student's t-test between the $2PLR_Q$ and other methods is shown in the bracket. When the absolute t-value is larger than 2.02 (1.98) in each experiment (for the average of all experiments), the result is significant at the 95% probability level. The value is bolded and underlined in the cell if the performance of $2PLR_Q$ is better than that specific method at a 95% significance level.

The experimental results show that the selection of the center and the width affects the performance of the RBF network significantly. Generally, the RBF network using the DT as the center selection method yields the lowest testing classification accuracy while the Km method performs the best. For the Thyroid case, the average performance by the DT is 72.95%, while LVQ is 87.17% and Km is 89.34%. When the width is decided using the Knn,

the performance of the RBF network is better than the one using the Var. For example, the average testing accuracy of using Var is 78.78% and the Knn yields 87.52% for the Thyroid dataset. It is not surprising that the overall performance of using Km and Knn is the best while the one using DT and Var is the worst. Again for the Thyroid case, the Km and the Knn is 94.02% and the DT and Var is 65.35%.

Table (2.3) also demonstrates that, in general, GRBF and WD perform better than SVD and GD. Our results are in agreement with existing findings [Bishop 1995, Ma et al 2001, Moody 1991]. This is due to the fact that the regularization term can be used to improve the generalization capability of a classifier.

The $2PLR_Q$ outperforms all other weight learning methods. Although sometimes the performance of $2PLR_Q$ is worse than others when a particular center and width determination method is used, the average result of $2PLR_Q$ is always better than SVD, GD, GRBF and WD. For example, in the case of Breast Cancer, GRBF is 0.3% better than $2PLR_Q$ when Km and Knn are used while the average testing accuracy of GRBF is 0.77% lower than $2PLR_Q$. The improvement of $2PLR_Q$ is significant in most cases. The LR uses all training samples as centers, hence its performance is independent of any center selection method. This explains why the poor performance of using DT does not affect the LR method, and in some cases the average testing accuracy of LR is better than $2PLR_Q$. For example, , the average testing accuracy of the Connectionist dataset using the LR is 72.07% and that of using the $2PLR_Q$ is 70.71%. When a more suitable center and width selection method like the Km and the Knn is used, the performance of the $2PLR_Q$ is much better than that of the LR.

The average training time and the average number of neurons are shown in Table (2.4). For the training time, since the weights can be found by using inverse method for SVD, GRBF and WD, the process is very fast. For the GD, LR and $2PLR_Q$, the learning

time is much longer since the gradient descent method is used. The training time of LR is the longest since it uses the large number of centers. The GD only considers the training error as its objective function which is simpler than the R'_Q. Hence, a shorter time is needed for the GD.

In conclusion, the experimental results show that for a predefined set of centers and widths for RBF network, the $2PLR_Q$ can train a set of weights which offer better generalization capability in comparison with other methods. This indicates that minimizing the smoothness of a classifier is not enough to achieve a better generalization capability. R'_Q may be considered as a more suitable objective function to reduce the generalization error. However, the long training time of $2PLR_Q$ is a major concern. In the next section, it will be shown that by applying the R'_Q to the $3PLR_Q$, the training time could be reduced.

Table 2.3 $2PLR_Q$ VS Other Methods (Training of Weights for a predefined set of centers and widths) Average Classification Accuracy, Variance and Student's t-test Value (In Brackets) of Testing Set over Twenty Independent Runs

	Center, Width Selection Method	SVD	GD	WD	GRBF	LR*	$2PLR_Q$
Breast Cancer	Km, K-nn	**93.17 ± 0.01 (8.03)**	**92.99 ± 0.17 (3.19)**	**94.05 ± 0.10 (2.69)**	96.36 ± 0.02 (-0.76)	**89.88 ± 0.08 (9.15)**	96.06 ± 0.01
	LVQ, K-nn	**91.71 ± 0.01 (4.85)**	**91.54 ± 0.01 (5.27)**	92.70 ± 0.02 (1.54)	**90.78 ± 0.02 (6.42)**	**89.88 ± 0.08 (5.14)**	93.30 ± 0.01
	DT, K-nn	95.02 ± 0.02 (0.76)	92.56 ± 0.52 (1.70)	94.64 ± 0.29 (0.77)	95.65 ± 0.07 (0.21)	**89.88 ± 0.08 (4.63)**	95.92 ± 0.26
	Km, Var	**93.47 ± 0.03 (2.55)**	**93.72 ± 0.02 (2.19)**	**92.57 ± 0.02 (4.35)**	94.39 ± 0.03 (0.80)	**83.45 ± 0.41 (7.64)**	94.81 ± 0.03
	LVQ, Var	**88.36 ± 0.02 (8.64)**	**89.19 ± 0.04 (6.09)**	89.09 ± 0.02 (6.99)	**91.44 ± 0.03 (2.60)**	**83.45 ± 0.41 (6.34)**	92.96 ± 0.04
	DT, Var	63.89 ± 0.78 (0.41)	63.71 ± 0.90 (0.46)	64.94 ± 0.83 (0.04)	64.82 ± 0.82 (0.09)	83.45 ± 0.41 (-7.34)	65.07 ± 0.84
	Average	87.60 ± 0.88 (1.57)	87.28 ± 1.02 (1.75)	88.00 ± 0.56 (1.38)	88.91 ± 1.29 (0.54)	**86.66 ± 0.19 (2.78)**	89.69 ± 1.23
Car	Km, K-nn	**89.48 ± 0.02 (7.03)**	**90.78 ± 0.03 (3.59)**	**90.37 ± 0.02 (5.05)**	**91.19 ± 0.02 (3.18)**	79.66 ± 0.21 (11.99)	92.58 ± 0.02
	LVQ, K-nn	**78.71 ± 0.18 (2.41)**	80.50 ± 0.24 (1.01)	80.30 ± 0.24 (1.15)	79.58 ± 0.20 (1.72)	79.66 ± 0.21 (1.64)	81.97 ± 0.19
	DT, K-nn	**88.15 ± 0.01 (7.96)**	90.15 ± 0.01 (1.86)	**90.10 ± 0.01 (2.04)**	**90.08 ± 0.01 (2.19)**	**79.66 ± 0.21 (10.52)**	90.78 ± 0.01
	Km, Var	**86.22 ± 0.12 (2.29)**	**82.15 ± 0.31 (4.32)**	86.45 ± 0.22 (1.86)	88.66 ± 0.06 (0.46)	88.05 ± 0.03 (1.02)	89.19 ± 0.22
	LVQ, Var	**62.95 ± 0.23 (2.02)**	63.51 ± 1.57 (0.82)	64.45 ± 0.70 (0.71)	64.86 ± 0.41 (0.62)	88.05 ± 0.03 (-19.53)	65.97 ± 0.22
	DT, Var	**84.81 ± 0.02 (2.10)**	**81.43 ± 1.55 (2.18)**	84.90 ± 1.07 (1.28)	85.30 ± 0.04 (1.80)	88.05 ± 0.03 (0.32)	88.64 ± 0.65
	Average	**81.72 ± 0.89 (2.46)**	**81.42 ± 2.15 (2.12)**	82.76 ± 1.02 (1.61)	83.28 ± 0.96 (1.23)	83.85 ± 0.31 (0.96)	84.86 ± 1.01
Connectionist	Km, K-nn	**76.80 ± 0.13 (2.40)**	78.60 ± 0.13 (0.73)	78.63 ± 0.13 (0.70)	78.60 ± 0.25 (0.59)	**70.73 ± 0.13 (7.88)**	79.40 ± 0.11
	LVQ, K-nn	**66.40 ± 0.01 (5.23)**	**67.20 ± 0.30 (2.64)**	**67.57 ± 0.30 (2.38)**	**67.40 ± 0.36 (2.37)**	70.73 ± 0.13 (0.39)	71.20 ± 0.16
	DT, K-nn	**78.80 ± 0.00 (2.24)**	79.40 ± 0.01 (-0.89)	79.05 ± 0.04 (0.30)	79.20 ± 0.08 (0.00)	**70.73 ± 0.13 (10.19)**	79.20 ± 0.00
	Km, Var	**65.80 ± 0.06 (2.26)**	**65.00 ± 0.05 (3.05)**	**65.31 ± 0.05 (2.73)**	**64.80 ± 0.13 (2.80)**	73.40 ± 0.17 (-4.02)	68.20 ± 0.17
	LVQ, Var	63.20 ± 0.27 (0.02)	**57.60 ± 0.00 (4.86)**	62.89 ± 0.13 (0.24)	61.60 ± 0.00 (1.41)	73.40 ± 0.17 (-6.89)	63.24 ± 0.27
	DT, Var	61.60 ± 0.55 (0.57)	61.40 ± 0.57 (0.65)	61.51 ± 0.57 (0.61)	61.20 ± 0.57 (0.73)	73.40 ± 0.17 (-5.19)	63.00 ± 0.64
	Average	68.77 ± 0.64 (1.88)	**68.20 ± 1.01 (2.13)**	69.16 ± 0.75 (1.43)	68.80 ± 1.08 (1.59)	72.07 ± 0.02 (-1.83)	70.71 ± 0.64
Credit Approval	Km, K-nn	**84.75 ± 0.01 (8.15)**	**84.68 ± 0.01 (8.61)**	**85.99 ± 0.02 (3.98)**	**85.96 ± 0.01 (5.15)**	**84.08 ± 0.06 (5.75)**	87.89 ± 0.02
	LVQ, K-nn	**83.23 ± 0.02 (5.49)**	85.06 ± 0.06 (0.91)	**82.56 ± 0.03 (6.13)**	85.27 ± 0.04 (0.68)	**84.08 ± 0.06 (2.41)**	85.65 ± 0.02
	DT, K-nn	**75.36 ± 0.13 (2.87)**	77.03 ± 0.64 (0.93)	76.72 ± 0.46 (1.23)	**75.35 ± 0.36 (2.18)**	84.08 ± 0.06 (-4.72)	78.91 ± 0.18
	Km, Var	81.78 ± 0.34 (0.46)	80.06 ± 0.44 (1.19)	81.56 ± 0.61 (0.50)	82.71 ± 0.34 (0.03)	83.89 ± 0.10 (-0.60)	82.77 ± 0.59
	LVQ, Var	**82.92 ± 0.00 (19.26)**	**83.06 ± 0.01 (14.56)**	86.45 ± 0.01 (0.08)	86.37 ± 0.01 (0.40)	**83.89 ± 0.10 (3.59)**	86.47 ± 0.00
	DT, Var	74.12 ± 0.38 (1.81)	73.46 ± 0.49 (2.00)	75.45 ± 0.57 (1.10)	76.57 ± 0.46 (0.67)	83.89 ± 0.10 (-3.15)	78.09 ± 0.58
	Average	**80.36 ± 0.30 (4.55)**	**80.56 ± 1.02 (2.72)**	**81.46 ± 0.82 (2.00)**	82.04 ± 0.73 (1.43)	83.99 ± 0.00 (-1.69)	83.30 ± 0.20
Dermatology	Km, K-nn	97.83 ± 0.01 (0.88)	**96.69 ± 0.01 (5.55)**	**97.56 ± 0.01 (2.03)**	97.83 ± 0.01 (0.88)	**79.96 ± 0.96 (8.22)**	98.06 ± 0.01
	LVQ, K-nn	**91.95 ± 0.04 (3.72)**	**91.31 ± 0.22 (2.61)**	91.65 ± 1.12 (1.10)	93.72 ± 1.95 (0.18)	**79.96 ± 0.96 (6.40)**	94.29 ± 0.04
	DT, K-nn	**96.37 ± 0.00 (3.23)**	**96.58 ± 0.00 (13.26)**	**96.95 ± 0.00 (9.14)**	97.26 ± 0.00 (3.51)	**79.96 ± 0.96 (8.03)**	97.60 ± 0.01
	Km, Var	**72.95 ± 0.26 (2.07)**	74.20 ± 0.38 (1.31)	73.24 ± 1.05 (1.34)	73.63 ± 0.36 (1.61)	83.39 ± 0.33 (-3.20)	76.94 ± 0.49
	LVQ, Var	87.98 ± 0.19 (2.22)	**85.91 ± 0.03 (4.81)**	88.35 ± 0.28 (1.78)	90.12 ± 0.60 (0.49)	**83.39 ± 0.33 (4.76)**	91.10 ± 0.20

	DT, Var	42.58 ± 0.57 (0.77)	43.12 ± 0.84 (0.53)	42.08 ± 0.76 (0.90)	42.01 ± 1.19 (0.82)		44.69 ± 0.94
	Average	81.61 ± 3.75 (0.83)	81.30 ± 5.22 (0.87)	81.64 ± 3.94 (0.81)	82.43 ± 2.71 (0.55)	81.67 ± 0.15 (1.06)	83.78 ± 4.55
Pima	Km, K-nn	**74.12 ± 0.04 (2.62)**	**73.29 ± 0.03 (4.24)**	**74.43 ± 0.04 (2.23)**	**74.53 ± 0.04 (2.03)**	**71.66 ± 0.10 (4.88)**	75.87 ± 0.05
	Km, Var	72.12 ± 0.16 (1.79)	**69.45 ± 0.01 (5.99)**	74.03 ± 0.15 (0.22)	73.29 ± 0.17 (0.82)	**69.62 ± 0.03 (5.33)**	74.29 ± 0.13
	LVQ, K-nn	**73.71 ± 0.01 (7.26)**	**74.00 ± 0.01 (4.43)**	**74.43 ± 0.01 (3.19)**	**73.29 ± 0.02 (5.84)**	**71.66 ± 0.10 (5.06)**	75.33 ± 0.00
	LVQ, Var	76.26 ± 0.02 (0.92)	75.04 ± 0.01 (4.85)	76.07 ± 0.02 (1.46)	75.86 ± 0.02 (1.84)	**69.62 ± 0.03 (15.69)**	76.63 ± 0.01
	DT, K-nn	64.08 ± 0.48 (0.89)	64.95 ± 0.74 (0.41)	66.20 ± 0.78 (-0.12)	65.12 ± 1.00 (0.31)	71.66 ± 0.10 (-3.80)	65.91 ± 0.36
	DT, Var	**70.02 ± 0.13 (2.17)**	**71.07 ± 0.02 (1.65)**	**70.24 ± 0.11 (2.06)**	71.86 ± 0.13 (0.52)	**69.62 ± 0.03 (3.31)**	72.45 ± 0.12
	Average	**71.72 ± 0.15 (3.04)**	**71.30 ± 0.27 (3.32)**	72.57 ± 0.23 (1.38)	72.32 ± 0.34 (1.59)	**70.64 ± 0.00 (6.44)**	73.41 ± 0.22
Solar Flare	Km, K-nn	**64.91 ± 0.01 (8.36)**	**64.25 ± 0.01 (9.95)**	**66.05 ± 0.01 (4.43)**	**66.15 ± 0.00 (4.44)**	**62.90 ± 0.15 (4.98)**	67.27 ± 0.01
	Km, Var	**62.32 ± 0.01 (2.93)**	**57.95 ± 0.26 (4.84)**	65.49 ± 0.16 (0.12)	**61.08 ± 0.07 (3.64)**	63.43 ± 0.02 (1.90)	65.66 ± 0.25
	LVQ, K-nn	**62.61 ± 0.02 (6.22)**	**57.70 ± 0.49 (4.80)**	64.13 ± 0.11 (1.49)	**62.16 ± 0.33 (2.42)**	**62.90 ± 0.15 (2.69)**	65.35 ± 0.02
	LVQ, Var	**63.75 ± 0.01 (9.97)**	**64.07 ± 0.05 (4.56)**	**62.30 ± 0.05 (7.69)**	66.24 ± 0.00 (1.48)	**63.43 ± 0.02 (7.72)**	66.67 ± 0.01
	DT, K-nn	**61.25 ± 0.02 (5.03)**	61.54 ± 0.25 (1.47)	**60.61 ± 0.22 (2.43)**	**60.50 ± 0.18 (2.79)**	62.90 ± 0.15 (0.38)	63.25 ± 0.02
	DT, Var	**56.37 ± 0.05 (3.47)**	58.50 ± 0.01 (1.09)	59.17 ± 0.07 (0.08)	**57.30 ± 0.05 (2.36)**	63.43 ± 0.02 (-5.65)	59.24 ± 0.09
	Average	**61.87 ± 0.11 (6.05)**	**60.67 ± 0.17 (7.81)**	**62.96 ± 0.14 (3.40)**	**62.24 ± 0.12 (5.11)**	**63.17 ± 0.08 (3.36)**	64.57 ± 0.13
German Credit	Km, K-nn	72.02 ± 0.04 (1.01)	71.86 ± 0.02 (1.47)	72.39 ± 0.03 (0.41)	72.21 ± 0.02 (0.78)	**71.00 ± 0.06 (2.36)**	72.62 ± 0.03
	LVQ, K-nn	**70.79 ± 0.02 (4.17)**	**70.12 ± 0.00 (10.32)**	**70.78 ± 0.00 (6.66)**	**70.05 ± 0.00 (11.88)**	**71.00 ± 0.06 (2.30)**	72.33 ± 0.01
	DT, K-nn	71.62 ± 0.04 (1.51)	68.26 ± 1.01 (1.98)	**69.43 ± 0.47 (2.06)**	**69.54 ± 0.35 (2.24)**	**71.00 ± 0.06 (2.09)**	72.95 ± 0.11
	Km, Var	**69.43 ± 0.00 (6.11)**	**69.12 ± 0.00 (12.19)**	**69.48 ± 0.00 (5.98)**	69.98 ± 0.00 (0.75)	**67.92 ± 0.07 (3.59)**	70.02 ± 0.00
	LVQ, Var	70.33 ± 0.10 (1.91)	70.86 ± 0.03 (1.99)	70.85 ± 0.08 (1.38)	72.07 ± 0.00 (-0.80)	**67.92 ± 0.07 (5.85)**	71.81 ± 0.02
	DT, Var	**63.76 ± 0.11 (10.93)**	**71.29 ± 0.03 (2.42)**	**68.51 ± 0.03 (7.87)**	**71.24 ± 0.02 (2.89)**	**67.92 ± 0.07 (6.87)**	72.60 ± 0.02
	Average	**69.66 ± 0.28 (4.83)**	**70.25 ± 0.92 (2.05)**	**70.24 ± 0.01 (13.81)**	**70.85 ± 0.36 (2.16)**	**69.46 ± 0.05 (11.54)**	72.06 ± 0.01
Glass	Km, K-nn	84.60 ± 0.09 (1.37)	85.04 ± 0.10 (0.91)	84.99 ± 0.15 (0.89)	84.71 ± 0.20 (1.04)	**69.02 ± 0.61 (8.79)**	86.05 ± 0.14
	LVQ, K-nn	**75.46 ± 0.11 (3.77)**	**76.01 ± 0.10 (3.27)**	**74.18 ± 0.09 (5.25)**	**73.04 ± 0.09 (6.43)**	**69.02 ± 0.61 (5.44)**	79.24 ± 0.09
	DT, K-nn	**61.83 ± 0.06 (2.24)**	61.00 ± 0.50 (1.71)	61.91 ± 0.36 (1.79)	60.62 ± 0.60 (1.78)	69.02 ± 0.61 (-2.56)	64.05 ± 0.14
	Km, Var	76.56 ± 0.21 (1.53)	77.79 ± 0.25 (0.65)	**74.36 ± 0.21 (3.05)**	**75.77 ± 0.19 (2.13)**	**74.48 ± 0.23 (2.89)**	78.79 ± 0.22
	LVQ, Var	**74.91 ± 0.07 (7.57)**	**74.78 ± 0.42 (3.91)**	**76.41 ± 0.39 (2.93)**	**78.65 ± 0.13 (2.25)**	**74.48 ± 0.23 (5.30)**	80.80 ± 0.05
	DT, Var	47.99 ± 0.24 (1.66)	**47.71 ± 0.06 (3.99)**	48.97 ± 0.07 (1.44)	**46.50 ± 0.27 (2.88)**	74.48 ± 0.23 (-22.99)	49.81 ± 0.00
	Average	70.22 ± 2.78 (1.45)	70.39 ± 2.37 (1.43)	70.12 ± 2.10 (1.68)	69.88 ± 1.89 (1.80)	71.75 ± 0.48 (0.95)	73.12 ± 2.01
Heart	Km, K-nn	79.81 ± 0.04 (1.67)	80.16 ± 0.07 (0.88)	80.08 ± 0.05 (1.11)	81.34 ± 0.07 (-0.81)	79.24 ± 0.19 (1.48)	80.78 ± 0.03
	LVQ, K-nn	**76.10 ± 0.25 (4.29)**	80.86 ± 0.09 (1.47)	81.57 ± 0.25 (0.72)	**79.98 ± 0.10 (2.20)**	**79.24 ± 0.19 (2.40)**	82.69 ± 0.22
	DT, K-nn	73.02 ± 0.26 (0.62)	**66.38 ± 0.94 (3.12)**	72.94 ± 1.14 (0.44)	74.34 ± 0.63 (-0.15)	79.24 ± 0.19 (-3.48)	74.02 ± 0.26
	Km, Var	77.43 ± 0.14 (0.63)	76.98 ± 0.17 (0.94)	77.74 ± 0.16 (0.37)	77.16 ± 0.15 (0.84)	75.84 ± 0.12 (1.96)	78.22 ± 0.17
	LVQ, Var	80.34 ± 0.08 (0.71)	79.54 ± 0.12 (1.45)	80.63 ± 0.09 (0.36)	80.78 ± 0.10 (0.19)	**75.84 ± 0.12 (5.23)**	80.95 ± 0.07
	DT, Var	**64.46 ± 0.36 (3.18)**	**65.87 ± 0.70 (2.09)**	**64.99 ± 0.39 (2.86)**	66.05 ± 0.80 (1.93)	75.84 ± 0.12 (-2.90)	70.90 ± 0.46
	Average	**75.19 ± 0.64 (2.60)**	**74.97 ± 1.96 (1.99)**	76.33 ± 1.27 (1.25)	76.61 ± 0.59 (1.28)	77.54 ± 0.03 (0.50)	77.93 ± 0.69
Hepatitis	Km, K-nn	82.44 ± 0.16 (0.76)	82.44 ± 0.16 (0.76)	**80.94 ± 0.12 (2.18)**	82.44 ± 0.10 (0.86)	**71.92 ± 0.55 (6.26)**	83.33 ± 0.12
	LVQ, K-nn	**73.51 ± 0.56 (5.94)**	**74.14 ± 0.06 (15.34)**	81.79 ± 0.45 (1.20)	**82.44 ± 0.03 (2.44)**	**71.92 ± 0.55 (6.94)**	83.63 ± 0.02
	DT, K-nn	82.74 ± 0.04 (1.15)	78.29 ± 1.32 (1.95)	78.40 ± 2.90 (1.29)	82.44 ± 0.03 (1.96)	**71.92 ± 0.55 (6.80)**	83.33 ± 0.01
	Km, Var	84.23 ± 0.06 (0.40)	84.23 ± 0.06 (0.40)	84.33 ± 0.03 (0.30)	84.52 ± 0.03 (0.00)	78.87 ± 1.74 (1.89)	84.52 ± 0.06
	LVQ, Var	**80.65 ± 0.55 (2.23)**	**81.23 ± 0.07 (4.15)**	**81.27 ± 0.44 (2.07)**	83.63 ± 0.01 (1.60)	78.87 ± 1.74 (1.89)	84.52 ± 0.06
	DT, Var	74.11 ± 0.06 (0.07)	71.29 ± 1.58 (0.81)	74.45 ± 1.05 (-0.05)	74.11 ± 1.22 (0.05)	78.87 ± 1.74 (-1.20)	74.29 ± 1.18
	Average	**79.61 ± 1.22 (2.03)**	**78.60 ± 2.19 (2.31)**	80.20 ± 1.11 (1.63)	81.60 ± 0.79 (0.58)	**75.40 ± 0.11 (7.75)**	82.27 ± 0.84
Ionosphere	Km, K-nn	**87.41 ± 0.03 (2.08)**	**86.46 ± 0.03 (4.09)**	**86.46 ± 0.02 (4.47)**	87.47 ± 0.03 (1.93)	**73.51 ± 0.61 (8.46)**	88.42 ± 0.02
	LVQ, K-nn	**79.76 ± 0.08 (6.75)**	**79.55 ± 0.05 (7.96)**	**81.40 ± 0.31 (2.57)**	**80.70 ± 0.37 (2.87)**	**73.51 ± 0.61 (6.31)**	84.80 ± 0.04
	DT, K-nn	71.23 ± 0.60 (1.25)	70.28 ± 1.58 (1.21)	68.65 ± 1.63 (1.69)	**64.30 ± 2.44 (2.56)**	73.51 ± 0.61 (0.32)	74.29 ± 0.61
	Km, Var	63.04 ± 0.43 (1.06)	69.40 ± 0.48 (-1.56)	67.55 ± 0.51 (-0.80)	67.57 ± 0.23 (-0.92)	78.87 ± 1.74 (-3.79)	65.57 ± 0.72
	LVQ, Var	**74.57 ± 0.01 (10.29)**	**78.61 ± 0.15 (3.14)**	81.78 ± 0.08 (0.29)	**77.56 ± 0.08 (4.87)**	78.87 ± 1.74 (1.05)	82.06 ± 0.09
	DT, Var	67.53 ± 0.53 (1.78)	67.84 ± 0.42 (1.75)	68.76 ± 0.25 (1.44)	70.08 ± 0.11 (0.76)	78.87 ± 1.74 (-2.40)	71.18 ± 0.31
	Average	**73.92 ± 0.78 (3.00)**	75.35 ± 0.94 (1.80)	75.77 ± 1.02 (1.46)	**74.62 ± 1.64 (2.04)**	76.19 ± 0.10 (1.51)	77.72 ± 1.14
Iris	Km, K-nn	95.87 ± 0.01 (0.85)	96.19 ± 0.01 (-0.29)	96.15 ± 0.01 (-0.17)	96.19 ± 0.01 (-0.29)	94.34 ± 0.73 (0.92)	96.10 ± 0.01
	LVQ, K-nn	**91.59 ± 0.06 (3.31)**	93.97 ± 0.08 (0.61)	92.40 ± 0.09 (2.20)	93.33 ± 0.09 (1.24)	94.34 ± 0.73 (0.10)	94.54 ± 0.10
	DT, K-nn	93.02 ± 0.11 (1.07)	93.02 ± 0.12 (1.06)	93.41 ± 0.09 (0.75)	93.49 ± 0.08 (0.69)	94.34 ± 0.73 (-0.08)	94.17 ± 0.12
	Km, Var	95.87 ± 0.03 (0.56)	**86.51 ± 0.05 (15.05)**	95.06 ± 0.05 (1.78)	**94.76 ± 0.02 (2.83)**	**90.74 ± 0.37 (3.84)**	96.19 ± 0.03
	LVQ, Var	95.71 ± 0.03 (1.53)	**83.57 ± 0.34 (9.44)**	**85.63 ± 0.13 (11.63)**	**93.02 ± 0.14 (3.76)**	**90.74 ± 0.37 (4.11)**	96.71 ± 0.05
	DT, Var	**84.76 ± 0.93 (2.36)**	87.30 ± 0.47 (1.71)	90.32 ± 1.06 (0.27)	86.03 ± 0.72 (2.04)	90.74 ± 0.37 (0.16)	91.08 ± 0.51
	Average	**92.80 ± 0.22 (3.43)**	**90.09 ± 0.41 (6.71)**	**92.16 ± 0.43 (3.67)**	**92.80 ± 0.27 (3.24)**	**92.54 ± 1.16 (2.14)**	94.80 ± 0.19
Thyroid	Km, K-nn	**94.13 ± 0.06 (2.61)**	**94.46 ± 0.05 (2.15)**	95.22 ± 0.04 (1.18)	95.13 ± 0.04 (1.36)	**89.19 ± 0.07 (8.78)**	96.01 ± 0.05
	LVQ, K-nn	**85.37 ± 0.08 (5.67)**	89.48 ± 0.06 (0.72)	**87.01 ± 0.18 (2.81)**	**86.93 ± 0.22 (2.65)**	89.19 ± 0.07 (1.07)	90.03 ± 0.05
	DT, K-nn	**76.74 ± 0.34 (3.60)**	77.30 ± 1.09 (1.89)	80.10 ± 0.25 (1.40)	**78.09 ± 0.64 (2.00)**	89.19 ± 0.07 (-9.14)	81.84 ± 0.06
	Km, Var	**83.49 ± 0.04 (4.73)**	**83.41 ± 0.11 (3.72)**	**84.12 ± 0.08 (3.23)**	**83.61 ± 0.08 (3.79)**	86.64 ± 0.10 (0.08)	86.71 ± 0.05
	LVQ, Var	88.59 ± 0.08 (0.42)	**80.18 ± 0.09 (9.83)**	**87.10 ± 0.08 (2.14)**	**86.50 ± 0.09 (2.75)**	**86.64 ± 0.10 (2.46)**	88.95 ± 0.07
	DT, Var	60.24 ± 1.13 (0.60)	60.24 ± 1.25 (0.58)	61.82 ± 1.53 (0.09)	61.02 ± 0.49 (0.43)	86.64 ± 0.10 (-11.26)	62.13 ± 0.84
	Average	**81.43 ± 1.10 (2.15)**	**80.84 ± 1.41 (2.42)**	82.56 ± 1.26 (1.25)	81.88 ± 0.81 (1.95)	87.92 ± 0.09 (-3.80)	84.28 ± 1.01
Tic Tac Toe	Km, K-nn	73.91 ± 0.04 (1.68)	74.73 ± 0.03 (0.41)	74.87 ± 0.04 (0.18)	74.93 ± 0.03 (0.08)	73.46 ± 0.08 (1.94)	74.98 ± 0.04
	LVQ, K-nn	67.40 ± 0.06 (1.82)	67.88 ± 0.08 (1.15)	68.97 ± 0.06 (-0.09)	67.49 ± 0.05 (1.77)	73.46 ± 0.08 (-5.19)	68.89 ± 0.07
	DT, K-nn	**86.42 ± 0.09 (2.12)**	**84.27 ± 0.40 (2.66)**	**84.11 ± 0.33 (2.97)**	**85.39 ± 0.18 (2.61)**	**73.46 ± 0.08 (16.21)**	88.42 ± 0.09
	Km, Var	**75.53 ± 0.05 (5.46)**	**74.68 ± 0.56 (2.72)**	**75.33 ± 0.30 (3.09)**	78.60 ± 0.06 (1.14)	**73.55 ± 0.06 (7.62)**	79.45 ± 0.06
	LVQ, Var	62.51 ± 0.14 (0.95)	**60.02 ± 0.47 (2.07)**	63.14 ± 0.71 (0.24)	61.29 ± 0.31 (1.56)	73.55 ± 0.06 (-9.74)	63.63 ± 0.14
	DT, Var	**64.39 ± 0.01 (7.29)**	63.99 ± 1.35 (1.23)	65.67 ± 1.15 (0.64)	**65.39 ± 0.00 (5.77)**	73.55 ± 0.06 (-9.77)	67.22 ± 0.02
	Average	71.69 ± 0.86 (1.95)	**70.93 ± 1.07 (2.49)**	72.01 ± 0.62 (1.82)	72.18 ± 0.68 (1.60)	73.50 ± 0.01 (0.41)	73.77 ± 0.49
Titanic	Km, K-nn	78.34 ± 0.00 (0.06)	77.80 ± 0.02 (1.64)	78.23 ± 0.01 (0.40)	78.31 ± 0.00 (0.24)	78.28 ± 0.00 (0.41)	78.35 ± 0.00
	LVQ, K-nn	78.31 ± 0.00 (-0.41)	78.07 ± 0.01 (0.84)	78.22 ± 0.00 (0.12)	**77.60 ± 0.00 (4.97)**	78.28 ± 0.00 (-0.22)	78.24 ± 0.00
	DT, K-nn	78.74 ± 0.01 (0.04)	**78.21 ± 0.01 (2.04)**	78.48 ± 0.01 (0.91)	78.81 ± 0.00 (-0.31)	**78.28 ± 0.00 (2.58)**	78.75 ± 0.00

	Km, Var	78.31 ± 0.00 (0.04)	**69.90 ± 0.09 (3.79)**	76.48 ± 0.48 (0.72)	77.78 ± 0.01 (0.28)	78.33 ± 0.00 (0.03)	78.39 ± 0.91
	LVQ, Var	78.22 ± 0.00 (-0.10)	77.68 ± 0.01 (1.33)	**77.20 ± 0.01 (2.66)**	77.67 ± 0.00 (1.55)	78.33 ± 0.00 (-0.43)	78.19 ± 0.02
	DT, Var	72.32 ± 0.01 (0.32)	72.35 ± 0.02 (0.18)	72.33 ± 0.02 (0.23)	72.36 ± 0.00 (0.27)	78.33 ± 0.00 (-23.91)	72.42 ± 0.01
	Average	77.37 ± 0.06 (0.05)	**75.67 ± 0.13 (4.33)**	**76.83 ± 0.03 (2.05)**	77.09 ± 0.06 (0.97)	78.30 ± 0.00 (-4.00)	77.39 ± 0.06
Waveform	Km, K-nn	**83.43 ± 0.02 (1.88)**	**80.16 ± 0.06 (5.77)**	83.73 ± 0.01 (1.48)	85.43 ± 0.02 (-1.33)	**71.98 ± 1.41 (4.66)**	84.60 ± 0.06
Waveform	LVQ, K-nn	**80.73 ± 0.03 (2.24)**	79.10 ± 1.33 (1.08)	81.61 ± 0.80 (0.15)	**80.09 ± 0.14 (2.03)**	**71.98 ± 1.41 (3.72)**	81.93 ± 0.03
Waveform	DT, K-nn	**84.30 ± 0.00 (2.65)**	**82.38 ± 0.12 (3.62)**	**84.44 ± 0.03 (2.09)**	85.36 ± 0.01 (0.88)	**71.98 ± 1.41 (5.13)**	85.93 ± 0.07
Waveform	Km, Var	78.78 ± 0.04 (0.94)	**77.13 ± 0.00 (3.59)**	79.18 ± 0.04 (0.45)	79.88 ± 0.04 (-0.42)	**75.16 ± 0.01 (6.05)**	79.54 ± 0.09
Waveform	LVQ, Var	**81.38 ± 0.01 (2.50)**	**80.13 ± 0.00 (6.34)**	**79.85 ± 0.01 (5.90)**	81.93 ± 0.02 (1.24)	**75.16 ± 0.01 (15.80)**	82.53 ± 0.03
Waveform	DT, Var	**72.56 ± 0.49 (2.90)**	74.60 ± 1.46 (1.16)	72.97 ± 1.30 (1.81)	76.28 ± 0.02 (1.58)	**75.16 ± 0.01 (2.67)**	77.93 ± 0.20
Waveform	Average	**80.20 ± 0.22 (2.05)**	**78.92 ± 1.40 (2.34)**	80.30 ± 0.42 (1.77)	81.50 ± 0.14 (0.65)	**73.57 ± 0.04 (10.23)**	82.08 ± 0.79
Wine	Km, K-nn	94.79 ± 0.02 (1.23)	95.06 ± 0.02 (0.62)	95.04 ± 0.02 (0.72)	95.19 ± 0.02 (0.31)	**82.67 ± 0.26 (10.80)**	95.33 ± 0.01
Wine	LVQ, K-nn	**88.79 ± 0.08 (5.74)**	92.12 ± 0.17 (1.74)	**91.25 ± 0.15 (2.65)**	92.52 ± 0.14 (1.50)	**82.67 ± 0.26 (8.67)**	94.13 ± 0.09
Wine	DT, K-nn	95.59 ± 0.45 (0.59)	94.11 ± 0.41 (1.30)	95.32 ± 0.48 (0.70)	95.52 ± 0.60 (0.58)	**82.67 ± 0.26 (7.40)**	96.85 ± 0.47
Wine	Km, Var	94.53 ± 0.13 (0.23)	94.39 ± 0.09 (0.37)	94.40 ± 0.11 (0.35)	93.73 ± 0.15 (0.89)	**87.53 ± 0.70 (3.54)**	94.79 ± 0.14
Wine	LVQ, Var	93.86 ± 0.10 (1.89)	95.99 ± 0.04 (-0.33)	95.26 ± 0.05 (0.57)	96.53 ± 0.03 (-1.04)	**87.53 ± 0.70 (4.12)**	95.73 ± 0.09
Wine	DT, Var	63.15 ± 1.67 (1.37)	62.20 ± 1.47 (1.65)	65.08 ± 1.76 (0.90)	64.41 ± 1.81 (1.05)	87.53 ± 0.70 (-5.28)	68.87 ± 1.79
Wine	Average	88.45 ± 2.97 (1.08)	88.98 ± 2.77 (0.86)	89.39 ± 2.87 (0.68)	89.65 ± 3.00 (0.56)	**85.10 ± 0.06 (3.42)**	90.95 ± 3.44

* LR is not affected by any center selection method, i.e., Km, LVQ, and DT.

Table 2.4 $2PLR_Q$ VS Other Methods (Training of Weights for a predefined set of centers and widths)
Average Number of Center and Average Training Time over Twenty Independent Runs

dataset	Center #	Learning Time (s)					
		SVD	GD	WD	GRBF	LR	$2PLR_Q$
Breast Cancer	13.67	0.10	28.30	0.81	0.13	211.15	86.59
Car	21.93	0.44	101.76	0.57	0.48	659.28	255.09
Connectionist	23.86	0.13	7.98	0.43	0.23	234.61	22.53
Credit Approval	22.13	0.11	30.56	0.52	0.13	214.97	107.99
Dermatology	29.99	0.16	45.11	0.60	0.19	383.59	118.16
Pima	31.13	0.13	32.59	0.39	0.41	238.31	93.89
Solar Flare	23.14	0.17	45.65	0.27	0.19	396.48	166.96
German Credit	28.65	0.24	57.40	0.09	0.32	725.64	269.86
Glass	14.72	0.04	22.11	0.48	0.05	160.23	41.44
Heart	16.82	0.03	10.80	0.38	0.7	78.04	35.49
Hepatitis	10.65	0.01	3.21	0.37	0.02	65.29	7.40
Ionosphere	14.24	0.06	17.61	0.12	0.08	186.52	68.78
Iris	6.35	0.02	7.08	0.55	0.12	49.94	12.82
Thyroid	11.08	0.02	7.44	0.90	0.04	75.70	16.56
Tic Tac Toe	42.18	0.16	41.91	0.53	0.18	320.04	133.00
Titanic	12.95	0.39	93.10	0.09	0.41	498.26	259.18
Waveform	31.89	2.11	242.85	0.20	2.86	1087.76	664.43
Wine	16.44	0.02	9.85	0.42	0.03	66.11	24.33

2.4.2 Comparing the Experimental Results of the $3PLR_Q$ with Others

The previous section shows that the $2PLR_Q$ outperforms regularization methods for RBF network training. In this section, the performance of $3PLR_Q$ is evaluated experimentally. The performance of the Three-Phase Learning (3PL) which adjusts the

center, width and weight by using the training error as the objective function is compared with the $3PLR_Q$.

Similar to the setting in the previous section, the center and width are trained using different methods for the first phase of RBF network, i.e., the Km, LVQ and DT for finding the centers, and the Knn and Var for finding the widths. The weights of RBF network are initialized by the SVD in the second phase learning. As no prior information is assumed, q is determined by cross validation for the $3PLR_Q$.

The experimental result is shown in Table (2.5). For comparison purpose, the previous results of SVD, GD, WD, GRBF, LR and $2PLR_Q$ listed in Table (2.3) are included again in Table (2.5). The variances and student t-test values have been removed in Table (2.5) to facilitate a clearer presentation. Similar to the previous experiments, when the absolute t-value is larger than 2.02 (1.98) in each experiment (for the average of all experiments), the result is significant at the 95% probability level. The value is bolded and underlined in the cell if the performance of 2PLRQ is better than that specific method at a 95% significance level.

The experimental result presented in Table (2.5) shows that the $3PLR_Q$ performs better than the SVD, GD, WD, GRBF, LR, $2PLR_Q$ and 3PL in almost all cases. For the Glass datasets, the $3PLR_Q$ is around 10% better than the SVD, GD, WD, GRBF, LR and $2PLR_Q$. When the center and width selection is poor, the difference is more significant. For example, in the case of the Breast Cancer, SVD, GD, WD, GRBF, LR and $2PLR_Q$ are 63% - 65% when DT and Var are the selection methods. After the adjustments of centers, widths and weights, the $3PLR_Q$ greatly improves the performance to 76.75%. It demonstrates that RBF network can be improved significantly by adjusting the center and width parameters appropriately. Although the weight, center and width are refined in the third phase, the generalization performance of the RBF network with poor initialization cannot be as good as

the one with a good initialization. For example, the testing accuracies for the Heart dataset using the $3PLR_Q$ are higher than 80%. However, when DT and Var are used, it falls to 75%. This is caused by the local minimum problem in the gradient descent method and the RBF classifier cannot be further improved. It also explains why sometimes the $2PLR_Q$ is better than the $3PLR_Q$, for example, for the Hepatitis dataset using LVQ and K-nn, the testing accuracy of the $2PLR_Q$ is 83.63% while $3PLR_Q$ is only 80.65%. Similarly, the 3PL also faces this same problem. The result shows that in such cases, the $3PLR_Q$ still can perform better than the 3PL and the SVD. On average, the testing accuracy of the $3PLR_Q$ is 2% to 3% higher than the 3PL. It shows that the generalization capability of a trained classifier could be improved by tuning the parameters via minimizing the R'_Q.

The average number of neurons and the time complexity of using the $3PLR_Q$ are shown in Table (2.6). Longer training time is needed by $3PLR_Q$ comparing with the 3PL since an additional sensitivity term is calculated during the training. The training time of the $3PLR_Q$ is much less than the $2PLR_Q$ since the RBF network is already initialized using the Two-Phase Learning method, and thus the time needed by the third phase learning is reduced.

Table 2.5 $3PLR_Q$ and 3PL (Training of weights, centers and widths)
VS Other Methods (Training of weights only)
Average Classification Accuracy of Testing Set over Twenty Independent Runs

	Center, Width Selection Method	SVD	GD	WD	GRBF	LR*	$2PLR_Q$	3PL	$3PLR_Q$
Breast Cancer	Km, K-nn	**93.17**	**92.99**	**94.05**	96.36	**89.88**	96.06	**95.35**	96.69
	LVQ, K-nn	**91.71**	**91.54**	**92.70**	**90.78**	**89.88**	**93.30**	**93.21**	94.05
	DT, K-nn	**95.02**	**92.56**	94.64	95.65	**89.88**	95.92	95.43	96.06
	Km, Var	**93.47**	**93.72**	**92.57**	**94.39**	**83.45**	**94.81**	96.02	96.23
	LVQ, Var	**88.36**	**89.19**	**89.09**	**91.44**	**83.45**	92.96	**88.46**	92.96
	DT, Var	**63.89**	**63.71**	**64.94**	**64.82**	83.45	**65.07**	76.67	76.75
	Average	**87.60**	**87.28**	**88.00**	**88.91**	**86.66**	89.69	90.86	92.12
Car	Km, K-nn	**89.48**	**90.78**	**90.37**	**91.19**	**79.66**	**92.58**	**93.77**	96.82
	LVQ, K-nn	**78.71**	**80.50**	**80.30**	**79.58**	**79.66**	**81.97**	**87.30**	90.07
	DT, K-nn	**88.15**	**90.15**	**90.10**	**90.08**	**79.66**	**90.78**	**91.15**	94.07
	Km, Var	**86.22**	**82.15**	**86.45**	88.66	**88.05**	89.19	89.68	91.19
	LVQ, Var	**62.95**	63.51	64.45	64.86	88.05	65.97	65.42	68.11
	DT, Var	**84.81**	**81.43**	**84.90**	**85.30**	**88.05**	88.64	89.71	89.70
	Average	**81.72**	**81.42**	**82.76**	**83.28**	**83.85**	84.86	86.17	88.33
Connection	Km, K-nn	**76.80**	**78.60**	**78.63**	**78.60**	**70.73**	**79.40**	79.60	82.30
	LVQ, K-nn	**66.40**	**67.20**	**67.57**	**67.40**	70.73	71.20	**69.60**	72.20
	DT, K-nn	**78.80**	**79.40**	**79.05**	**79.20**	**70.73**	**79.20**	**80.20**	83.90
	Km, Var	**65.80**	**65.00**	**65.31**	**64.80**	73.40	68.20	**66.40**	69.20

	LVQ, Var	**63.20**	**57.60**	**62.89**	**61.60**		**63.24**	66.60	67.80
	DT, Var	61.60	61.40	61.51	61.20		63.00	62.80	64.40
	Average	**68.77**	**68.20**	**69.16**	**68.80**	72.07	70.71	**70.87**	73.30
Credit Approval	Km, K-nn	**84.75**	**84.68**	**85.99**	**85.96**	**84.08**	87.89	**85.33**	87.37
	LVQ, K-nn	**83.23**	**85.06**	**82.56**	**85.27**	**84.08**	**85.65**	**85.92**	87.20
	DT, K-nn	**75.36**	**77.03**	**76.72**	**75.35**	**84.08**	**78.91**	85.71	85.54
	Km, Var	81.78	80.06	81.56	82.71	83.89	82.77	83.05	83.49
	LVQ, Var	**82.92**	**83.06**	86.45	86.37	**83.89**	86.47	**84.58**	86.75
	DT, Var	**74.12**	**73.46**	75.45	76.57	83.89	78.09	77.02	80.69
	Average	**80.36**	**80.56**	**81.46**	**82.04**	**83.99**	83.30	**83.60**	85.17
Dermatology	Km, K-nn	97.83	**96.69**	**97.56**	97.83	**79.96**	98.06	**97.72**	98.29
	LVQ, K-nn	**91.95**	**91.31**	91.65	93.72	**79.96**	94.29	94.86	94.87
	DT, K-nn	**96.37**	**96.58**	**96.95**	**97.26**	**79.96**	**97.60**	**97.37**	97.72
	Km, Var	**72.95**	**74.20**	**73.24**	**73.63**	83.39	76.94	75.24	79.21
	LVQ, Var	**87.98**	**85.91**	**88.35**	90.12	**83.39**	91.10	92.92	92.20
	DT, Var	**42.58**	**43.12**	**42.08**	**42.01**	83.39	**44.69**	47.34	49.90
	Average	81.61	81.30	81.64	82.43	81.67	83.78	84.24	85.36
Pima	Km, K-nn	**74.12**	**73.29**	**74.43**	**74.53**	**71.66**	**75.87**	**75.24**	77.96
	LVQ, K-nn	**73.71**	**74.00**	**74.43**	**73.29**	**71.66**	75.33	76.05	75.89
	DT, K-nn	**64.08**	**64.95**	**66.20**	**65.12**	**71.66**	**65.91**	74.74	74.93
	Km, Var	**72.12**	**69.45**	74.03	73.29	**69.62**	74.29	**73.01**	75.85
	LVQ, Var	76.26	75.04	76.07	75.86	**69.62**	76.63	77.80	78.01
	DT, Var	**70.02**	**71.07**	**70.24**	71.86	**69.62**	72.45	74.00	75.03
	Average	**71.72**	**71.30**	**72.57**	**72.32**	**70.64**	**73.41**	75.14	76.28
Solar Flare	Km, K-nn	**64.91**	**64.25**	**66.05**	**66.15**	**62.90**	67.27	**66.01**	67.39
	LVQ, K-nn	**62.61**	**57.70**	64.13	**62.16**	62.90	65.35	65.11	65.38
	DT, K-nn	**61.25**	**61.54**	**60.61**	**60.50**	**62.90**	**63.25**	**65.26**	66.89
	Km, Var	**62.32**	**57.95**	65.49	**61.08**	63.43	65.66	66.23	65.73
	LVQ, Var	**63.75**	**64.07**	**62.30**	**66.24**	**63.43**	**66.67**	**65.94**	68.73
	DT, Var	**56.37**	**58.50**	**59.17**	**57.30**	63.43	**59.24**	61.12	62.25
	Average	**61.87**	**60.67**	**62.96**	**62.24**	**63.17**	64.57	64.95	66.06
German Credit	Km, K-nn	**72.02**	**71.86**	72.39	**72.21**	**71.00**	72.62	72.74	73.36
	LVQ, K-nn	**70.79**	**70.12**	**70.78**	**70.05**	**71.00**	**72.33**	**72.57**	74.76
	DT, K-nn	**71.62**	**68.26**	**69.43**	**69.54**	**71.00**	72.95	73.52	73.67
	LVQ, Var	**70.33**	**70.86**	**70.85**	**72.07**	**67.92**	**71.81**	74.10	74.81
	Km, Var	69.43	69.12	69.48	69.98	**67.92**	70.02	69.57	71.86
	DT, Var	**63.76**	71.29	**68.51**	71.24	**67.92**	72.60	**69.00**	71.86
	Average	**69.66**	**70.25**	**70.24**	**70.85**	**69.46**	72.06	71.92	73.38
Glass	Km, K-nn	**84.60**	85.04	84.99	84.71	**69.02**	86.05	86.34	86.77
	LVQ, K-nn	**75.46**	**76.01**	**74.18**	**73.04**	**69.02**	**79.24**	**83.93**	86.00
	DT, K-nn	**61.83**	**61.00**	**61.21**	**60.62**	**69.02**	**64.05**	**86.61**	88.82
	Km, Var	**76.56**	77.79	**74.36**	**75.77**	**74.48**	78.79	78.41	81.09
	LVQ, Var	**74.91**	**74.78**	**76.41**	78.65	**74.48**	80.80	83.15	82.27
	DT, Var	**47.99**	**47.71**	**48.97**	**46.50**	74.48	**49.81**	**52.09**	61.26
	Average	**70.22**	**70.39**	**70.02**	**69.88**	**71.75**	**73.12**	78.42	81.04
Heart	Km, K-nn	**79.81**	**80.16**	**80.08**	**81.34**	**79.24**	**80.78**	**80.10**	83.95
	LVQ, K-nn	**76.10**	**80.86**	**81.57**	**79.98**	**79.24**	82.69	**79.45**	84.39
	DT, K-nn	**73.02**	**66.38**	**72.94**	**74.34**	79.24	**74.02**	81.31	81.37
	Km, Var	**77.43**	**76.98**	**77.74**	**77.16**	**75.84**	**78.22**	78.55	81.56
	LVQ, Var	**80.34**	**79.54**	**80.63**	**80.78**	**75.84**	**80.95**	**81.31**	82.98
	DT, Var	**64.46**	**65.87**	**64.99**	**66.05**	75.84	**70.90**	70.93	75.38
	Average	**75.19**	**74.97**	**76.33**	**76.61**	**77.54**	77.93	**78.61**	81.61
Hepatitis	Km, K-nn	82.44	82.44	80.94	82.44	**71.92**	83.33	82.14	82.74
	LVQ, K-nn	**73.51**	**74.14**	81.79	82.44	**71.92**	83.63	**76.79**	80.65
	DT, K-nn	82.74	78.29	78.40	82.44	**71.92**	83.33	82.44	82.44
	Km, Var	84.23	84.23	84.33	84.52	78.87	84.52	85.38	86.18
	LVQ, Var	**80.65**	**81.23**	**81.27**	**83.63**	**78.87**	84.52	83.46	85.95
	DT, Var	**74.11**	**71.29**	**74.45**	**74.11**	78.87	**74.29**	**75.30**	84.23
	Average	**79.61**	**78.60**	**80.20**	81.60	**75.40**	82.27	**80.92**	83.70
Ionosphere	Km, K-nn	**87.41**	**86.46**	**86.46**	**87.47**	**73.51**	**88.42**	91.33	90.86
	LVQ, K-nn	**79.76**	**79.55**	**81.40**	**80.70**	**73.51**	84.80	**82.13**	85.21
	DT, K-nn	**71.23**	**70.28**	**68.65**	**64.30**	**73.51**	**74.29**	90.86	90.32
	Km, Var	**63.04**	**69.40**	**67.55**	**67.57**	78.87	**65.57**	**70.62**	73.10
	LVQ, Var	**74.57**	**78.61**	**81.78**	**77.56**	78.87	82.06	82.80	83.45
	DT, Var	**67.53**	**67.84**	68.76	70.08	78.87	71.18	72.07	72.75
	Average	**73.92**	**75.35**	**75.77**	**74.62**	**76.19**	77.72	81.63	82.61
Iris	Km, K-nn	95.87	96.19	96.15	96.19	94.34	96.10	96.53	96.73
	LVQ, K-nn	**91.59**	**93.97**	**92.40**	**93.33**	94.34	**94.54**	**93.17**	97.26
	DT, K-nn	**93.02**	**93.02**	**93.41**	**93.49**	94.34	94.17	**93.97**	95.24
	Km, Var	95.87	**86.51**	**95.06**	**94.76**	**90.74**	96.19	96.80	96.75
	LVQ, Var	95.71	**83.57**	**85.63**	**93.02**	**90.74**	96.71	96.25	97.46
	DT, Var	**84.76**	**87.30**	90.32	**86.03**	90.74	91.08	92.38	91.33
	Average	**92.80**	**90.09**	**92.16**	**92.80**	**92.54**	94.80	94.85	95.80
Th	Km, K-nn	94.13	94.46	95.22	95.13	**89.19**	96.01	95.36	95.47

	LVQ, K-nn	**85.37**	**89.48**	**87.01**	**86.93**		**90.03**	**89.30**	92.38
	DT, K-nn	**76.74**	**77.30**	**80.10**	**78.09**		**81.84**	86.91	87.26
	Km, Var	**83.49**	**83.41**	**84.12**	**83.61**	**86.64**	**86.71**	88.64	91.10
	LVQ, Var	88.59	**80.18**	**87.10**	**86.50**	**86.64**	88.95	91.07	90.77
	DT, Var	**60.24**	**60.24**	**61.82**	**61.02**	86.64	**62.13**	**68.60**	71.71
	Average	**81.43**	**80.84**	**82.56**	**81.88**	**87.92**	**84.28**	90.52	91.70
Tic Tac Toe	Km, K-nn	**73.91**	**74.73**	**74.87**	**74.93**	**73.46**	**74.98**	**82.12**	88.83
	LVQ, K-nn	**67.40**	**67.88**	**68.97**	**67.49**	**73.46**	**68.89**	**74.04**	76.04
	DT, K-nn	**86.42**	**84.27**	**84.11**	**85.39**	**73.46**	**88.42**	**88.79**	92.43
	Km, Var	**75.53**	**74.68**	**75.33**	**78.60**	**73.55**	**79.45**	**81.69**	90.60
	LVQ, Var	**62.51**	**60.02**	63.14	**61.29**	73.55	**63.63**	**62.47**	68.25
	DT, Var	**64.39**	63.99	65.67	**65.39**	73.55	67.22	69.20	71.43
	Average	**71.69**	**70.93**	**72.01**	**72.18**	**73.50**	**73.77**	**76.38**	81.26
Titanic	Km, K-nn	78.34	**77.80**	78.23	78.31	78.28	78.35	78.61	78.53
	LVQ, K-nn	**78.31**	**78.07**	**78.22**	**77.60**	**78.28**	**78.24**	78.83	78.90
	DT, K-nn	78.74	78.21	78.48	78.81	**78.28**	78.75	78.78	78.77
	Km, Var	78.31	**69.90**	76.48	**77.78**	78.33	78.39	78.63	78.64
	LVQ, Var	78.22	**77.68**	**77.20**	**77.67**	78.33	78.19	78.39	78.45
	DT, Var	**72.32**	**72.35**	**72.33**	**72.36**	78.33	**72.42**	**74.60**	76.32
	Average	**77.37**	**75.67**	**76.83**	**77.09**	78.30	77.39	77.97	78.27
Waveform	Km, K-nn	**83.43**	**80.16**	**83.73**	85.43	**71.98**	**84.60**	**82.43**	86.08
	LVQ, K-nn	**80.73**	79.10	81.61	**80.09**	**71.98**	**81.93**	83.14	83.34
	DT, K-nn	**84.30**	**82.38**	**84.44**	85.36	**71.98**	85.93	85.59	85.96
	Km, Var	**78.78**	**77.13**	**79.18**	**79.88**	**75.16**	**79.54**	**81.84**	83.76
	LVQ, Var	**81.38**	**80.13**	**79.85**	**81.93**	**75.16**	82.53	**81.72**	83.98
	DT, Var	**72.56**	74.60	**72.97**	**76.28**	**75.16**	77.93	**75.13**	79.93
	Average	**80.20**	**78.92**	**80.30**	81.50	**73.57**	82.08	81.64	83.84
Wine	Km, K-nn	94.79	95.06	95.04	95.19	**82.67**	95.33	95.33	95.49
	LVQ, K-nn	**88.79**	92.12	91.25	92.52	**82.67**	94.13	**90.92**	92.92
	DT, K-nn	95.59	94.11	95.32	95.52	**82.67**	96.85	95.33	95.99
	Km, Var	**94.53**	**94.39**	**94.40**	**93.73**	**87.53**	**94.79**	96.80	96.66
	LVQ, Var	93.86	95.99	95.26	96.53	**87.53**	95.73	95.05	94.98
	DT, Var	**63.15**	**62.20**	**65.08**	**64.41**	87.53	**68.87**	**71.68**	76.22
	Average	88.45	88.98	89.39	89.65	**85.10**	90.95	90.85	92.05

* LR is not affected by any center selection method, i.e., Km, LVQ, and DT.

Table 2.6 $3PLR_Q$ and 3PL (Training of weights, centers and widths)
VS Other Methods (training of weights only)
Average Number of Center and Average Training Time Over Twenty Independent Runs

Dataset	Center #	Learning Time (s)							
		SVD	GD	WD	GRBF	LR	$2PLR_Q$	3PL	$3PLR_Q$
Breast Cancer	13.67	0.10	28.30	0.81	0.13	211.15	86.59	17.84	41.65
Car	21.93	0.44	101.76	0.57	0.48	659.28	255.09	45.83	64.02
Connectionist	23.86	0.13	7.98	0.43	0.23	234.61	22.53	6.84	27.63
Credit Approval	22.13	0.11	30.56	0.52	0.13	214.97	107.99	13.44	34.56
Dermatology	29.99	0.16	45.11	0.60	0.19	383.59	118.16	36.98	53.60
Pima	31.13	0.13	32.59	0.39	0.41	238.31	93.89	18.52	35.39
Solar Flare	23.14	0.17	45.65	0.27	0.19	396.48	166.96	27.23	51.46
German Credit	28.65	0.24	57.40	0.09	0.32	725.64	269.86	44.68	76.86
Glass	14.72	0.04	22.11	0.48	0.05	160.23	41.44	13.34	18.56
Heart	16.82	0.03	10.80	0.38	0.7	78.04	35.49	6.60	22.43
Hepatitis	10.65	0.01	3.21	0.37	0.02	65.29	7.40	2.45	6.74
Ionosphere	14.24	0.06	17.61	0.12	0.08	186.52	68.78	16.31	52.68
Iris	6.35	0.02	7.08	0.55	0.12	49.94	12.82	2.98	9.94
Thyroid	11.08	0.02	7.44	0.90	0.04	75.70	16.56	3.30	13.98
Tic Tac Toe	42.18	0.16	41.91	0.53	0.18	320.04	133.00	21.34	45.67
Titanic	12.95	0.39	93.10	0.09	0.41	498.26	259.18	45.42	89.14
Waveform	31.89	2.11	242.85	0.20	2.86	1087.76	664.43	155.08	316.95
Wine	16.44	0.02	9.85	0.42	0.03	66.11	24.33	6.67	21.20

2.4.3 Experiment on a Biased Dataset

For a reasonable classification problem, a training set is expected to represent the problem in general. This means that the unseen samples should be similar to the training samples. Otherwise one can not expect a good performance of the classifier. Our L-GEM model is based on the concept of the Q neighborhood near the training samples which is expected to cover most of the unseen samples. For example, in the case of the Heart dataset, even a small value of 0.1 for q, most of the testing samples will be covered by its Q neighborhood when the training and testing set are randomly divided (Figure (2.5)).

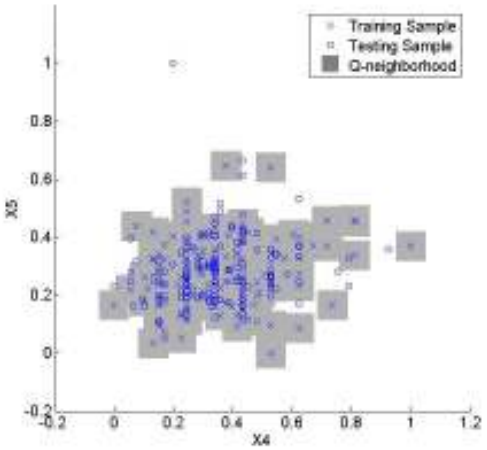


Figure 2.5 Distribution of Training and Testing sets divided by random selection for the Heart Dataset.

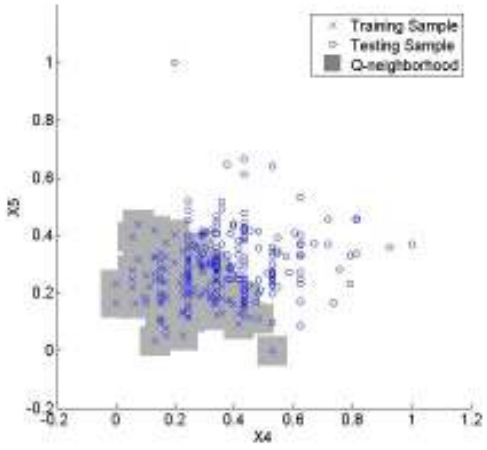


Figure 2.6 Distribution of Training and Testing sets divided by biased selection for the Heart Dataset.

This section discusses the performances of different learning methods in their handling of a special situation when the training set is sampled poorly and it cannot represent the classification problem in general. Heart dataset is again used in this experiment. When the training and testing set are divided randomly, the testing accuracy of all methods is around 76%. A Biased sampling is then selected so that the testing set does not resemble the training set. Figure (2.6) shows that most of the testing samples are not covered by the Q neighborhood.

Tables (2.7) and (2.8) show the performances of the various learning methods on the biased Heart dataset. Comparing with the randomly spitted Heart dataset, the overall

performances for the biased dataset by the same methods drop about 10%. This confirms the belief that, when the training samples don't represent the problem (which means they do not resemble the unseen samples), good performance of the classifier can not be expected. It is not surprising to observe that the variance for the learning methods for the biased dataset is larger due to the larger variations in the testing set.

Our experimental results show that the R'_Q still outperforms other methods significantly under the biased situation. This indicates a classifier trained with R'_Q can achieve a relatively better generalization capability although most of the unseen samples are not included in the Q neighborhood. In Table (2.7), LR sharply drops around 20%. This is because the performance of the LR depends too much on the training samples as it uses all training samples as centers. The performance of the 3PL drops noticeably from 78.61% to 63.13% (-15%) in Table (2.8). This is because the 3PL focuses too much on the training error. The 3PL tries to refine the trained RBF network to achieve a lower training accuracy. However, since in this case the training samples cannot represent the testing samples, its performance is poor.

Table 2.7 $2PLR_Q$ VS Other Methods (Training of Weights for a predefined set of centers and widths)
Average Classification Accuracy, Variance and Student's t-test Value (In Brackets) of Testing Set over Twenty Independent Runs

Center, Width Selection Method	SVD	GD	WD	GRBF	LR*	$2PLR_Q$
Km, K-nn	69.63 ± 0.53 (0.18)	68.89 ± 0.66 (0.47)	**62.31 ± 0.63 (3.14)**	66.35 ± 1.15 (1.25)	**58.20 ± 0.95 (4.28)**	70.05 ± 0.59
LVQ, K-nn	**62.12 ± 1.20 (3.69)**	**67.01 ± 0.40 (2.76)**	**63.20 ± 1.44 (3.12)**	68.68 ± 1.42 (1.40)	**58.20 ± 0.95 (5.40)**	73.12 ± 0.58
DT, K-nn	62.33 ± 1.70 (1.08)	61.48 ± 1.00 (1.44)	61.23 ± 0.69 (1.60)	64.13 ± 2.04 (0.61)	**58.20 ± 0.95 (2.35)**	66.77 ± 1.71
Km, Var	59.79 ± 2.78 (1.03)	60.63 ± 3.26 (0.81)	57.62 ± 2.50 (1.55)	59.89 ± 2.95 (0.99)	**56.05 ± 0.94 (2.39)**	64.66 ± 1.65
LVQ, Var	68.15 ± 0.94 (1.16)	67.41 ± 1.77 (1.16)	68.16 ± 0.71 (1.24)	64.76 ± 2.60 (1.65)	**56.05 ± 0.94 (5.32)**	71.53 ± 0.76
DT, Var	64.66 ± 0.95 (1.42)	**62.96 ± 0.71 (2.14)**	66.23 ± 1.10 (0.86)	65.71 ± 1.62 (0.91)	**56.05 ± 0.94 (4.33)**	68.89 ± 0.82
Average	**64.44 ± 0.24 (8.63)**	**65.40 ± 0.21 (7.20)**	**63.13 ± 0.16 (12.52)**	**64.92 ± 0.12 (9.50)**	**57.13 ± 0.10 (28.13)**	69.17 ± 0.12

* LR is not affected by any center selection method, i.e., Km, LVQ, and DT.

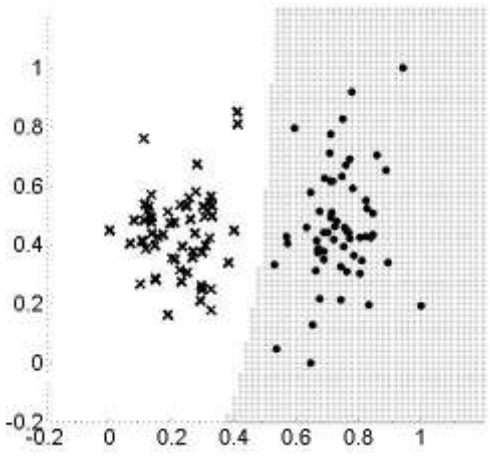
Table 2.8 $3PLR_Q$ and 3PL (Training of weights, centers and widths)
VS Other Methods (training of weights only)
Average Classification Accuracy of Testing Set over Twenty Independent Runs

Center, Width Selection Method	SVD	GD	WD	GRBF	LR*	$2PLR_Q$	3PL	$3PLR_Q$
Km, K-nn	69.63	68.89	**62.31**	66.35	**58.20**	70.05	70.26	69.95
LVQ, K-nn	**62.12**	**67.01**	**63.20**	68.68	**58.20**	73.12	**62.22**	74.50
DT, K-nn	62.33	**61.48**	**61.23**	64.13	**58.20**	66.77	**60.32**	66.78
Km, Var	59.79	60.63	57.62	59.89	**56.05**	64.66	**53.55**	64.50
LVQ, Var	68.15	67.41	68.16	64.76	**56.05**	71.53	67.83	71.79
DT, Var	**64.66**	**62.96**	66.23	65.71	**56.05**	68.89	**55.66**	69.42
Average	**64.44**	**64.73**	**63.13**	**64.92**	**57.13**	69.17	**61.64**	69.49

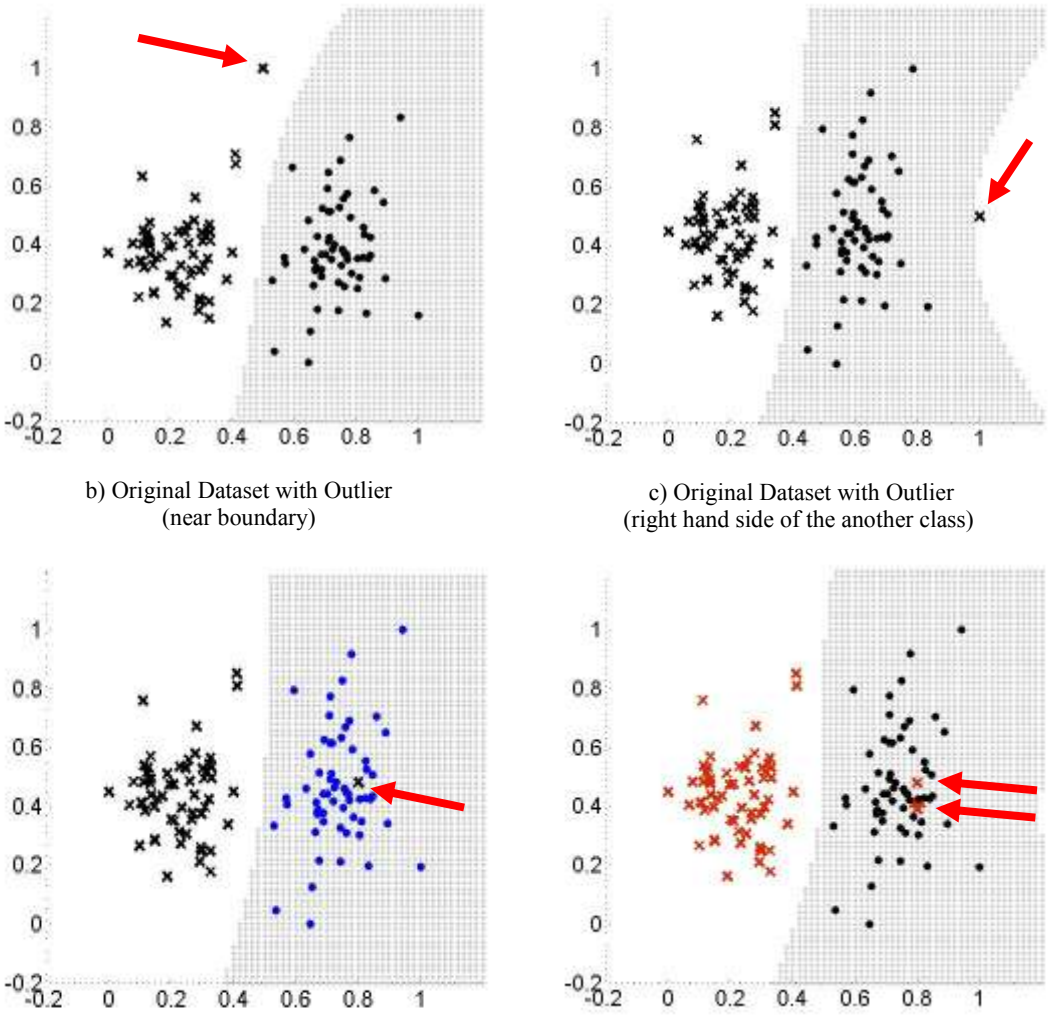
* LR is not affected by any center selection method, i.e., Km, LVQ, and DT.

2.4.4 Experiment on the dataset with outliers

The influence of outliers to a classifier trained by L-GEM is studied in this section. Figure (2.7) shows that the decision plane of a classifier trained by using $R_Q^{'}$ as objective function with different training sets. The decision boundaries are similar in datasets with different outliers. This shows that the outliers do not affect the classifiers trained by $R_Q^{'}$ significantly. It may because although the influence of outliers will be boosted, on the other handle, the effect of "good" training samples will increase too. The increasing influence of "good" training samples may cancel the effect of outlier.



a) Original Dataset



b) Original Dataset with Outlier
(near boundary)

c) Original Dataset with Outlier
(right hand side of the another class)

d) Original Dataset with Outlier
(inside the another class)

e) Original Dataset with many Outliers
(inside the another class)

Figure 2.7 The decision planes of the RBF network on the different Artificial Dataset
"O" and "X" represent the samples in different classes.
The white and gray areas represent the decision regions of the RBF network.

2.5 Summary

In this chapter, a novel training objective function (R'_Q) based on the Localized Generalization Error Model (L-GEM) is proposed. It takes into consideration the generalization error of the unseen samples in a neighborhood of the training samples. The assumption of the L-GEM that each feature of all centers must have the same width is

relaxed. R'_Q can be used to train any parameter of RBF network, including the weight, center and width.

The R'_Q and a number of representative regularization methods are compared conceptually and experimentally. Both the sensitivity term of R'_Q and the regularization term are utilized for the training of a smooth classifier. However, a major conceptual difference between the two is that the R'_Q is closely related to the errors of unseen samples in a region near the training samples, but the regularization term is not directly related to the classifier's generalization capability. The experimental results show that for a given center and width initialization, the testing accuracies of RBF network trained by minimizing the R'_Q outperforms RBF network trained by other methods which minimizes the training error and the regularization term. One possible explanation is that the R'_Q minimizes the generalization error bound of the unseen samples during the RBF network learning. It is also shown when R'_Q is applied to the Three-Phase Learning method ($3PLR_Q$), it performs better than the traditional Three-Phase Learning method minimizing the MSE. The case of the training set not resembling the testing set is also discussed. In this situation, the R'_Q still outperforms all other methods.

CHAPTER 3
DYNAMIC FUSION METHOD FOR MCSS

A critical research issue in the study of MCSs is how to combine the base classifiers, which is known as "the fusion method" [Dietterich 1997, Ho et al 1994, Ko et al 2008, Koppel et al 1996, Merz 1998, Santos et al 2008, Woods et al 1997]. Broadly speaking, base classifiers in the MCSs can be combined in two ways: static and dynamic. For a static fusion method (e.g. Majority Vote [Battiti et al 1994, Lam et al 1997] and Weighted Average [Freund et al 1996, 1997, Fumera et al 2008]), the fusion parameters are decided completely during the training phase and they will not be changed in the classification phase. However, for a dynamic fusion method (e.g. Mixture of Experts [Jacobs et al 1991] and Dynamic Integration [Puuronen et al 1999, Tsymbal et al 1998]), some fusion parameters could be changed according to the characteristics of the testing samples and the base classifiers for each testing sample.

One of the major drawbacks of a static fusion method is the assumption that all base classifiers will have the same performance in the whole input space. For example, the Weighted Average is one of the most popular static fusion methods. A weight is assigned to each base classifier according to the training accuracy. A more accurate base classifier is assigned a larger value of the weight and vice versa. However, a base classifier may perform poorly on average but it could have a good performance in a certain region (R) of the input space. The contribution in R may be ignored since a smaller weight is assigned to it. On the contrast, in a dynamic fusion method, weight is assigned to each base classifier according to

its performance on the local region where the testing sample is located. This means that each base classifier can contribute to the MCS according to its local competence. Moreover, the performance of a MCS using a static fusion method relies on the assumption that the base classifiers make independent errors. This independent error criterion guarantees an improvement of MCSs in terms of classification accuracy comparing with its base classifiers [Tumer et al 1996b, Kittler et al 1998]. However, in real pattern classification applications, it is not easy to design a set of base classifiers which satisfy this criterion [Giacinto et al 1999]. While in a dynamic fusion method, MCSs need just one base classifier that correctly classifies a testing sample [Giacinto et al 1999]. This assumption is easier to achieve than the assumption on independent error. Many studies show that dynamic fusion methods outperform static fusion methods [Paradeda et al 2008, Puuronen et al 1999, Ruta (2001), Tsymbal et al 1998, Woods et al 1997].

The weight assignment mechanism in a dynamic fusion method is called the Oracle. The Oracle decides the value of weight for each base classifier when classifying a testing sample. The information considered by the Oracle can be categorized into two types. The first type is the classification accuracy of the base classifiers. Usually, the performance of base classifiers on the entire training set or the nearest K training samples (validation samples) of the testing sample is considered [Giacinto et al 1999, Kim et al 2005, Ko et al 2008, Puuronen et al 1999, Tsymbal et al 1998]. A more accurate base classifier gets a larger weight. Another type of information considered during the weight assignment is the distance between the testing sample and the training samples. It usually acts as a punishment to a base classifier if it recognizes wrongly a sample near to the testing sample. In summary, the current methods estimate the performance of base classifiers on the testing sample using only the information provided by the training samples. Since the L-GEM developed by Yeung et al [Yeung et al 2007] estimates the error bound on the unseen samples located

within a neighborhood of the training samples, the information on the unseen samples may be useful to evaluate the local competence of base classifiers to predict the testing sample.

A review of dynamic fusion methods is presented in Section 3.1. The new Dynamic Fusion Method is presented and discussed in Section 3.2. Experimental results are shown and analyzed in Section 3.3 and Section 3.4 concludes this chapter.

3.1 Dynamic Fusion Method

Consider a population of L base classifiers trained by a given training set $D = \{(x_i, y_i)\}_{i=1}^{N}$, where N is the number of training samples. $x_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T$ is a n dimension vector denoting the i^{th} training sample, n is the number of features and the superscript T is the vector transpose. y_i represents the true class ID of x_i and $y_i \in \{\omega_c\}_{c=1}^{C}$, where C is the number of classes. For each class ω_c, the output of the MCSs using a dynamic fusion method is defined by:

$$f_c^{mcs}(x) = \sum_{l=1}^{L} w_c^l(x) f_c^l(x), \tag{3.1}$$

where x is a sample, $w_c^l(\,)$ is the weight assigned to the l^{th} base classifier calculated by the Oracle and $f_c^l(\,)$ is the output of the l^{th} base classifier. When a base classifier outputs a class label, $f_c^l(\,)$ is equal to 1 if the classifier predicts that x belongs to ω_c, otherwise $f_c^l(\,)$ is equal to 0. While the output is the probability of the base classifier to decide that the sample belongs to ω_c, $f_c^l(\,)$ is a continuous value. The class ID estimated by the MCS (y^{mcs}) of the sample x is defined by:

$$y^{mcs} = \arg\max_c f_c^{mcs}(x). \tag{3.2}$$

Figure (3.1) shows the architecture of MCSs using the dynamic fusion method. When a testing sample is presented for classification, each base classifier makes a decision on the sample. On the other hand, the Oracle assigns a weight to each base classifier. The final decision is calculated by using Equations (3.1) and (3.2).

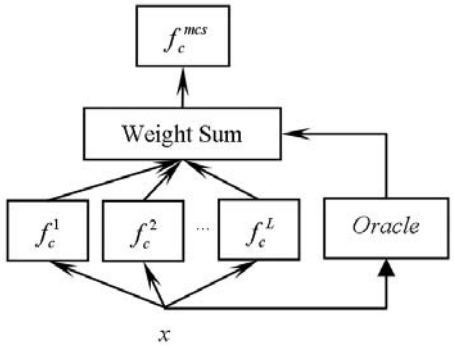


Figure 3.1 Architecture of Multiple Classifier Systems using Dynamic Fusion Method

In Dynamic Integration [Puuronen et al 1999, Tsymbal et al 1998], the base classifier's performance on the nearest *K* training samples of a testing sample is estimated using the cross validation. The weight is the product of this local accuracy and the distance between the corresponding *K* training samples and the testing sample. The concept of dynamic weight has been applied to Dynamic Selection (the best classifier is used), Dynamic Voting (weighted Voting) and Dynamic Voting with Selection (half of the best base classifiers are combined by weighted Voting). These methods have been applied to many different base classifier construction methods, e.g. Bagging, Boosting and Random Forest [Puuronen et al 2001, 2008, Tsymbal 2000, Tsymbal et al 2000a, 2000b, 2003, 2006]. The experimental results show that dynamic weight fusion methods outperform the static methods. The main drawback is that many classifiers need to be trained by the cross validation method for accuracy estimation. The estimated local performance may not reflect the true local performance of the final base classifiers. This is because the cross validation method is used to estimate the performance of a classifier on the entire input space rather than a local region [Duda et al 2000].

K-Nearest-Oracles (KNORA) Dynamic Selection Method was proposed in [Ko et al 2008]. The concept of KNORA is similar to the Dynamic Integration. Rather than estimating the performance of the base classifiers by cross validation, a validation set is used. For a testing sample, the weight of a base classifier is calculated according to its performance on the nearest *K* neighbors in the validation set. Four different methods were proposed. In KNORA-ELMINATE, only the outputs of the base classifiers which classify the nearest *K* validation samples correctly are used to make the final decision. If no base classifier can classify all *K* samples correctly, *K* will be decreased until there is one classifier which can perfectly classify the sample. The weighted voting method is used in the KNORA-UNION method. A base classifier has a larger value of weight if it can classify more samples in the *K* nearest validation set. KNORA-ELIMINATE-W and KNORA-UNION-W are the same as KNORA-ELIMINATE and KNORA-UNION respectively but each vote is weighted by the Euclidean distance between the testing sample and the nearest *K* validation samples.

Dynamic Weight Update was proposed and applied to Learn++ in [Polikar et al 2003]. Every base classifier is trained by using different random training datasets. The weight of a classifier is determined by the Mahalanobis distance between the testing sample and the training dataset of that classifier. Classifiers trained with datasets closer to the testing sample are given larger weight values. As mentioned in [Polikar et al 2003], using Mahalanobis distance implicitly assumes the dataset follows a Gaussian distribution, which in general is not true. Moreover, the classifier trained with datasets close to the testing sample may also perform badly in that region. Assigning a large weight value to this classifier may not be reasonable.

In [Abraham et al 2008], a Dynamic Fusion Method is proposed which assigns a larger weight to a base classifier with higher confidence about its outputs. The continuous-

valued output can be interpreted as the probability of a sample in a class. When outputs of classifiers are near to 0 or 1, it means the classifiers have confidence about the decision. Hence a larger weight is assigned to it. A classifier will have the smallest value of weight when it outputs 0.5.

$$w_c^l(x) = \begin{cases} f_c^l(x) & f_c^l(x) \geq 0.5 \\ 1 - f_c^l(x) & otherwise \end{cases} . \tag{3.3}$$

The value of the weight is independent of the accuracy of a classifier and it depends only on the classifier's output of the testing sample. A classifier may be wrong although it has relatively high confidence on its output.

Table 3.1 Dynamic Fusion Methods proposed by Woloszynski et al

Name	M matrix	Distance Function
C1	$m_c^l(x) = f_c^l(x)$	g_1
C2	$m_c^l(x) = \begin{cases} 1 & f_c^l(x) = \max_c f_c^l(x) \\ -1 & otherwise \end{cases}$	g_2
C3	$m_c^l(x) = \begin{cases} f_c^l(x) & f_c^l(x) = \max_c f_c^l(x) \\ -\max_c f_c^l(x) & otherwise \end{cases}$	g_1
C4	$m_c^l(x) = \begin{cases} f_c^l(x) & f_c^l(x) = \max_c f_c^l(x) \\ -\max_c f_c^l(x) & otherwise \end{cases}$	g_2
C5	$m_c^l(x) = \begin{cases} f_c^l(x) & f_c^l(x) = \max_c f_c^l(x) \\ 0 & otherwise \end{cases}$	g_1
C6	$m_c^l(x) = \begin{cases} f_c^l(x) & f_c^l(x) = \max_c f_c^l(x) \\ 0 & otherwise \end{cases}$	g_2

The weight that depends on the distance function and information of the base classifier's outputs on the training set was proposed in [Woloszynski et al 2006]. The distance function measures the distances between the testing sample and the training samples. The weight for the testing sample is calculated using the entire training dataset.

$$w_c^l(x) = \sum_{i=1}^{N} g(x, x_i) m_c^l(x_i) , \tag{3.4}$$

where g is the distance dependent function. Two types of functions had been used: $g_1(x, x_i) = 1/d(x, x_i)$ and $g_2(x, x_i) = 1/\left(1 + (d(x, x_i))^2\right)$. The value of m depends on the output of a base classifier. Totally six methods had been proposed [Woloszynski et al 2006] and are summarized in Table (3.1).

An additional training process is required in some dynamic fusion methods. In Mixture of Experts [Jacobs et al 1991], the weights of the base classifiers are calculated by a neural network called the Gating Network. The Gating Network is trained using the training samples to minimize the following objective function:

$$E = \sum_{l=1}^{L}\sum_{i=1}^{N} w_c^l(x_i)\left(f_c^l(x_i) - F_c(x_i)\right)^2 \quad , \quad \text{where } \sum_{l=1}^{L} w_c^l(x) = 1 \ . \tag{3.5}$$

$F_c(x)$ is the target output of sample x in class c. Similar to the Mixture of Experts, E. Kim et al [Kim et al 2005] proposed to use a neural network to estimate the local confidence for each base classifier. The local confidence is defined into three types:

LC1: $$LC^l(x) = \frac{1 - \sum_{l=1}^{C}\left|f_c^l(x) - F_c(x)\right|}{C}, \tag{3.6}$$

LC2: $$LC_c^l(x) = 1 - \left|f_c^l(x) - F_c(x)\right|, \tag{3.7}$$

LC3: $$LC^l(x) = \begin{cases} 1 & y^l = y \\ 0 & otherwise \end{cases}, \tag{3.8}$$

where y^l denotes the estimated class ID of the classifier l. Finally, the neural network is trained to minimize the following objective function:

$$E = \sum_{l=1}^{L}\sum_{i=1}^{N}\left(LC_c^l(x_i) - w_c^l(x_i)\right)^2 \ . \tag{3.9}$$

Obviously, these fusion methods require additional training and they are more time consuming since additional training is required.

Classifier selection method [Ho et al 1994, Giacinto et al 1999] is a special case of dynamic fusion method. Rather than assigning different weights to the base classifiers, the Oracle only selects the best base classifier. The most common selection method estimates a prior and a posterior probability of the base classifier classifying the testing sample correctly using the K-nearest neighbors.

3.2 Dynamic Fusion Method using L-GEM

y^{mcs} = **ClassifySample** $\left(x_{test}, f^1, f^2, ..., f^L, D, K\right)$

1. Calculate distance between training sample and testing sample
   $dist\left(x_{test}, x_i\right) = \| x_{test} - x_i \|_2$
2. Build nearest K neighborhoods set (X^K)
   Samples with the K smallest $dist(x_{test}, x_i)$ are selected
3. Calculate the q^{max} value
   $q^{max} = \max(dist(x_{test}, x_i^K))$
4. Calculate the weight for each base classifier,
   $$w_c^t = \frac{1}{R'_{Q^K}\left(f_c^t, X^K, q^{\max}\right)}$$
5. Normalized the weight if necessary,
   $$w_c^t = \frac{w_c^t}{\sum_{l=1}^{L} w_c^l}$$
6. Combine the decisions for each class,
   $$\mu_c\left(x_{test}\right) = \sum_{l=1}^{L} w_c^l f_c^l\left(x_{test}\right)$$
7. Final Decision
   $$y^{mcs} = \arg\max_c \mu_c\left(x_{test}\right)$$

Figure 3.2 The algorithm to classify a sample using LFM

The algorithm of the L-GEM Fusion Method (LFM) is introduced in this section. The framework of LFM is as the same as the currecty existing dynamic fusion methods but different local competence measures are applied. The idea of LFM is using the L-GEM as Oracle to calculate the weights for base classifiers. When classifying a new sample, the Oracle estimates the local generalization error bound of the local region where the sample is

located by using L-GEM. The weight to each base classifier is assigned according to the estimated local error bound. The outputs issued by the base classifiers are then combined using weighted averages.

Figure (3.2) gives a description of the classification algorithm using LFM. Before classifying a testing sample (x_{test}), L base classifiers are trained by any base classifier construction method. f^l is a vector ($[f_1^l, f_2^l, ..., f_C^l]$) which denotes the l^{th} base classifier's outputs, where C is the number of classes and $l = 1...L$. D denotes the training set and K is a parameter for LFM which represents the number of training samples that are used to estimate the local generalization error bound in the L-GEM.

R^*_Q in Equation (1.7) is the MSE upper bound of unseen samples in a Q neighborhood of the entire training set. In LFM, we are only interested in the performance of a classifier in the local region where the testing sample is located. As a result, the local region of the testing sample is defined first. The distances (*dist*) between each training sample (x_i) and the testing sample are measured, where $i = 1 \ldots N$ and N is the number of training samples. The K training samples (x_i', where $i = 1...K$) with the smallest distances are chosen to form the K neighborhood set ($X^K = \{x_i'\}_{i=1}^K$). The largest value of the distance between the testing sample and samples in X^K is used as the value of q^{max} for the L-GEM. Choosing the largest value of the distance will ensure that the local neighborhood (Q_i') of each sample in X^K can cover the testing sample. The local neighborhood of the testing sample (Q^K) is defined as the union of all local neighborhoods:

$$Q^K = \bigcup_{i=1}^{K} Q_i' \text{ and } Q_i' = \left\{x \mid x = x_i' + \Delta x; |\Delta x| \le q^{\max}\right\}. \tag{3.10}$$

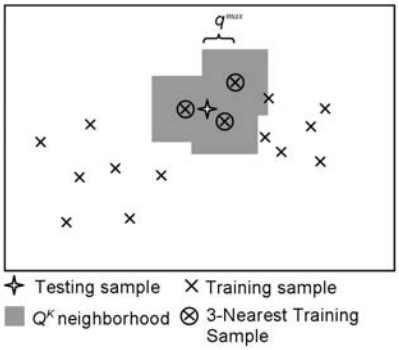


Figure 3.3 Q^K neighborhood for a testing sample with 3-nearest neighborhoods

Figure (3.3) illustrates an example with 3-nearest training samples of a testing sample in an artificial dataset. L-GEM is then used to measure the local generalization error bound ($R^*_{Q^K}$) of the Q^K region for each base classifier. Similar to Equation (1.7), the $R^*_{Q^K}$ for f can be defined as:

$$R^*_{Q^K} = \left(\sqrt{\frac{1}{K}\sum_i^K err(f, x_i)} + \sqrt{\frac{1}{K}\sum_i^K \underset{Q_i'}{E}\left(\left(\Delta Y\left(f, x_i, q^{\max}\right)\right)^2 \right)} + A \right)^2 . \tag{3.11}$$

It is noted that, different from Equation (1.7), only the K-nearest training samples are used to calculate the local generalization error bound in the LFM. Since A is fixed for a given K neighborhood set, they will not affect the value of $R^*_{Q^K}$. Thus, this parameter is ignored and the new function is now defined as:

$$R'_{Q^K}\left(f, X^K, q^{\max}\right) = \sqrt{\frac{1}{K}\sum_i^K err(f, x_i)} + \sqrt{\frac{1}{K}\sum_i^K \underset{Q_i'}{E}\left(\left(\Delta Y\left(f, x_i, q^{\max}\right)\right)^2 \right)} . \tag{3.12}$$

The definitions of *err* and ΔY are the same as in Equation (1.7). They denote the training error and the sensitivity term respectively. The definition of the sensitivity term of R'_{Q^K} is:

$$\frac{1}{K}\sum_i^K \underset{Q_i^K}{E}\left(\left(\Delta Y\left(f, x_i, q^{\max}\right)\right)^2 \right) = \frac{\left(q^{\max}\right)^2}{3K}\sum_i^K \left(\frac{\partial f}{\partial x_i} \right)^T \left(\frac{\partial f}{\partial x_i} \right). \tag{3.13}$$

Throughout this chapter, RBF Network is used for the LFM experiments. The function of RBF Network is defined as:

$$f^{RBF}(x_i)=\sum_{m=1}^{M}\alpha_m \exp\left(-\sum_{j=1}^{n}\left(\frac{x_{i_j}-u_{m_j}}{\sqrt{2}v_{m_j}}\right)^2\right), \tag{3.14}$$

where

M denotes the number of hidden neurons

α_m denotes weight of the m^{th} Gaussian output function

u_m denotes the peak position of the m^{th} center and u_{mj} is the j^{th} feature of u_m

v_m denotes the width of the m^{th} center and v_{mj} is the j^{th} feature of v_m

x_i denotes the sample i and x_{ij} is the j^{th} feature of x_i

The sensitivity term of RBF Network ($\partial f^{RBF}/\partial x_i$ and df^{RBF}/dx_{it}) is defined as:

$$\frac{\partial f^{RBF}}{\partial x_i}=\left[\frac{df^{RBF}}{dx_{i_1}},\frac{df^{RBF}}{dx_{i_2}},\ldots,\frac{df^{RBF}}{dx_{i_n}}\right]^T, \tag{3.15}$$

$$\frac{df^{RBF}}{dx_{i_t}}=-\sum_{m=1}^{M}\frac{\alpha_m\left(x_{i_t}-u_{m_t}\right)}{v_{m_t}^2}\exp\left(-\sum_{j=1}^{n}\left(\frac{x_{i_j}-u_{m_j}}{\sqrt{2}v_{m_j}}\right)^2\right). \tag{3.16}$$

A base classifier with smaller local generalization error bound is more preferred. Thus, the weights for the base classifier could be defined as the inverse of R'_{Q^k}. If necessary, the weights can be normalized to [0,1] before used. Finally, the outputs of the base classifiers are combined using weighted averaging method. The class having the largest value is assigned to the testing sample.

Before classifying a testing sample, L base classifiers should be trained using the given training set. In MCSs, each base classifier must be different from the others. Otherwise it is not necessary to combine the same base classifiers. Thus, the objective of a MCS construction method is to build a set of diverse base classifiers. There are many ways to construct a MCS, e.g. Bagging and Boosting. The LFM can be applied to any of these methods. The only requirement of the LFM is the sensitivity term which must be defined and calculated for a base classifier. In this chapter, the sensitivity term is defined as a differentiation of the base classifier function. Hence, any differentiable base classifier, e.g. MLP Neural Network, RBF Network and SVM, may be used.

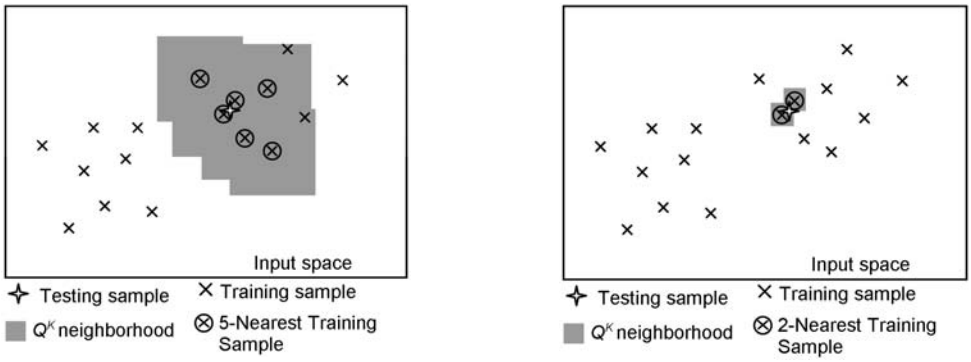
3.2.1 Time complexity

Time complexity is a concern in the dynamic fusion methods. LFM assigns the weights to base classifiers according to the value of R'_{Q^κ} for each testing sample. In general, the time required for classifying a sample is more critical than the training time. This is because the training process can be completed off-line while classification is a real time task. To reduce the time complexity of classifying a sample, any computation independent from testing samples can be done during the training process. For the LFM, the training error and $\partial f/\partial x_i$ of the sensitivity term (shown in Equation (3.13)) of the base classifiers are calculated and stored during the training of the base classifiers. Let N denote the number of training samples, n be the number of features, and M be the number of hidden neurons. Assuming the RBF Network is used, the time of calculating the training error and $\partial f/\partial x_i$ for a training set is $O(2nNM)$, in comparison with $O(NMn)$, which is the time required for training error calculation only.

When classifying a sample, similar to most other dynamic fusion methods, the distance between the testing sample and each training sample is calculated. The value of q^{max}

is equal to the largest distance between the K nearest training samples and the testing samples. R'_{Q^K} is calculated by using the value of q^{max} and retrieving the training error and $\partial f/\partial x_i$ of K nearest training samples. The time complexity is O(KMn) , which is the same as other dynamic fusion methods.

3.2.2 Size of nearest neighborhood (*K*)



a) 5-Nearest Training Sample b) 2-Nearest Training Sample

Figure 3.4 Q neighborhood for a testing sample with different K

An important parameter of LFM is the size of K-nearest neighborhood. The value of K determines the number of training samples used to evaluate the local competence of a base classifier. It also determines the size of the local neighborhood (Q^K). Figure (3.4) shows the effect of K on Q^K for the same dataset. The size of Q^K is a non-decreasing function of K. When K is large, more training samples are included and bigger region is covered. However, the training samples may be located in a region far away from the testing sample. Figure (3.4a) shows an example of this situation. This may result in an estimation which may not represent the local competence of the base classifiers. On the other hand, while K is small, most samples in X^K are near to the testing sample. However, only a small region will be considered. In this case, the estimation may be inaccurate due to lack of information. Figure (3.4b) shows the size of Q^K when only two training samples are used. The small Q^K neighborhood region may provide very little information for a good estimation. As a result,

the value of K cannot be too small. Experimental discussion on K is presented in Section 3.3.1.

3.2.3 Why does LFM work?

Only measuring the performance on the training samples (or validation samples) and the distance between testing sample and training sample (or validation samples) may not suffice to represent the local competence of a base classifier. A simple example illustrates this observation (Figure (3.5)).

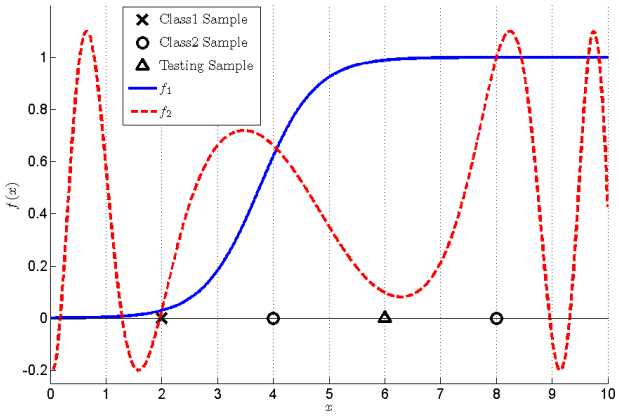


Figure 3.5 Function of two classifiers on a simple artificial data points

Figure (3.5) shows a simple 2-class classification problem. Each sample has only one feature. Class 1 has only one, which is $x = 2$, while class 2 contains two samples, which are $x = 4$ and 8. An unseen sample ($x = 6$) is considered for classification. Two classifiers (f_1 and f_2) are trained and combined as a MCS for this classification problem. When the value of the classifier's output is bigger than 0.5, the sample is labeled as class 2, otherwise, it is class 1. The output of classifier 1 (f_1) is represented by a solid line in the figure and is stable. A dotted line denotes classifier 2 (f_1) and its output is fluctuating.

The output of a classifier is the estimation of the posteriori probability ($p(\omega_c|x)$), which is the probability of the sample belongs to class c given that feature value x has been

measured [Duda et al 2000]. It is reasonable to expect that the posteriori probability of two similar inputs should not be very different. From another point of view, noises may appear in samples. The actual value of a sample may be slightly different from the one being collected. The performance of a classifier is considered not good if the outputs of two similar inputs are very different. However, if the output of a base classifier is fluctuating in the local region of the test sample, the base classifier is not expected to recognize that samples correctly. This is because the base classifier changes so frequently. As a result, if the training error is ignored, a smooth classifier is more preferable in general. A smooth classifier should be assigned a larger weight in a MCS.

The current dynamic fusion methods only consider the error of the training (or validation) samples and the distance between the testing sample and the training (or validation) samples. In this example, since the outputs of both classifiers on the training samples are the same, i.e., $f_1(2) = f_2(2)$, $f_1(4) = f_2(4)$ and $f_1(8) = f_2(8)$, the errors of both classifiers on each training sample are the same. Hence, the classifiers are assigned the same weight when classifying the testing sample ($x = 6$) by using the current dynamic fusion methods. Different from the current dynamic fusion methods, the LFM uses not only the information on the training samples but also the unseen samples in a local region of the testing sample. It measures the generalization error bound of the local area where the testing sample located. The generalization error bound of the LFM contains both the training accuracy term and the sensitivity term which measures the smoothness of the classifier output. In this example, as the output of f_2 changes rapidly comparing with f_1, the sensitivity term of classifier 2 is much larger than that for classifier 1. Thus, a small weight is assigned to classifier 2. The LFM awards a classifier which has stable outputs in the local region. This explains why a MCS using the LFM may achieve a better performance than the one

using recent dynamic fusion methods. The experimental results of using the LFM and other fusion methods will be discussed and analyzed in the next section.

3.3 Experiments

In this section, the performance of the LFM is evaluated experimentally. Twenty datasets shown in Table (3.2) from the UCI machine learning repository [MLR] and Intelligent Data Analysis Group [DAG] were used. They cover a wide range of applications involving two-class and multi-class problems. Each dataset is equally divided into three parts: training (35%), validation (15%) and testing (50%) set. Thirty independent runs were generated for each dataset. Only samples in the training set are used during training. Some dynamic fusion methods require a validation set in classifying samples. The testing set is reserved to evaluate the performance of the trained MCSs. The inputs of all samples are normalized to [0, 1] to eliminate the effect of a large range of values.

Table 3.2 Twenty Datasets

Dataset	Short Name	# Class	# Sample	# Feature
Breast Cancer Wisconsin	Canc	2	569	32
Car Evaluation	Car	4	1728	6
Connectionist	Conn	2	208	60
Credit Approval	Cred	2	690	15
Dermatology	Derm	6	366	34
Pima Indians Diabetes	Pima	2	768	8
Solar Flare	Solar	2	1066	9
German Credit Data	Germ	2	1000	24
Glass Identification	Glass	7	214	10
Heart	Heart	2	270	13
Hepatitis	Hepa	2	80	19
Ionosphere	Iono	2	351	33
Iris	Iris	3	150	4
Multiple Features	Feat	10	2000	649
Image Segmentation	Img	7	2310	19
Spambase	Spam	2	4601	57
Thyroid	Thy	2	215	5
Tic-Tac-Toe Endgame	TTT	2	958	9
Waveform	Wave	3	5000	21
Wine	Wine	3	178	13

RBF Network is used as the base classifiers. The number of neurons of RBF Network is also selected randomly from two to fifty. The center and width of the neuron is determined by K-mean [Kiernan et al 1996] and the K-nearest-neighbor algorithm [Musavi et al 1992] respectively. The weight is calculated using Singular Value Decomposition (SVD) method [Mak et al 1998]. To diversify a set of base classifiers in a MCS, Bagging method [Breiman 1996] is employed. Each base classifier is assigned a different training set randomly selected from the original training set with replacement.

The effect of K-Nearest neighborhood size of the LFM is discussed in Section 3.3.1. In Section 3.3.2, the performance of the LFM and the other twenty one fusion methods are compared and analyzed using different number of base classifiers.

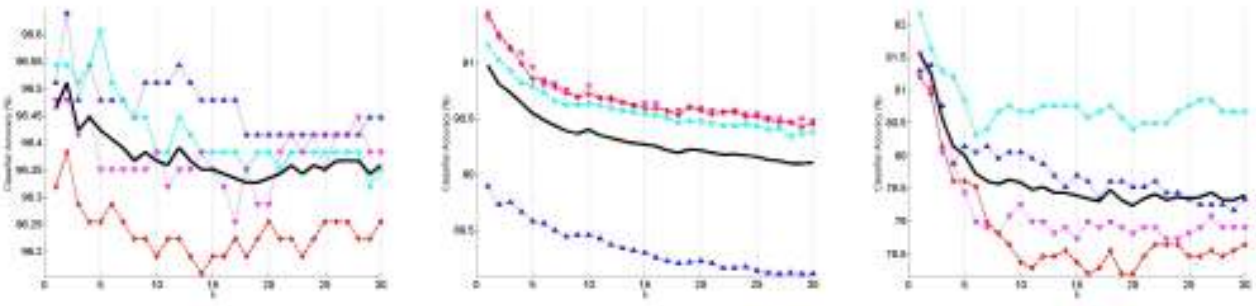
3.3.1 Effect of *K*-Nearest neighborhood size

The effect of the parameter K on the performance of LFM is discussed experimentally in this section. Different sizes of MCSs (L = 5, 10, 20 and 30) using LFM with K from 1 to 30 are evaluated for each dataset. Figure (3.6) shows the effect of K on the testing accuracies of the MCSs. The line represents the average testing accuracy of MCSs on thirty independent runs on different datasets. The dotted lines denote the performance of MCSs with different number of individual classifiers while the bold solid black line is the average of these MCSs. X-axis and Y-axis represent the value of K and the testing classification accuracy respectively.

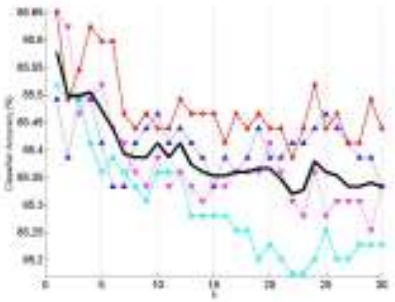
The dotted lines have similar shapes in most datasets, especially for Car, Conn, Spam, TTT and Wave, in Figure (3.6). This indicates that the value of K has similar effect on MCSs with different number of classifiers. Also, combining more base classifiers may degrade the performance of MCSs. For example, in the Connectionist dataset, MCSs with

10 base classifiers is the most accurate while the performance of the MCSs using 30 base classifiers is the worst.

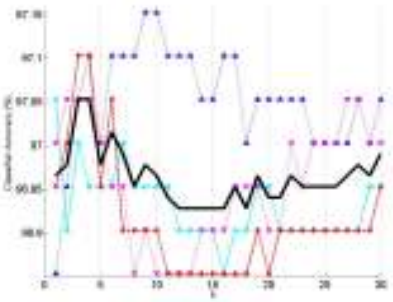
In most datasets, the performance of MCSs decreases when the value of K increases. This indicates that the local region (Q^K) composed by larger number of nearest samples cannot reflect the actual local performance of individual base classifiers. The performance of the LFM drops when K increases. On the other side, MCSs with the LFM perform better while K is smaller. However, the smallest K, when K=1, does not guarantee having the best MCS. The results in dermatology, Solar Flare, German Credit Data, Thyroid and Tic-Tac-Toe Endgame show that the smallest value is not necessarily the best choice for K. This observation from the experimental results agrees with the discussion in Section 3.2.2. Moreover, in most datasets, with the exception of Multiple Features, Thyroid and Pima Indians Diabetes, the LFM performs the best when K is between 1 and 5. In short, since K determines the size of the local region surrounding the testing sample, it affects the performance of the LFM. According to the above experimental results, the range of 1 to 5 may be a reasonable choice for K in general.



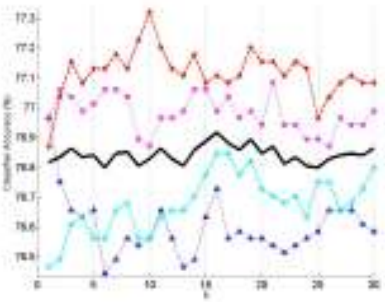
a) Canc b) Car c) Conn



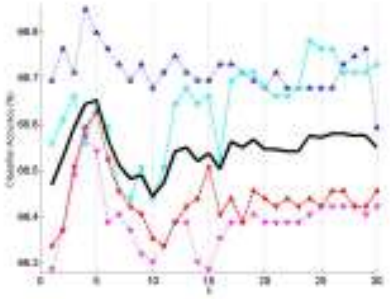
d) Cred



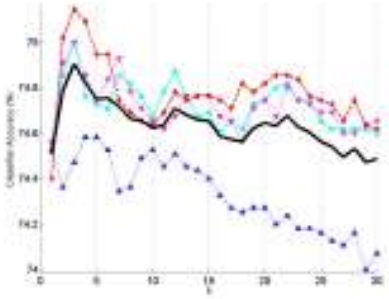
e) Derm



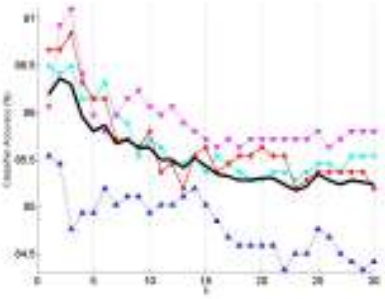
f) Pima



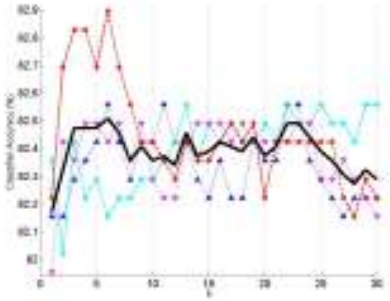
g) Solar



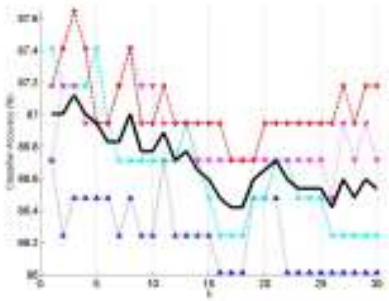
h) Germ



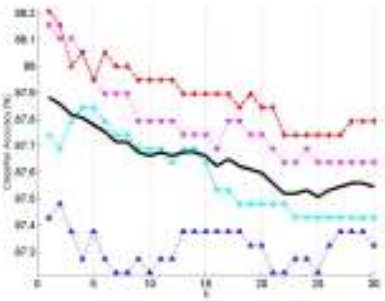
i) Glass



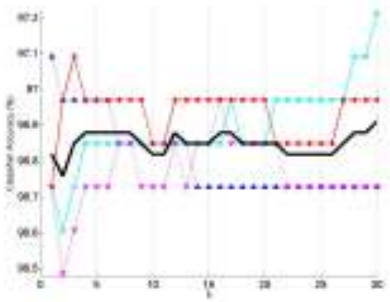
j) Heart



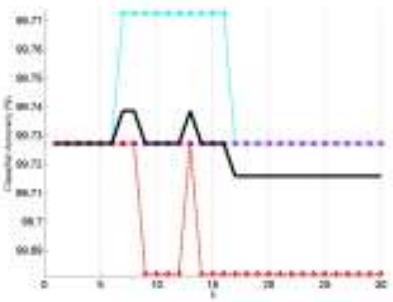
k) Hepa



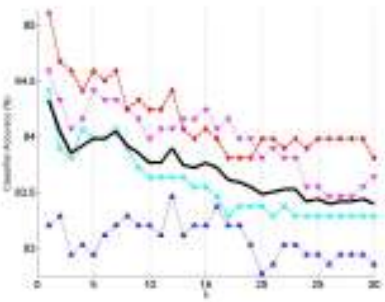
l) Iono



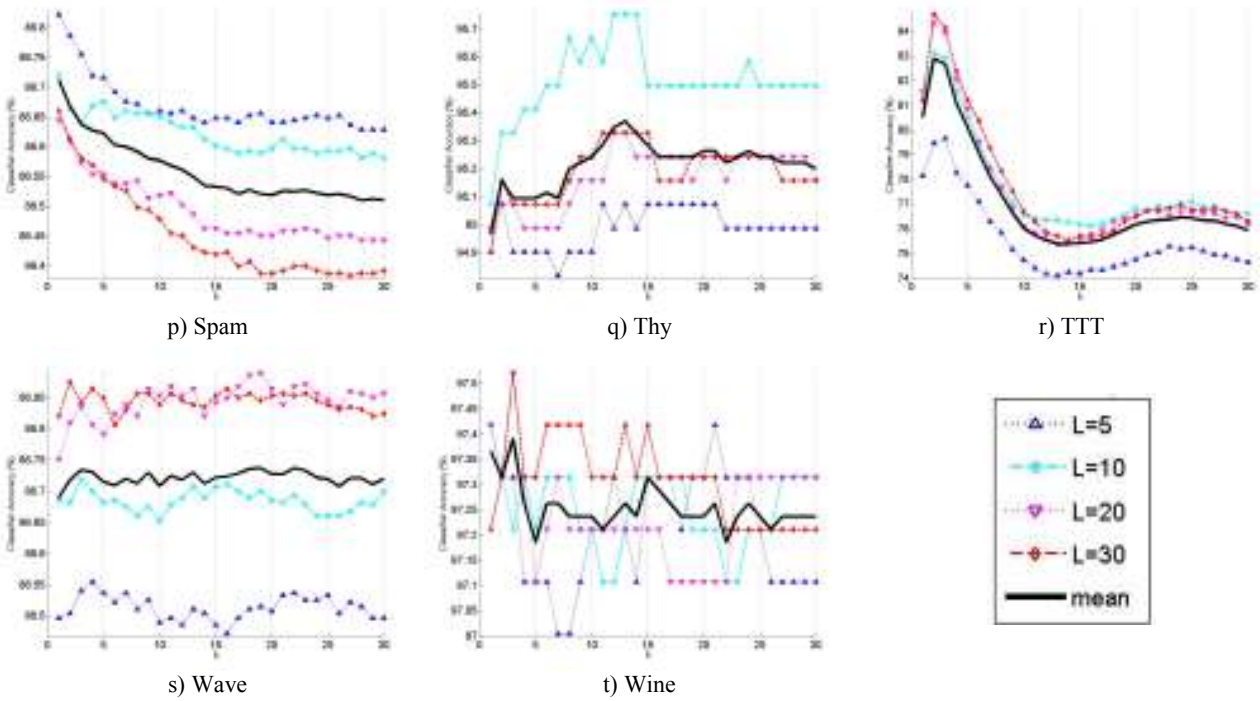
m) Iris



n) Feat



o) Img



p) Spam q) Thy r) TTT

s) Wave t) Wine

Figure 3.6 Classification Accuracy of LFM with Different Number of Nearest Neighborhoods over Thirty Independent Runs

3.3.2 Performance comparison between LFM and other methods

In this section, LFM is compared with twenty one well known dynamic fusion methods mentioned in Section 3.1: Dynamic Selection (DS), Dynamic Voting (DV), Dynamic Voting with Selection (DVS), *K*-Nearest-Oracles Union (KU), weighted *K*-Nearest-Oracles Union (W-KU), *K*-Nearest-Oracles Eliminate (KE), weighted *K*-Nearest-Oracles Eliminate (W-KE), Confidence Measure Method (CM), Mahalanobis Distance method (MD), the six methods using different matrixes shown in Table (3.1) (C1 – C6), Mixture of Experts (ME), three varies of Dynamic Integration (LC1 – LC3), a priori (Pri) and a posteriori (Post) method. The best value of *K* for LFM, DS, DV, DVS, KU, W-KU, KE, W-KE, Pri and Post methods are selected using cross-validation. MLP Neural Network is acted as the Gating Network in ME and LC1 – LC3 methods. The summary of these 21 dynamic fusion methods are shown in Table (3.3). If the dynamic fusion methods consider

the accuracy of a classifier or distance between the unseen sample and training samples, a "✓" is shown in the columns respectively.

Table 3.3 Summary of Dynamic Fusion Methods Comparied in experiments

Name	Acronym	Accuracy	Distance	References
Dynamic Selection	DS	✓		Puuronen et al 1999, Tsymbal et al 1998
Dynamic Voting	DV	✓	✓	
Dynamic Voting with Selection	DVS	✓	✓	
K-Nearest-Oracles Union	KU	✓		Ko et al 2008
Weighted K-Nearest-Oracles Union	W-KU	✓	✓	
K-Nearest-Oracles Eliminate	KE	✓		
Weighted K-Nearest-Oracles Eliminate	W-KE	✓	✓	
Confidence Measure Method	CM			Abraham et al 2008
Mahalanobis Distance	MD	✓	✓	Polikar et al 2003
Dynamic Method using different functions by Woloszynski	C1 – C6	✓	✓	Woloszynski et al 2006
Mixture of Experts	ME	✓	✓	Jacobs et al 1991
Dynamic Integration	LC1 – LC6	✓	✓	Kim et al 2005
Priori Probability	Pri	✓		Ho et al 1994, Giacinto et al 1999
Posteriori Probability	Post	✓		

The dynamic fusion methods are used to combine the same set of base classifiers to form the MCSs. The only difference is the weight of each base classifier. The performances of different MCSs (L = 5, 10, 20 and 30) are evaluated.

In Table (3.4), the Win-Tie-Loss gives the number of datasets for which the MCS with LFM performs better/same/worse in comparison with the one by using other fusion methods. For example, 15-1-4 is shown in the first cell. It means out of 20 datasets, the MCS using LFM performs better in 15, the same in 1 and worse in 4 datasets comparing with the one by using DV on the average of 30 independent runs. Each column represents a MCS with a different number of base classifiers. Each row represents a fusion method. Let T denote the number of datasets. If the number of wins is bigger than or equal to $T/2+1.96\sqrt{T}/2$, then the LFM claims to perform significantly better at the 95% probability level. In this experiment, as T = 20, the LFM performs significantly better at the 95% probability level when it wins more than 13 datasets. The value is bolded and

underlined in the cell if the performance of the MCS with LFM is better significantly. The last row gives the average of figures in each column.

The table shows the MCS with LFM outperforms other methods in most datasets and wins more than 17 datasets on the average. LFM is significantly better than all other fusion methods for all sizes of MCSs. Especially comparing with DS, DVS, CM, MD, C1, C2, C3, C4 and C5, the MCS with LFM is more accurate in more than 17 datasets. It is noted that LFM even has better performances than C3 in all datasets for all sizes of MCSs. The performance of the MCS with W-KU and W-KE is the closest to the one with LFM. But even in these cases, LFM still wins 15.25 and loses 4.75 on the average for all sizes of MCSs. From the average value of Win-Tie-Loss, there is no significant difference between the MCSs using different number of base classifiers. LFM is consistently better than other fusion methods for about 17.18 datasets, and only worse in for about 2.45 datasets.

Table 3.4 LFM VS Other Fusion Methods
Win-Tie-Loss Comparison over Twenty Datasets

	$L = 5$	$L = 10$	$L = 20$	$L = 30$
DS	**20-0-0**	**19-0-1**	**20-0-0**	**20-0-0**
DV	**15-1-4**	**13-1-6**	**16-1-3**	**17-0-3**
DVS	**19-0-1**	**19-0-1**	**17-0-3**	**19-0-1**
KU	**17-1-2**	**18-0-2**	**16-1-3**	**14-2-4**
W-KU	**15-0-5**	**15-0-5**	**16-0-4**	**15-0-5**
KE	**17-0-3**	**18-0-2**	**16-0-4**	**14-0-6**
W-KE	**15-0-5**	**15-0-5**	**17-1-2**	**18-0-2**
CM	**19-0-1**	**17-0-3**	**18-1-1**	**19-0-1**
MD	**19-0-1**	**18-0-2**	**17-1-2**	**18-0-2**
C1	**20-0-0**	**20-0-0**	**18-2-0**	**20-0-0**
C2	**19-0-1**	**18-0-2**	**18-0-2**	**18-0-2**
C3	**20-0-0**	**20-0-0**	**20-0-0**	**20-0-0**
C4	**19-0-1**	**18-0-2**	**18-0-2**	**18-0-2**
C5	**20-0-0**	**19-0-1**	**17-2-1**	**20-0-0**
C6	**18-0-2**	**17-0-3**	**16-2-2**	**18-0-2**
ME	**17-1-2**	**18-0-2**	**16-1-3**	**18-0-2**
LC1	**18-0-2**	**17-0-3**	**17-1-2**	**18-0-2**
LC2	**16-1-3**	**18-0-2**	**17-1-2**	**16-1-3**
LC3	**17-0-3**	**16-0-4**	**17-1-2**	**18-0-2**
Pri	**15-0-5**	**15-0-5**	**17-0-3**	**16-0-4**
Post	**15-0-5**	**14-0-6**	**15-0-5**	**16-0-4**
Average	**17.5-0.3-2.3**	**17.2-0.1-2.8**	**17.0-0.8-2.3**	**17.5-0.2-2.4**

The average percentage of classification accuracy of the testing sets of MCSs using different fusion methods over 30 independent runs are shown in Tables (3.5) – (3.8). Each table shows the performance of a MCS with 5, 10, 20 or 30 base classifiers. A column represents a fusion method while a dataset is represented by a row. The first value and second value in a cell are the average classification testing accuracy and the variance respectively. The Student's t-test is applied to examine the statistical significance of the improvement made by LFM. When the absolute t-value is larger than 2.00 in each experiment, a difference between two means is significant at the 95% probability level. The value is bolded and underlined in the cell if the performance of MCSs with LFM is significantly better than the one using other methods.

Generally, the MCSs with all fusion methods are slightly more accurate when the number of base classifiers increases. The performance of the MCS increases about 0.6% on average when the MCS has 30 base classifiers comparing with having 5 base classifiers. Especially for Tic-Tac-Toe Endgame, the testing accuracy of MCSs using LFM is 79.65 when 5 base classifiers are used. When the number of base classifiers increases to 10, 20 and 30, the performances of the MCSs are 83.13, 84.30 and 84.68 respectively. MCSs using other dynamic fusion methods also improve by 4.5% for the Tic-Tac-Toe Endgame dataset when more base classifiers are used.

The MCS using LFM has better testing accuracy than other fusion methods in nearly all cases. In most cases the improvement is 95% significant. For Breast Cancer Wisconsin, Dermatology, Solar Flare and Tic-Tac-Toe Endgame dataset, LFM performs significantly better than most other dynamic fusion methods. Moreover, the performance of LFM is stable. The variance of the testing accuracy is 0.03 on average and it is the smallest among all dynamic fusion methods. The variance of DVS is 0.05 which is the largest. The LFM has also demonstrated good performances consistently among different datasets. On the contrast,

the performance of DS, W-KU, W-KE and Pri is fluctuated. For example, K-KU has great performance in Breast Cancer Wisconsin while performing badly in Solar Flare.

Table 3.5 LFM VS Other Fusion Methods
Average Classification Accuracy and Variance of Testing Set of MCSs with 5 base classifiers over Thirty Independent Runs

L=5	LFM	DS	DV	DVS	KU	W-KU	KE	W-KE	CM	MD	C1	C2	C3	C4	C5	C6	ME	LC1	LC2	LC3	Pri	Post
Canc	96.64 ±0.00	**96.16 ±0.00**	**96.25 ±0.00**	**96.09 ±0.00**	**95.84 ±0.00**	**96.37 ±0.00**	**95.84 ±0.00**	**96.38 ±0.00**	**96.19 ±0.00**	**96.16 ±0.00**	**96.19 ±0.00**	**96.16 ±0.00**	**96.09 ±0.00**	**96.09 ±0.00**	**96.16 ±0.00**	**96.13 ±0.00**	**96.19 ±0.00**	**95.36 ±0.00**	**96.22 ±0.00**	**95.93 ±0.00**	**96.27 ±0.00**	**95.99 ±0.00**
Car	89.90 ±0.01	**88.64 ±0.01**	89.96 ±0.01	**86.76 ±0.02**	90.03 ±0.01	91.10 ±0.01	90.10 ±0.01	90.10 ±0.01	**89.19 ±0.01**	**89.12 ±0.01**	**89.10 ±0.01**	**88.77 ±0.01**	**88.89 ±0.01**	**88.88 ±0.01**	**89.29 ±0.01**	**89.28 ±0.01**	**89.41 ±0.01**	**87.24 ±0.02**	**89.38 ±0.01**	89.99 ±0.01	91.00 ±0.01	90.54 ±0.01
Conn	81.38 ±0.09	**78.70 ±0.09**	80.32 ±0.10	**76.08 ±0.08**	83.50 ±0.06	82.41 ±0.08	83.85 ±0.06	80.14 ±0.10	**79.17 ±0.09**	**79.08 ±0.07**	**78.82 ±0.08**	**75.64 ±0.11**	**74.85 ±0.28**	**76.96 ±0.13**	**79.26 ±0.09**	**79.35 ±0.09**	80.58 ±0.12	**78.82 ±0.13**	80.32 ±0.12	**79.79 ±0.17**	82.31 ±0.08	81.61 ±0.07
Cred	85.49 ±0.01	85.23 ±0.01	85.20 ±0.01	85.31 ±0.01	**81.77 ±0.01**	**85.03 ±0.01**	**81.91 ±0.01**	85.23 ±0.02	85.07 ±0.01	85.15 ±0.01	85.15 ±0.01	**84.41 ±0.01**	**84.70 ±0.01**	**84.73 ±0.01**	85.15 ±0.01	85.17 ±0.01	85.02 ±0.01	**85.02 ±0.01**	85.17 ±0.02	85.33 ±0.02	**84.93 ±0.01**	**84.69 ±0.01**
Derm	97.15 ±0.00	**96.47 ±0.02**	96.90 ±0.01	**95.70 ±0.06**	**96.75 ±0.01**	**96.70 ±0.00**	**96.71 ±0.01**	96.90 ±0.00	96.90 ±0.00	**96.85 ±0.00**	**96.85 ±0.00**	**92.36 ±0.14**	**96.65 ±0.01**	**95.25 ±0.02**	97.00 ±0.00	97.00 ±0.00	**96.25 ±0.01**	**91.56 ±0.12**	**96.10 ±0.01**	96.90 ±0.01	**96.60 ±0.00**	**96.80 ±0.00**
Pima	76.96 ±0.01	**76.31 ±0.01**	76.70 ±0.01	**75.73 ±0.01**	**68.96 ±0.02**	**76.29 ±0.01**	**69.10 ±0.02**	76.59 ±0.01	76.52 ±0.01	76.52 ±0.01	76.56 ±0.01	**76.44 ±0.01**	**76.37 ±0.01**	**76.42 ±0.01**	76.56 ±0.01	76.59 ±0.01	**76.44 ±0.01**	**74.64 ±0.01**	**76.37 ±0.01**	**76.30 ±0.01**	**76.19 ±0.01**	**76.10 ±0.01**
Solar	66.85 ±0.01	**66.35 ±0.01**	**66.23 ±0.01**	**66.40 ±0.01**	**55.25 ±0.00**	**56.98 ±0.05**	**56.24 ±0.00**	**62.66 ±0.14**	66.44 ±0.01	66.44 ±0.01	**55.45 ±0.00**	**66.05 ±0.01**	**55.40 ±0.00**	**65.38 ±0.02**	**55.38 ±0.00**	**66.39 ±0.01**	**66.42 ±0.01**	**66.18 ±0.01**	**66.27 ±0.01**	**65.93 ±0.01**	**56.88 ±0.05**	**66.14 ±0.01**
Germ	74.58 ±0.02	73.99 ±0.02	74.27 ±0.03	**73.78 ±0.02**	**72.85 ±0.01**	74.04 ±0.02	**72.59 ±0.01**	74.16 ±0.02	74.05 ±0.02	**73.91 ±0.02**	74.09 ±0.02	**73.82 ±0.02**	74.07 ±0.02	74.02 ±0.02	74.04 ±0.02	**73.93 ±0.02**	**73.89 ±0.02**	**72.24 ±0.02**	74.38 ±0.02	**73.80 ±0.02**	**73.94 ±0.02**	**73.59 ±0.02**
Glass	85.54 ±0.11	84.27 ±0.10	85.45 ±0.11	**82.51 ±0.07**	84.76 ±0.18	86.55 ±0.12	84.61 ±0.18	86.41 ±0.11	84.68 ±0.10	84.68 ±0.12	84.68 ±0.11	84.68 ±0.09	**82.34 ±0.06**	**81.82 ±0.12**	85.37 ±0.15	85.45 ±0.15	85.71 ±0.14	**77.40 ±0.16**	85.19 ±0.15	84.85 ±0.08	86.45 ±0.12	86.19 ±0.12
Heart	82.56 ±0.03	82.09 ±0.05	82.22 ±0.05	82.02 ±0.06	**77.85 ±0.03**	81.75 ±0.04	**77.49 ±0.03**	82.02 ±0.05	81.95 ±0.05	82.09 ±0.05	81.95 ±0.05	**80.47 ±0.07**	**81.14 ±0.04**	**80.74 ±0.05**	81.95 ±0.05	81.89 ±0.05	81.75 ±0.04	**80.13 ±0.02**	81.68 ±0.05	**80.74 ±0.05**	**81.65 ±0.04**	**81.52 ±0.05**
Hepa	86.71 ±0.04	86.09 ±0.11	86.01 ±0.10	86.25 ±0.11	86.01 ±0.07	85.81 ±0.10	85.96 ±0.07	86.01 ±0.10	**86.01 ±0.10**	86.01 ±0.10	85.78 ±0.09	86.01 ±0.10	86.01 ±0.10	86.01 ±0.10	85.78 ±0.09	85.78 ±0.09	**84.62 ±0.10**	**85.55 ±0.11**	**85.55 ±0.09**	86.01 ±0.08	85.71 ±0.10	**85.48 ±0.09**
Iono	87.48 ±0.01	87.41 ±0.01	87.58 ±0.01	87.27 ±0.01	**84.83 ±0.01**	87.38 ±0.01	**84.63 ±0.01**	87.53 ±0.01	87.01 ±0.01	87.43 ±0.01	**87.38 ±0.01**	87.27 ±0.01	**87.01 ±0.01**	**87.32 ±0.01**	**87.32 ±0.01**	87.69 ±0.01	**86.44 ±0.02**	**85.25 ±0.02**	**86.18 ±0.01**	**86.65 ±0.02**	87.28 ±0.01	87.49 ±0.01
Iris	97.09 ±0.01	96.77 ±0.01	96.73 ±0.01	96.85 ±0.01	**95.52 ±0.02**	**96.53 ±0.01**	**95.56 ±0.02**	96.73 ±0.01	96.85 ±0.01	96.85 ±0.01	96.12 ±0.01	**96.73 ±0.01**	**95.39 ±0.02**	**96.61 ±0.01**	96.12 ±0.01	96.73 ±0.01	**96.61 ±0.01**	**96.61 ±0.02**	96.73 ±0.01	96.73 ±0.01	**96.43 ±0.01**	**96.43 ±0.01**
Feat	99.73 ±0.00	99.68 ±0.00	99.73 ±0.00	**99.64 ±0.00**	99.73 ±0.00	**99.53 ±0.00**	99.73 ±0.00	99.68 ±0.00	99.68 ±0.00	99.68 ±0.00	99.68 ±0.00	99.64 ±0.00	**99.64 ±0.00**	99.59 ±0.00	99.68 ±0.00	99.68 ±0.00	**99.64 ±0.00**	99.73 ±0.00	99.68 ±0.00	99.73 ±0.00	**99.43 ±0.00**	**99.43 ±0.00**
Img	83.46 ±0.12	**81.36 ±0.27**	83.38 ±0.13	**77.66 ±0.53**	**80.09 ±0.07**	83.18 ±0.14	**80.57 ±0.07**	82.77 ±0.14	82.86 ±0.13	82.42 ±0.14	82.77 ±0.14	82.42 ±0.14	**80.17 ±0.17**	82.94 ±0.15	82.86 ±0.15	82.42 ±0.16	83.12 ±0.10	**70.65 ±0.32**	83.03 ±0.11	**81.13 ±0.15**	83.08 ±0.14	82.99 ±0.12
Spam	86.82 ±0.01	**86.44 ±0.01**	86.90 ±0.01	**85.74 ±0.01**	**85.92 ±0.01**	87.39 ±0.01	**85.79 ±0.01**	87.19 ±0.01	86.64 ±0.01	86.67 ±0.01	**84.31 ±0.01**	86.65 ±0.01	**84.21 ±0.01**	86.64 ±0.01	**84.33 ±0.01**	86.67 ±0.01	86.87 ±0.01	87.09 ±0.01	86.96 ±0.01	87.21 ±0.01	87.29 ±0.01	87.06 ±0.01
Thy	95.07 ±0.01	94.62 ±0.02	94.82 ±0.02	**94.22 ±0.02**	94.82 ±0.02	94.62 ±0.02	94.88 ±0.02	94.90 ±0.02	94.65 ±0.02	94.65 ±0.02	94.73 ±0.02	**94.48 ±0.02**	94.56 ±0.02	94.65 ±0.02	94.82 ±0.02	94.73 ±0.02	94.73 ±0.02	**94.05 ±0.02**	**94.05 ±0.02**	**94.14 ±0.02**	94.52 ±0.02	**94.43 ±0.02**
TTT	79.65 ±0.06	**75.17 ±0.04**	78.99 ±0.04	**70.77 ±0.05**	**75.37 ±0.02**	79.93 ±0.02	**75.80 ±0.03**	**77.98 ±0.04**	**75.50 ±0.04**	**75.61 ±0.04**	**75.48 ±0.04**	**74.55 ±0.03**	**73.68 ±0.03**	**73.64 ±0.03**	**76.41 ±0.05**	**76.43 ±0.05**	**77.87 ±0.05**	**75.69 ±0.04**	**77.17 ±0.05**	**77.51 ±0.05**	79.83 ±0.02	**77.10 ±0.02**
Wave	86.55 ±0.00	**86.33 ±0.00**	86.53 ±0.00	**85.95 ±0.00**	**79.79 ±0.00**	**86.38 ±0.00**	**80.17 ±0.00**	86.58 ±0.00	86.50 ±0.00	86.47 ±0.00	86.47 ±0.00	86.55 ±0.00	86.51 ±0.00	86.50 ±0.00	86.47 ±0.00	86.47 ±0.00	86.49 ±0.00	**85.26 ±0.00**	86.51 ±0.00	86.53 ±0.00	**86.28 ±0.00**	**86.22 ±0.00**
Wine	97.42 ±0.02	**96.76 ±0.02**	97.11 ±0.02	**96.18 ±0.02**	**96.59 ±0.02**	**96.80 ±0.02**	**96.56 ±0.02**	97.00 ±0.02	**96.80 ±0.02**	97.00 ±0.02	97.00 ±0.02	96.90 ±0.02	97.00 ±0.02	97.00 ±0.02	97.00 ±0.02	97.00 ±0.02	**96.59 ±0.02**	**95.76 ±0.03**	**95.97 ±0.04**	97.00 ±0.01	**96.70 ±0.02**	**96.70 ±0.02**

Table 3.6 LFM VS Other Fusion Methods
Average Classification Accuracy and Variance of Testing Set of MCSs with 10 base classifiers over Thirty Independent Runs

L=10	LFM	DS	DV	DVS	KU	W-KU	KE	W-KE	CM	MD	C1	C2	C3	C4	C5	C6	ME	LC1	LC2	LC3	Pri	Post
Canc	96.61 ±0.00	**96.02 ±0.00**	**96.09 ±0.00**	**95.90 ±0.00**	**95.84 ±0.00**	**96.21 ±0.00**	**95.86 ±0.00**	**96.22 ±0.00**	**96.06 ±0.00**	**96.06 ±0.00**	**96.06 ±0.00**	**95.97 ±0.00**	**96.06 ±0.00**	**95.97 ±0.00**	**96.06 ±0.00**	**96.06 ±0.00**	**96.06 ±0.00**	**95.74 ±0.00**	**96.06 ±0.00**	**95.77 ±0.00**	**96.11 ±0.00**	**95.89 ±0.00**
Car	91.16 ±0.00	**89.52 ±0.00**	91.19 ±0.00	**86.95 ±0.01**	92.05 ±0.00	92.16 ±0.00	91.92 ±0.00	91.20 ±0.00	**90.28 ±0.00**	**90.30 ±0.00**	**90.39 ±0.00**	**90.01 ±0.00**	**90.16 ±0.00**	**90.19 ±0.00**	**90.46 ±0.00**	**90.42 ±0.00**	**90.56 ±0.00**	**89.50 ±0.01**	**90.61 ±0.00**	91.02 ±0.00	92.06 ±0.00	91.63 ±0.00
Conn	82.17 ±0.09	**79.41 ±0.07**	81.82 ±0.04	**75.82 ±0.12**	84.29 ±0.08	82.94 ±0.06	83.98 ±0.08	81.38 ±0.07	**80.67 ±0.05**	**79.96 ±0.06**	**80.05 ±0.06**	**74.67 ±0.10**	**75.64 ±0.16**	**75.73 ±0.14**	**80.32 ±0.06**	**80.32 ±0.06**	**81.02 ±0.06**	**80.32 ±0.07**	81.64 ±0.06	**80.67 ±0.05**	82.84 ±0.06	82.40 ±0.05
Cred	85.52 ±0.02	85.30 ±0.01	85.36 ±0.02	85.20 ±0.01	**81.95 ±0.01**	**85.08 ±0.01**	**82.21 ±0.01**	85.31 ±0.02	**85.25 ±0.02**	**85.41 ±0.01**	**85.39 ±0.01**	**84.78 ±0.02**	**84.88 ±0.02**	**84.80 ±0.01**	85.36 ±0.01	85.36 ±0.01	85.44 ±0.02	**84.67 ±0.01**	85.25 ±0.02	85.36 ±0.01	**84.98 ±0.01**	**84.85 ±0.01**
Derm	97.05 ±0.00	96.95 ±0.00	97.05 ±0.00	96.85 ±0.00	**96.60 ±0.01**	**96.80 ±0.00**	**96.67 ±0.01**	97.00 ±0.00	96.95 ±0.00	96.95 ±0.00	96.95 ±0.00	**92.36 ±0.10**	96.90 ±0.00	**95.45 ±0.02**	96.95 ±0.00	96.95 ±0.00	96.85 ±0.00	**93.21 ±0.03**	**96.15 ±0.01**	96.90 ±0.01	**96.70 ±0.00**	**96.75 ±0.00**
Pima	76.85 ±0.01	**76.14 ±0.02**	76.40 ±0.02	**75.62 ±0.02**	**69.13 ±0.03**	**76.10 ±0.02**	**69.12 ±0.03**	76.47 ±0.01	**76.25 ±0.02**	76.37 ±0.02	76.61 ±0.02	76.33 ±0.02	76.35 ±0.02	**76.30 ±0.02**	76.66 ±0.02	76.66 ±0.02	**76.21 ±0.02**	**75.31 ±0.01**	76.40 ±0.02	**76.18 ±0.02**	**76.00 ±0.02**	**76.31 ±0.02**
Solar	66.78 ±0.01	**66.38 ±0.01**	**66.37 ±0.01**	66.56 ±0.01	**55.26 ±0.00**	**57.10 ±0.04**	**56.25 ±0.00**	**62.66 ±0.14**	66.47 ±0.01	66.46 ±0.01	**55.40 ±0.00**	**66.11 ±0.01**	**55.37 ±0.00**	**65.21 ±0.02**	**55.40 ±0.00**	66.58 ±0.01	**66.20 ±0.01**	**66.08 ±0.00**	**66.13 ±0.01**	**66.06 ±0.01**	**57.00 ±0.04**	**66.33 ±0.00**
Germ	75.00 ±0.03	**74.34 ±0.02**	74.67 ±0.02	**74.11 ±0.01**	**73.55 ±0.01**	74.55 ±0.03	**73.39 ±0.01**	74.45 ±0.02	**74.22 ±0.02**	**74.27 ±0.02**	74.49 ±0.02	**74.05 ±0.02**	**74.22 ±0.02**	**74.20 ±0.02**	74.35 ±0.03	74.35 ±0.03	**74.31 ±0.03**	**73.20 ±0.01**	74.89 ±0.02	74.45 ±0.03	74.45 ±0.03	**74.23 ±0.03**
Glass	86.49 ±0.07	**85.05 ±0.09**	86.93 ±0.09	**81.56 ±0.09**	86.15 ±0.13	86.81 ±0.10	86.25 ±0.13	86.84 ±0.09	86.67 ±0.08	85.97 ±0.09	86.41 ±0.08	85.80 ±0.08	**84.42 ±0.09**	**83.55 ±0.17**	86.58 ±0.09	86.67 ±0.09	86.67 ±0.11	**76.88 ±0.11**	86.58 ±0.10	86.06 ±0.13	86.71 ±0.10	87.06 ±0.10
Heart	82.56 ±0.03	81.86 ±0.04	**81.75 ±0.03**	82.22 ±0.05	**78.32 ±0.02**	81.42 ±0.02	**78.32 ±0.02**	**81.68 ±0.02**	**81.48 ±0.03**	**81.62 ±0.03**	**81.62 ±0.03**	**80.13 ±0.03**	**80.61 ±0.03**	**80.47 ±0.02**	**81.48 ±0.03**	**81.48 ±0.03**	**81.35 ±0.03**	**80.74 ±0.02**	**81.01 ±0.04**	**81.14 ±0.03**	**81.32 ±0.02**	**81.18 ±0.03**
Hepa	87.41 ±0.02	**86.40 ±0.08**	**86.01 ±0.09**	87.18 ±0.06	**84.85 ±0.08**	85.81 ±0.09	**85.13 ±0.08**	**86.01 ±0.09**	**85.78 ±0.09**	**85.78 ±0.07**	**86.01 ±0.09**	**85.55 ±0.07**	**86.01 ±0.09**	**85.78 ±0.07**	**86.01 ±0.09**	**86.01 ±0.09**	**84.62 ±0.08**	**85.55 ±0.09**	**86.01 ±0.09**	**85.55 ±0.07**	**85.71 ±0.09**	**85.71 ±0.07**
Iono	87.84 ±0.00	87.86 ±0.01	88.26 ±0.01	**87.38 ±0.01**	**85.87 ±0.01**	87.96 ±0.01	**85.75 ±0.01**	88.00 ±0.01	87.58 ±0.01	87.90 ±0.01	87.69 ±0.01	87.90 ±0.01	87.58 ±0.01	87.84 ±0.01	87.74 ±0.01	88.05 ±0.01	**86.44 ±0.01**	**86.39 ±0.01**	**87.32 ±0.02**	**86.23 ±0.01**	87.86 ±0.01	88.12 ±0.00
Iris	97.21 ±0.02	**96.61 ±0.02**	**96.61 ±0.02**	96.73 ±0.01	**96.00 ±0.03**	**96.41 ±0.02**	**96.08 ±0.03**	**96.48 ±0.02**	**96.61 ±0.02**	**96.61 ±0.02**	**96.00 ±0.02**	**96.48 ±0.02**	**95.64 ±0.03**	**96.48 ±0.02**	**95.88 ±0.02**	**96.48 ±0.02**	**96.36 ±0.02**	**96.61 ±0.02**	**96.61 ±0.02**	**96.24 ±0.02**	**96.31 ±0.02**	**96.55 ±0.02**
Feat	99.77 ±0.00	**99.68 ±0.00**	99.73 ±0.00	**99.64 ±0.00**	99.73 ±0.00	**99.53 ±0.00**	99.73 ±0.00	**99.68 ±0.00**	**99.68 ±0.00**	**99.68 ±0.00**	**99.68 ±0.00**	**99.59 ±0.00**	**99.59 ±0.00**	**99.59 ±0.00**	**99.68 ±0.00**	**99.68 ±0.00**	**99.64 ±0.00**	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	**99.43 ±0.00**	**99.43 ±0.00**
Img	84.42 ±0.08	**81.33 ±0.25**	83.55 ±0.09	**77.14 ±0.55**	**81.13 ±0.09**	84.04 ±0.10	**81.07 ±0.09**	**82.94 ±0.10**	**82.68 ±0.11**	**82.94 ±0.10**	**82.94 ±0.09**	**83.12 ±0.10**	**80.61 ±0.11**	**82.94 ±0.10**	83.29 ±0.10	**83.03 ±0.10**	83.20 ±0.10	**71.95 ±0.22**	**83.12 ±0.09**	**82.25 ±0.11**	83.94 ±0.10	83.77 ±0.11
Spam	86.72 ±0.01	**86.23 ±0.00**	86.90 ±0.01	**85.23 ±0.00**	**86.19 ±0.01**	87.57 ±0.01	**86.10 ±0.01**	87.20 ±0.01	86.54 ±0.00	86.54 ±0.00	**84.19 ±0.00**	86.51 ±0.01	**84.09 ±0.01**	86.51 ±0.00	**84.23 ±0.00**	86.57 ±0.00	86.85 ±0.01	87.15 ±0.01	86.89 ±0.00	87.25 ±0.01	87.47 ±0.01	87.11 ±0.00
Thy	95.75 ±0.01	**94.65 ±0.03**	94.99 ±0.02	**94.14 ±0.05**	**95.07 ±0.01**	94.87 ±0.02	**94.99 ±0.01**	95.24 ±0.02	**94.65 ±0.03**	**94.65 ±0.02**	**94.82 ±0.03**	**94.39 ±0.02**	**94.31 ±0.02**	**94.56 ±0.03**	**94.99 ±0.02**	**94.73 ±0.02**	**94.90 ±0.02**	**94.65 ±0.02**	**94.99 ±0.02**	**94.39 ±0.03**	**94.77 ±0.02**	**95.03 ±0.02**
TTT	83.13 ±0.04	**77.43 ±0.04**	82.56 ±0.04	**71.17 ±0.02**	**80.13 ±0.02**	83.04 ±0.03	**79.90 ±0.02**	**81.51 ±0.03**	**78.57 ±0.04**	**78.44 ±0.05**	**78.31 ±0.05**	**76.33 ±0.03**	**75.08 ±0.04**	**74.97 ±0.03**	**78.84 ±0.05**	**78.86 ±0.05**	**80.72 ±0.06**	**80.15 ±0.02**	**80.57 ±0.05**	**80.57 ±0.04**	82.94 ±0.03	**80.87 ±0.04**
Wave	86.72 ±0.00	**86.48 ±0.00**	86.73 ±0.00	**86.01 ±0.00**	**81.02 ±0.00**	**86.53 ±0.00**	**80.96 ±0.00**	86.77 ±0.00	86.72 ±0.00	86.70 ±0.00	86.70 ±0.00	86.68 ±0.00	86.69 ±0.00	86.68 ±0.00	86.70 ±0.00	86.70 ±0.00	86.72 ±0.00	**85.58 ±0.00**	86.70 ±0.00	86.67 ±0.00	**86.43 ±0.00**	**86.41 ±0.00**
Wine	97.42 ±0.02	**96.66 ±0.03**	96.90 ±0.02	**96.18 ±0.03**	97.00 ±0.02	**96.70 ±0.02**	96.97 ±0.02	96.90 ±0.02	**96.69 ±0.02**	96.90 ±0.02	96.90 ±0.02	96.90 ±0.02	**96.80 ±0.02**	96.90 ±0.02	96.90 ±0.02	96.90 ±0.02	**96.80 ±0.02**	**95.66 ±0.02**	**96.28 ±0.03**	96.90 ±0.02	**96.60 ±0.02**	**96.81 ±0.02**

Table 3.7 LFM VS Other Fusion Methods
Average Classification Accuracy and Variance of Testing Set of MCSs with 20 base classifiers over Thirty Independent Runs

L=20	LFM	DS	DS	DVS	KU	W-KU	KE	W-KE	CM	MD	C1	C2	C3	C4	C5	C6	ME	LC1	LC2	LC3	Pri	Post
Canc	96.48 ±0.00	**96.03 ±0.00**	**96.19 ±0.00**	**95.77 ±0.00**	**95.71 ±0.00**	**96.31 ±0.00**	**95.79 ±0.00**	**96.29 ±0.00**	**96.13 ±0.00**	**96.13 ±0.00**	**96.13 ±0.00**	**96.00 ±0.00**	**96.13 ±0.00**	**96.09 ±0.00**	**96.13 ±0.00**	**96.13 ±0.00**	**96.06 ±0.00**	**95.90 ±0.00**	**95.84 ±0.00**	**96.16 ±0.00**	**96.21 ±0.00**	**96.05 ±0.00**
Car	91.40 ±0.00	**89.72 ±0.00**	91.24 ±0.00	**87.44 ±0.01**	93.17 ±0.00	92.32 ±0.00	93.06 ±0.00	91.29 ±0.00	**90.49 ±0.00**	**90.39 ±0.00**	**90.39 ±0.00**	**90.23 ±0.00**	**90.32 ±0.00**	**90.35 ±0.00**	**90.57 ±0.00**	**90.57 ±0.00**	**90.66 ±0.00**	**89.42 ±0.01**	**90.76 ±0.00**	**91.13 ±0.00**	92.22 ±0.00	92.14 ±0.00
Conn	81.55 ±0.09	**77.61 ±0.07**	**80.23 ±0.08**	**74.14 ±0.07**	85.00 ±0.08	82.06 ±0.06	85.29 ±0.08	**79.52 ±0.07**	**78.82 ±0.05**	**78.55 ±0.07**	**78.11 ±0.07**	**73.61 ±0.13**	**75.11 ±0.15**	**75.29 ±0.18**	**78.46 ±0.07**	**78.55 ±0.07**	**79.44 ±0.08**	**80.05 ±0.06**	**79.79 ±0.06**	80.67 ±0.04	81.96 ±0.06	81.52 ±0.06
Cred	85.65 ±0.01	85.25 ±0.01	85.36 ±0.01	85.28 ±0.02	**82.06 ±0.01**	**85.03 ±0.02**	**82.48 ±0.01**	85.23 ±0.02	**85.12 ±0.01**	85.17 ±0.01	85.20 ±0.01	**84.54 ±0.01**	**84.38 ±0.01**	**84.38 ±0.01**	85.20 ±0.01	85.20 ±0.01	**85.10 ±0.01**	**84.46 ±0.02**	**85.04 ±0.02**	**85.02 ±0.01**	**84.93 ±0.02**	**84.66 ±0.01**
Derm	97.05 ±0.00	96.90 ±0.00	97.00 ±0.00	**96.75 ±0.01**	**96.70 ±0.00**	**96.80 ±0.00**	**96.77 ±0.01**	96.95 ±0.00	96.85 ±0.00	96.95 ±0.00	96.95 ±0.00	**92.61 ±0.08**	96.85 ±0.00	**94.91 ±0.02**	97.05 ±0.00	97.05 ±0.00	96.85 ±0.01	**94.66 ±0.02**	**96.55 ±0.01**	**96.65 ±0.00**	**96.70 ±0.00**	**96.75 ±0.00**
Pima	77.08 ±0.01	76.59 ±0.01	76.82 ±0.01	**76.11 ±0.01**	**69.91 ±0.03**	**76.60 ±0.01**	**70.14 ±0.03**	76.82 ±0.01	76.68 ±0.01	76.80 ±0.01	76.73 ±0.01	76.73 ±0.01	76.73 ±0.01	76.75 ±0.01	76.73 ±0.01	76.80 ±0.01	76.68 ±0.01	**75.54 ±0.01**	**76.52 ±0.01**	**76.42 ±0.01**	**76.50 ±0.01**	76.83 ±0.01
Solar	66.58 ±0.01	66.50 ±0.01	66.27 ±0.01	66.76 ±0.01	**55.25 ±0.00**	**57.08 ±0.04**	**56.20 ±0.00**	**62.53 ±0.14**	66.51 ±0.01	66.49 ±0.01	**55.37 ±0.00**	**65.58 ±0.02**	**55.38 ±0.00**	**65.26 ±0.03**	**55.37 ±0.00**	66.49 ±0.01	66.51 ±0.01	66.73 ±0.01	66.47 ±0.01	66.25 ±0.01	**56.98 ±0.04**	**66.05 ±0.01**
Germ	75.00 ±0.03	**74.23 ±0.02**	74.64 ±0.02	**73.82 ±0.02**	**73.33 ±0.01**	74.49 ±0.02	**73.33 ±0.01**	74.45 ±0.02	**74.25 ±0.02**	**74.29 ±0.02**	74.45 ±0.02	**74.27 ±0.02**	74.36 ±0.02	74.36 ±0.02	74.40 ±0.02	74.42 ±0.02	74.38 ±0.02	**73.51 ±0.02**	74.82 ±0.02	**74.29 ±0.02**	74.39 ±0.02	74.34 ±0.03
Glass	87.10 ±0.07	**84.91 ±0.10**	86.75 ±0.12	**82.25 ±0.09**	86.32 ±0.13	86.81 ±0.11	86.65 ±0.13	87.01 ±0.11	**85.28 ±0.12**	**85.11 ±0.10**	**85.63 ±0.10**	**85.63 ±0.14**	**83.64 ±0.09**	**84.50 ±0.13**	86.06 ±0.11	85.97 ±0.11	86.41 ±0.08	**78.61 ±0.03**	86.49 ±0.11	86.23 ±0.13	86.71 ±0.11	87.15 ±0.11
Heart	82.49 ±0.02	81.98 ±0.03	81.89 ±0.02	82.29 ±0.04	**78.45 ±0.02**	**81.62 ±0.03**	**78.46 ±0.02**	**81.82 ±0.02**	**81.48 ±0.02**	**81.68 ±0.03**	**81.68 ±0.02**	**80.54 ±0.02**	**80.34 ±0.02**	**80.07 ±0.02**	**81.62 ±0.02**	**81.62 ±0.02**	**81.48 ±0.02**	**79.93 ±0.02**	**80.74 ±0.03**	**81.01 ±0.04**	**81.52 ±0.03**	**81.25 ±0.02**
Hepa	87.41 ±0.06	86.87 ±0.06	86.48 ±0.05	87.65 ±0.06	86.25 ±0.13	86.51 ±0.06	86.32 ±0.10	86.71 ±0.06	**86.25 ±0.05**	86.48 ±0.05	86.48 ±0.05	86.48 ±0.05	86.48 ±0.05	86.48 ±0.05	86.71 ±0.06	86.71 ±0.06	**86.01 ±0.05**	**86.01 ±0.10**	**86.01 ±0.07**	**86.25 ±0.05**	86.41 ±0.06	**86.18 ±0.06**
Iono	88.16 ±0.01	88.09 ±0.01	88.42 ±0.01	**87.48 ±0.01**	**86.81 ±0.02**	88.16 ±0.01	**86.81 ±0.02**	88.42 ±0.01	87.84 ±0.01	88.31 ±0.01	88.16 ±0.01	88.26 ±0.01	88.00 ±0.01	88.31 ±0.01	88.36 ±0.01	88.73 ±0.01	**86.75 ±0.02**	**85.45 ±0.01**	**87.01 ±0.01**	**86.49 ±0.01**	88.06 ±0.01	88.58 ±0.01
Iris	96.97 ±0.02	96.61 ±0.02	96.61 ±0.02	96.61 ±0.01	**96.12 ±0.02**	96.41 ±0.02	**96.17 ±0.02**	96.61 ±0.02	96.61 ±0.02	96.61 ±0.02	**96.00 ±0.02**	96.61 ±0.02	**95.76 ±0.02**	96.61 ±0.02	**96.00 ±0.02**	96.61 ±0.02	96.61 ±0.02	96.73 ±0.01	96.73 ±0.02	96.48 ±0.02	**96.31 ±0.02**	96.43 ±0.02
Feat	99.73 ±0.00	99.70 ±0.00	99.73 ±0.00	**99.64 ±0.00**	99.73 ±0.00	**99.53 ±0.00**	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	**99.59 ±0.00**	**99.59 ±0.00**	**99.59 ±0.00**	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	99.73 ±0.00	**99.43 ±0.00**	**99.43 ±0.00**
Img	84.59 ±0.07	**81.76 ±0.19**	83.98 ±0.11	**77.58 ±0.36**	**81.99 ±0.11**	84.30 ±0.13	**82.25 ±0.11**	83.55 ±0.10	**83.29 ±0.10**	**83.20 ±0.10**	83.38 ±0.10	83.38 ±0.10	**80.78 ±0.10**	83.72 ±0.11	83.64 ±0.10	83.64 ±0.10	**83.03 ±0.10**	**71.43 ±0.23**	84.07 ±0.10	**82.94 ±0.08**	84.20 ±0.13	84.03 ±0.12
Spam	86.64 ±0.01	**86.10 ±0.01**	86.77 ±0.01	**85.08 ±0.01**	86.55 ±0.01	87.65 ±0.01	86.47 ±0.01	87.04 ±0.01	86.48 ±0.01	86.43 ±0.01	**84.08 ±0.01**	86.43 ±0.01	**84.00 ±0.01**	86.43 ±0.01	**84.12 ±0.01**	86.45 ±0.01	86.70 ±0.01	87.13 ±0.01	86.79 ±0.01	87.17 ±0.01	87.55 ±0.01	87.14 ±0.01
Thy	95.33 ±0.01	**94.68 ±0.03**	94.82 ±0.02	**94.39 ±0.03**	95.41 ±0.01	**94.70 ±0.02**	95.35 ±0.01	95.07 ±0.02	**94.65 ±0.02**	**94.73 ±0.02**	94.82 ±0.02	**94.39 ±0.02**	**94.31 ±0.02**	**94.48 ±0.02**	94.82 ±0.02	94.90 ±0.02	94.82 ±0.02	94.99 ±0.02	**94.73 ±0.03**	95.07 ±0.02	**94.60 ±0.02**	**94.77 ±0.02**
TTT	84.30 ±0.04	**78.55 ±0.05**	83.91 ±0.06	**72.31 ±0.05**	**82.39 ±0.04**	84.24 ±0.05	**82.61 ±0.03**	**82.79 ±0.04**	**79.22 ±0.03**	**78.78 ±0.03**	**78.42 ±0.03**	**76.20 ±0.04**	**74.42 ±0.03**	**74.40 ±0.03**	**79.45 ±0.03**	**79.48 ±0.03**	**81.89 ±0.04**	**82.88 ±0.03**	**82.05 ±0.04**	**82.08 ±0.05**	84.14 ±0.05	**82.68 ±0.03**
Wave	86.89 ±0.00	**86.57 ±0.00**	**86.78 ±0.00**	**86.19 ±0.00**	**81.91 ±0.00**	**86.56 ±0.00**	**81.84 ±0.00**	**86.78 ±0.00**	**86.79 ±0.00**	**86.75 ±0.00**	**86.74 ±0.00**	86.82 ±0.00	**86.79 ±0.00**	86.80 ±0.00	**86.71 ±0.00**	**86.71 ±0.00**	**86.74 ±0.00**	**85.53 ±0.00**	**86.77 ±0.00**	**86.77 ±0.00**	**86.46 ±0.00**	**86.51 ±0.00**
Wine	97.52 ±0.02	**96.94 ±0.02**	97.11 ±0.02	**96.69 ±0.02**	97.31 ±0.02	**96.91 ±0.02**	97.32 ±0.02	97.11 ±0.02	97.00 ±0.02	97.00 ±0.02	97.00 ±0.02	**96.90 ±0.02**	**96.80 ±0.02**	**96.80 ±0.02**	97.11 ±0.02	97.11 ±0.02	**96.49 ±0.03**	**94.94 ±0.04**	**96.80 ±0.02**	97.11 ±0.02	**96.81 ±0.02**	**96.81 ±0.02**

Table 3.8 LFM VS Other Fusion Methods
Average Classification Accuracy and Variance of Testing Set of MCSs with 30 base classifiers over Thirty Independent Runs

L=30	LFM	DS	DV	DVS	KU	W-KU	KE	W-KE	CM	MD	C1	C2	C3	C4	C5	C6	ME	LC1	LC2	LC3	Pri	Post
Canc	96.38 ±0.00	**95.97 ±0.00**	**96.09 ±0.00**	**95.77 ±0.00**	**95.65 ±0.00**	**96.09 ±0.00**	**95.69 ±0.00**	**96.16 ±0.00**	**96.03 ±0.00**	**96.03 ±0.00**	**96.03 ±0.00**	**95.90 ±0.00**	**96.00 ±0.00**	**95.97 ±0.00**	**96.03 ±0.00**	**96.03 ±0.00**	**96.09 ±0.00**	**95.81 ±0.00**	**95.97 ±0.00**	**96.06 ±0.00**	**95.99 ±0.00**	**95.86 ±0.00**
Car	91.45 ±0.00	**89.71 ±0.00**	91.36 ±0.00	**87.31 ±0.01**	93.66 ±0.00	92.39 ±0.00	93.63 ±0.00	91.38 ±0.00	**90.47 ±0.00**	**90.33 ±0.00**	**90.37 ±0.00**	**90.26 ±0.00**	**90.33 ±0.00**	**90.35 ±0.00**	**90.55 ±0.00**	**90.56 ±0.00**	**90.67 ±0.00**	**89.49 ±0.01**	**90.69 ±0.00**	**91.17 ±0.00**	92.29 ±0.00	92.13 ±0.00
Conn	81.20 ±0.09	**78.05 ±0.08**	80.49 ±0.09	**74.93 ±0.10**	85.61 ±0.05	82.77 ±0.05	85.31 ±0.05	**79.79 ±0.07**	**78.55 ±0.05**	**78.02 ±0.06**	**77.93 ±0.07**	**74.67 ±0.16**	**76.43 ±0.12**	**75.38 ±0.21**	**78.11 ±0.06**	**78.02 ±0.06**	**79.52 ±0.08**	80.32 ±0.10	80.41 ±0.07	81.91 ±0.02	82.67 ±0.05	81.96 ±0.07
Cred	85.65 ±0.01	85.24 ±0.02	85.20 ±0.02	85.44 ±0.02	**82.21 ±0.01**	**84.97 ±0.02**	**81.80 ±0.01**	85.20 ±0.01	**85.10 ±0.01**	**85.10 ±0.01**	85.17 ±0.01	**84.41 ±0.02**	**84.73 ±0.02**	**84.62 ±0.01**	85.15 ±0.01	85.15 ±0.01	85.20 ±0.02	**84.01 ±0.02**	85.36 ±0.02	**85.10 ±0.02**	**84.87 ±0.02**	**84.69 ±0.01**
Derm	97.10 ±0.00	**96.87 ±0.00**	**96.85 ±0.00**	**96.85 ±0.00**	**96.80 ±0.01**	**96.70 ±0.00**	**96.82 ±0.00**	**96.85 ±0.00**	**96.80 ±0.00**	**96.80 ±0.00**	**96.80 ±0.00**	**92.91 ±0.06**	**96.75 ±0.00**	**95.40 ±0.01**	96.90 ±0.00	96.90 ±0.00	**96.55 ±0.01**	**94.41 ±0.02**	**96.25 ±0.01**	**96.85 ±0.00**	**96.60 ±0.00**	**96.65 ±0.00**
Pima	77.32 ±0.02	**76.51 ±0.02**	76.75 ±0.02	**76.09 ±0.02**	**70.19 ±0.02**	**76.46 ±0.02**	**69.99 ±0.02**	76.80 ±0.01	**76.56 ±0.02**	**76.59 ±0.02**	**76.59 ±0.02**	76.75 ±0.02	76.78 ±0.02	**76.70 ±0.02**	**76.61 ±0.02**	**76.61 ±0.02**	**76.61 ±0.02**	**75.57 ±0.01**	**76.59 ±0.02**	**76.47 ±0.02**	**76.36 ±0.02**	**76.57 ±0.01**
Solar	66.63 ±0.01	66.36 ±0.01	66.29 ±0.01	66.58 ±0.01	**55.25 ±0.00**	**57.08 ±0.04**	**56.25 ±0.00**	**62.78 ±0.15**	66.49 ±0.01	**66.22 ±0.01**	**55.37 ±0.00**	**66.22 ±0.01**	**55.49 ±0.00**	**66.01 ±0.01**	**55.33 ±0.00**	66.47 ±0.01	66.27 ±0.01	66.40 ±0.00	66.29 ±0.01	**66.18 ±0.01**	**56.98 ±0.04**	66.28 ±0.01
Germ	75.15 ±0.03	74.47 ±0.02	74.78 ±0.03	**74.24 ±0.01**	**73.56 ±0.01**	74.69 ±0.03	**73.49 ±0.01**	74.69 ±0.03	**74.35 ±0.03**	**74.42 ±0.02**	74.56 ±0.03	**74.27 ±0.03**	**74.40 ±0.03**	**74.35 ±0.03**	74.58 ±0.03	74.60 ±0.03	74.76 ±0.03	**73.22 ±0.01**	74.98 ±0.02	**74.33 ±0.03**	74.59 ±0.03	74.59 ±0.03
Glass	86.84 ±0.09	**84.79 ±0.10**	86.06 ±0.10	**82.60 ±0.08**	86.58 ±0.16	86.38 ±0.15	86.77 ±0.17	86.49 ±0.13	85.63 ±0.09	**85.19 ±0.10**	**85.37 ±0.11**	85.89 ±0.10	**83.90 ±0.09**	**83.81 ±0.11**	86.49 ±0.11	86.49 ±0.11	86.15 ±0.07	**78.35 ±0.09**	86.06 ±0.10	85.80 ±0.15	86.28 ±0.15	86.71 ±0.13
Heart	82.90 ±0.03	**82.09 ±0.03**	**81.95 ±0.03**	82.56 ±0.03	**78.45 ±0.02**	**81.55 ±0.03**	**78.40 ±0.02**	**81.75 ±0.03**	**81.89 ±0.03**	**81.82 ±0.03**	**81.62 ±0.03**	**80.27 ±0.03**	**80.34 ±0.02**	**80.74 ±0.00**	**81.55 ±0.03**	**81.55 ±0.03**	**81.68 ±0.03**	**79.80 ±0.03**	**81.41 ±0.03**	**81.62 ±0.03**	**81.45 ±0.03**	**81.32 ±0.02**
Hepa	87.65 ±0.03	86.79 ±0.07	**86.48 ±0.09**	87.41 ±0.04	**86.01 ±0.13**	**86.28 ±0.09**	**85.87 ±0.13**	**86.48 ±0.09**	**86.25 ±0.07**	**86.71 ±0.04**	**86.01 ±0.08**	**86.25 ±0.04**	**86.48 ±0.05**	**86.25 ±0.04**	**86.48 ±0.09**	**86.48 ±0.09**	**85.31 ±0.07**	**86.01 ±0.06**	**86.25 ±0.05**	**85.55 ±0.07**	**86.18 ±0.09**	**86.18 ±0.09**
Iono	88.21 ±0.00	88.07 ±0.01	88.57 ±0.01	**87.43 ±0.01**	**86.96 ±0.01**	88.32 ±0.01	**86.70 ±0.01**	88.47 ±0.01	88.10 ±0.01	88.21 ±0.01	88.10 ±0.01	88.21 ±0.01	87.90 ±0.01	88.21 ±0.01	88.16 ±0.01	88.47 ±0.01	**87.69 ±0.01**	**86.03 ±0.01**	**87.27 ±0.01**	**86.96 ±0.02**	88.22 ±0.01	88.48 ±0.01
Iris	97.09 ±0.02	96.61 ±0.02	**96.48 ±0.02**	96.85 ±0.02	**96.36 ±0.02**	**96.41 ±0.02**	**96.37 ±0.02**	96.61 ±0.02	96.73 ±0.02	**96.48 ±0.02**	**95.88 ±0.03**	**96.48 ±0.02**	**95.88 ±0.02**	**96.48 ±0.02**	**95.88 ±0.03**	**96.48 ±0.02**	96.61 ±0.02	96.61 ±0.02	96.61 ±0.02	**96.24 ±0.01**	**96.31 ±0.02**	**96.31 ±0.02**
Feat	99.73 ±0.00	99.65 ±0.00	99.68 ±0.00	**99.64 ±0.00**	99.73 ±0.00	**99.48 ±0.00**	99.73 ±0.00	**99.64 ±0.00**	**99.64 ±0.00**	**99.64 ±0.00**	**99.64 ±0.00**	**99.59 ±0.00**	**99.59 ±0.00**	**99.59 ±0.00**	**99.64 ±0.00**	**99.64 ±0.00**	**99.64 ±0.00**	99.68 ±0.00	99.68 ±0.00	99.73 ±0.00	**99.38 ±0.00**	**99.38 ±0.00**
Img	85.11 ±0.09	**82.77 ±0.10**	84.24 ±0.11	**80.17 ±0.08**	**82.86 ±0.11**	84.65 ±0.11	**82.61 ±0.11**	83.90 ±0.10	**83.81 ±0.10**	**83.81 ±0.10**	**83.72 ±0.10**	83.90 ±0.10	**80.87 ±0.08**	83.81 ±0.11	**83.72 ±0.10**	**83.72 ±0.10**	83.98 ±0.08	**73.59 ±0.13**	84.33 ±0.12	**83.03 ±0.08**	84.55 ±0.11	84.29 ±0.12
Spam	86.66 ±0.01	**86.10 ±0.01**	86.83 ±0.01	**85.00 ±0.00**	86.94 ±0.01	87.86 ±0.01	86.71 ±0.01	87.03 ±0.01	86.47 ±0.01	86.46 ±0.01	**84.08 ±0.01**	86.42 ±0.01	**84.00 ±0.01**	86.42 ±0.01	**84.14 ±0.01**	86.49 ±0.01	86.77 ±0.01	87.29 ±0.01	86.87 ±0.01	87.26 ±0.01	87.76 ±0.01	87.21 ±0.01
Thy	95.33 ±0.01	**94.70 ±0.02**	94.90 ±0.02	**94.39 ±0.02**	95.50 ±0.01	**94.79 ±0.02**	95.39 ±0.01	95.24 ±0.02	**94.73 ±0.02**	**94.82 ±0.02**	95.07 ±0.02	**94.73 ±0.02**	**94.56 ±0.02**	**94.73 ±0.02**	95.07 ±0.02	95.07 ±0.02	95.07 ±0.02	**94.56 ±0.02**	94.90 ±0.03	94.73 ±0.04	**94.69 ±0.02**	94.86 ±0.02
TTT	84.68 ±0.04	**79.22 ±0.03**	84.67 ±0.03	**72.96 ±0.04**	**83.34 ±0.03**	84.69 ±0.04	**83.44 ±0.03**	**83.34 ±0.04**	**80.00 ±0.02**	**79.75 ±0.02**	**78.61 ±0.02**	**76.66 ±0.02**	**74.63 ±0.02**	**74.59 ±0.02**	**79.83 ±0.02**	**79.83 ±0.02**	**82.41 ±0.03**	**83.51 ±0.03**	**82.67 ±0.03**	**82.44 ±0.03**	84.59 ±0.04	**83.55 ±0.02**
Wave	86.87 ±0.00	**86.63 ±0.00**	86.84 ±0.00	**86.22 ±0.00**	**82.23 ±0.00**	**86.66 ±0.00**	**81.80 ±0.00**	86.84 ±0.00	86.83 ±0.00	86.82 ±0.00	86.81 ±0.00	86.83 ±0.00	86.82 ±0.00	86.84 ±0.00	86.83 ±0.00	86.83 ±0.00	86.79 ±0.00	**85.78 ±0.00**	86.79 ±0.00	86.82 ±0.00	**86.56 ±0.00**	**86.56 ±0.00**
Wine	97.52 ±0.02	97.07 ±0.02	97.31 ±0.02	**96.69 ±0.02**	97.52 ±0.01	97.11 ±0.02	97.52 ±0.01	97.31 ±0.02	97.21 ±0.02	97.31 ±0.02	97.31 ±0.02	97.21 ±0.02	97.11 ±0.03	97.00 ±0.02	97.31 ±0.02	97.21 ±0.02	**96.69 ±0.02**	**95.87 ±0.02**	**96.49 ±0.03**	97.00 ±0.02	97.01 ±0.02	97.01 ±0.02

Figures (3.7) and (3.8) show the average training and testing time of MCSs with 30 base classifiers for different dynamic fusion methods on twenty datasets over thirty independent runs. Each bar represents the time of MCS using a fusion method.

The training of C1 – C6 methods is the fastest since no additional information is required after the base classifiers are trained. As MD only needs to calculate the mean and variance of the training set for each base classifier, it is slightly faster than the fusion method, such as, KU, W-KU, KE, W-KE, Pri and Post, which need to calculate the errors for the training sample. The calculation of an additional term (sensitivity term) causes LFM

to take longer training time which is roughly double the time of other methods using the training error only. Generally, DS, DV and DVS should take a longer training time than ME and LC1 – LC3. This is because when 5-cross validation is used in DS, DV and DVS, 5 times more classifiers are trained. However, in this experiment, RBF Network and MLP Neural Network are used as base classifiers and the Gate Network for ME and LC1 – LC3. The gradient descent training method of MLP Neural Network takes much longer time than the training of RBF network. That is the reason why DS, DV and DVS have a shorter training time than ME and LC1 – LC3. Moreover, due to the training of additional neural networks, the training time of DV, DVS, ME and LC1- LC3 are significantly longer than other methods.

Figure 3.7 LFM VS Other Fusion Methods
Average Training Time of MCSs with 30 base classifiers on Twenty Datasets over Thirty Independent Runs

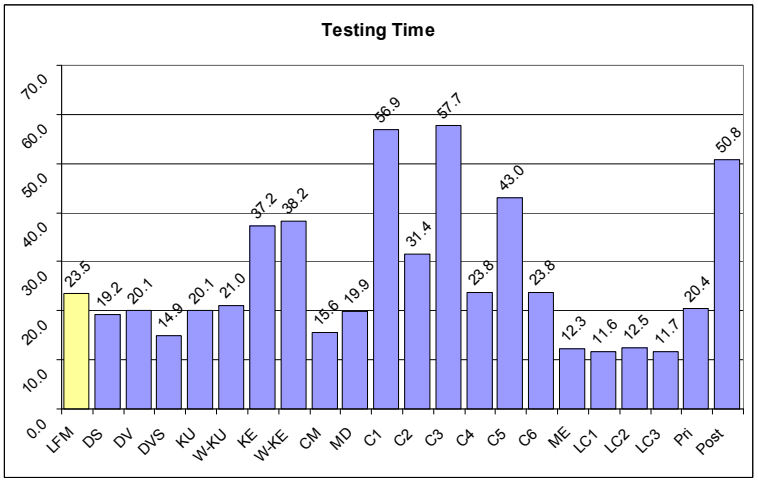


Figure 3.8 LFM VS Other Fusion Methods
Average Testing Time of MCSs with 30 base classifiers on Twenty Datasets over Thirty Independent Runs

The differences of testing time among fusion methods are smaller than those for the training time. As C1 – C6 and Post need a complex calculation on weights, they require a longer testing time. While ME and LC1 – LC3 have the shortest testing time since the values of their weights are only based on the gating network output. The testing time of the fusion methods (such as DS, DV, DVS, KU, W-KU, KE, W-KE and Pri) which measure the training errors on the K-nearest samples from the testing sample is 23.9 on average. This average testing time is similar to the time required by the LFM. This shows that LFM does not require much additional time on classifying samples although both training error and sensitivity term are considered.

In conclusion, as the LFM uses the localized generalization error bound, which consists of training error and sensitivity term, for weight assignments, additional training time (roughly 2 times) is needed comparing with other fusion methods which only calculate the training error. However, the training time of the LFM is still much shorter than some fusion methods, such as DS, DV, DVS, ME and LC1 – LC3. When classifying samples, the time required by LFM is similar to the one considering only the training error. As a result,

although additional information is required for the LFM, its training and testing time are comparable to those of the current dynamic fusion methods.

3.4 Conclusion

In this chapter, a novel dynamic fusion method, the L-GEM Fusion Method (LFM), is proposed. The currently available dynamic fusion methods estimate the performances of the base classifiers only based on the training samples. The limitation of these approaches is that the information of unseen samples near the testing sample have not been considered. The LFM measures the generalization error bound on a neighborhood containing the testing sample by the L-GEM. The L-GEM uses both the training error and the sensitivity of the classifier outputs to assign weight to each base classifier. Conceptually the LFM penalizes a fluctuating base classifier by assigning to it a smaller weight.

Although additional computations are required for LFM, its testing time complexity is the same as the dynamic fusion methods which consider the training error only. This is because the information on training error and sensitivity can be computed and stored during the training phase. The information can be retrieved when classifying testing samples. The experimental result also shows that the LFM is competitive with other dynamic fusion methods in terms of time complexity.

The effect of the number of nearest neighborhoods (K) in LFM is studied. The performance of a MCS using the LFM decreases when K increases. Based on the experimental studies, MCSs perform the best when K is between 1 and 5. LFM has also been compared with other twenty one dynamic fusion methods. The experimental results show that for a set of trained base classifiers, the testing accuracies of a MCS using LFM outperforms those using other fusion dynamic methods. It shows that weight assignment

based on localized generalization error bound in LFM can better reflect the actual local competence of base classifiers than other dynamic fusion methods. The sensitivity term, in addition to the training error term, proves to be useful in estimation the local competence for a MCS.

CHAPTER 4
L-GEM FOR MCSS (L-GEM^{MCS})

The generalization error is the single most important criterion in the evaluation of a classifier. However, the exact computation of its value is generally not feasible since the probability density function of the input or the true input-output mapping function or both are not known. This problem is true for both single and multiple classifiers. Some generalization error models proposed for MCSs [Krogh et al 1995, Tumer et al 1996a, Ueda et al 1996] are difficult to use since part of their models are not computable. In other cases their error bounds may be too loose [Freund et al 1996]. The major contribution of these models is their attempt to describe the relationship between the errors of the MCS and its base classifiers. This observation motivates us to extend the L-GEM from single classifiers to MCSs.

In this chapter, the factors which affect the performance of MCSs and the existing error models for MCSs are presented in Section 4.1. The extension of L-GEM, named L-GEM^{MCS}, form single classifiers to MCSs is derived in Section 4.2, which is the main result in this section. The components of the L-GEM^{MCS} are discussed in Section 4.3. Section 4.4 presents the analysis of the characteristics of the L-GEM^{MCS}. An effective way of computing the sensitivity terms in the L-GEM^{MCS} is proposed in Section 4.5. A comparison technique for MCSs using the L-GEM^{MCS} is presented in Section 4.6. Section 4.7 concludes this chapter.

4.1 Existing Error Models for MCSs

In this section, the existing error models error models for MCSs are introduced. The three factors which affact the error of MCSs are discussed and the

4.1.1 Three Factors Affect the Performance of MCSs

A MCS combines the outputs of a set of base classifiers to reach the final decision. The performance of a MCS is affected by three factors: Base Classifier Performance, Fusion Method and Diversity.

Base Classifier Performance

Since a MCS is made up by a number of base classifiers, its decision error will obviously be affected by its base classifiers. Discussions on error models for single classifiers are available in the literature and they will not be presented in this thesis.

Fusion Method

A fusion method defines the way to combine the outputs of the base classifiers to form the final output for the MCS. It is interesting to note that, although the outputs of the base classifiers are the same, the output of the MCS will be different if different fusion methods are used. Here is a simple example:

Table 4.1 MCS with Different Fusion Methods

	Class1	Class2
$f_1(x)$	0.2	0.8
$f_2(x)$	0.7	0.3
$f_3(x)$	0.7	0.3
$f^{mcs}(x)$ with Avg	0.53	0.47
$f^{mcs}(x)$ with Max	0.7	0.8

Table (4.1) shows that the base classifier 1 has 0.2 confidence of x belonging to class 1 and 0.8 confidence of x belonging to class 2. While both base classifiers 2 and 3 have 0.7 for class 1 and 0.3 for class 2, If we use the averaging fusion method, class 1 will have a

score of 0.53 and class 2 will get 0.47. In this case, the MCS outputs class 1. But when the maximum fusion method is used, the output of MCS is class 2 since the value for class 2 would be 0.8 which is greater than the value of 0.7 for class 1. This simple example shows that the performance of a MCS is seriously affected by the choice of the fusion method.

The simplest fusion method is the weighted average sum

$$f^{mcs}(x) = \sum_{i=1}^{L} w_i f_i(x), \tag{4.1}$$

where w_i is the non-negative weight for the i^{th} base classifier and $\sum_{l=1}^{L} w_l = 1$. When all weights are equal to $1/L$, it becomes a simple average fusion method. The weighted average and the simple average are the most intensively studied fusion technique [Brown 2004, Fumera et al 2008, Islam et al 2008, Shirai et al 2008, Skurichina et al 2000, Wolpert et al 1999] because of their linear property.

For non-linear fusion methods, the majority vote [Kuncheva et al 2003b, Lam et al 1996] and the weighted majority vote [Bhadoria et al 2005, Sun et al 2005] are the most well known ones. The idea of majority vote is very simple. The decision of a MCS is the class selected by most of the base classifiers. A more accurate base classifier is given a larger share of votes in the weighted majority vote.

Order statistics [Tumer et al 1995] has been used as fusion methods. The fusion of maximum, minimum and median are:

$$f^{mcs}(x) = \max_i \{f_i(x)\}, \tag{4.2}$$

$$f^{mcs}(x) = \min_i \{f_i(x)\}, \tag{4.3}$$

$$f^{mcs}(x) = \underset{i}{median}\{f_i(x)\} \quad . \tag{4.4}$$

There is no evidence that any one of these fusion methods always outperforms the others. In general, the performance of the weighted average shows the most satisfying result [Kuncheva 2004].

Diversity

Diversity is used to measure the degree of difference between the outputs of the base classifiers of a MCS. A MCS is said to be very diverse if the outputs of its base classifiers are totally different. On the other hand, if all base classifiers yield identical outputs, the MCS has zero diversity. If two base classifiers always classify correctly (wrongly), only one of them should be in the MCS since they cannot complement each others' errors. It is an intuitively appealing idea to have a most diversified MCS.

Table 4.2 Comparison between Diverse and Non-Diverse MCSs.
1 (0) denotes correct (wrong) recognition of the classifier

	f_1	f_2	f_3	f^{mcs}
x_1	1	1	1	1
x_2	0	0	0	0
x_3	0	0	0	0
x_4	1	1	1	1
x_5	1	1	1	1
Acc.	60%	60%	60%	**60%**

a) Non-Diverse MCS

	f_1	f_2	f_3	f^{mcs}
x_1	1	1	0	1
x_2	1	0	1	1
x_3	0	1	1	1
x_4	1	1	0	1
x_5	0	0	1	0
Acc.	60%	60%	60%	**80%**

b) Diverse MCS

	f_1	f_2	f_3	f^{mcs}
X_1	1	1	1	1
X_2	1	0	0	0
X_3	0	0	1	0
X_4	1	1	1	1
X_5	0	1	0	0
Acc.	60%	60%	60%	**40%**

c) Diverse MCS

Table (4.2) gives an example to show the effect of diversity upon the performance of a MCS. In this example, the accuracy of each base classifier is fixed at 60% and the

majority vote fusion method is used. The effect of different diversity measures will be investigated.

The MCS, shown in Table (4.2a), has identical base classifiers. This MCS has zero diversity. Obviously, the decision of the MCS is the same as any of its base classifiers. Therefore, its accuracy is also 60%. On the other hand, the outputs of the base classifiers are different in Table (4.2b). In this case, the accuracy of the MCS has increased to 80%. However, there is no guarantee on improvement of performance for a more diverse MCS. Table (4.2c) gives such an example. In fact the accuracy of the MCS is down to 40% although it is made up by more diverse base classifiers. It is clear that a more diverse MCS is desirable, but the performance of a MCS can only be improved by having good "diversity". The concept of diversity is rather abstract and its definition is by no means unique. Many diversity measures have been proposed and the following presents a brief discussion on a few representative ones [Banfield et al 2003, 2005, Giacinto et al 2001a, Ho 1998, Kuncheva et al 2003a, Liu et al 1998, Lu et al 2006, Rasheed et al 2008, Yule 1900].

The definition of diversity can be separated into two categories according to the types of base classifier's outputs: pairwise and non-pairwise measure. A Pairwise Diversity Measure considers a pair of classifiers at a time. Consider the outputs of two classifiers f_i and f_j, as shown in Table (4.3).

Table 4.3 Output Combinations of Two Classifiers

	f_i correct	f_i wrong
f_j correct	N^{11}	N^{10}
f_j wrong	N^{01}	N^{00}

For a pairwise diversity measure, since L base classifiers produce $L(L\text{-}1)/2$ pairwise diversity values, the diversity measure is equal to the mean of all pairs.

Yule [Yule 1900] proposed a statistical method to evaluate the similarity of two variables using the Q statistics (Q). The Q statistic is defined as:

$$Q_{ij} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}. \tag{4.5}$$

For statistically independent classifiers, $Q = 0$. The range of Q is between -1 and 1. When $Q > 0$, the two classifiers tend to give the same answer to the same objects and when $Q < 0$, the two classifiers tend to give the different answer to the same objects.

The Disagreement Measure (*DM*) may be the most intuitive diversity measure. It measures the probability of two classifiers disagreeing with each other. The value of *DM* is from 0 to 1. When DM is 1, it means the two classifiers always disagree. The definition of *DM* is:

$$DM_{ij} = \frac{N^{10} + N^{01}}{N^{11} + N^{10} + N^{01} + N^{00}}. \tag{4.6}$$

In the Entropy measure (*En*), the MCS is most diverse for a particular when $L/2$ of its base classifiers are correct and the rest are wrong. *En* is defined as:

$$En = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{(L - \lceil L/2 \rceil)}\min\{l(x_j), L - l(x_j)\}, \tag{4.7}$$

where $l(x)$ indicates the number of base classifiers in the MCS recognizing the sample x correctly. *En* varies between 0 and 1. $En = 0$ means no diversity and $En = 1$ means the highest diversity.

Percentage Correct Diversity Measure (*PCDM*) is an example of non-pairwise diversity measure [Banfield 2003]. It finds the number of samples which are classified correctly by K% of the base classifiers where K is between 10 and 90. The bounds of 10% and 90% are chosen empirically. *PCDM* is defined as:

$$PCDM = \frac{1}{N}\sum_{i=1}^{N}\delta_i \,, \tag{4.8}$$

where $\delta_i = \begin{cases} 1 & 0\% < \dfrac{L_i}{L} < 90\%, \\ 0 & otherwise \end{cases}$ and L_i denotes the number of base classifier which classifies sample x_i correctly. *PCDM* ranges from 1 (most diverse) to 0 (least diverse).

There is only one commonly used method to measure the diversity of MCS with continuous-valued outputs [Tumer et al 1996a]. This method measures the correlation coefficients of the errors of the outputs of individual classifiers. Similar to the label output, if the correlation coefficient is equal to 1, it means all classifiers are identical. When the correlation coefficient is zero, all classifiers are independent .When the correlation coefficient is -1, it is the best case since all classifiers are most diverse.

4.1.2 Hansen and Salamon Conditions

Hansen and Salamon [Hansen et al 1990] derive a set of necessary and sufficient conditions for a MCS to be more accurate than any of its individual members. The first condition is that each base classifier must be accurate. An error rate of each base classifier should be better than random guessing, for example, more than 50% for a two-class problem. Another condition is related to the diversity among the base classifiers. The errors of the base classifiers should be independent. Under these restrictive constraints, the error of a MCS can be simply expressed as a combination of the probabilities of base classifiers' errors when the majority vote fusion method is used:

$$\sum_{i=\lceil L/2 \rceil}^{L} \binom{L}{i} p^i (1-p)^{L-i} \tag{4.9}$$

where p is the probability of error of a base classifier. For example, assuming a MCS with 9 base classifiers using the majority vote as the fusion method and the error of the base

classifiers is 40%, the error of this MCS is 32%. It has been shown that Equation (4.9) is monotonically decreasing in L. When the performances of the base classifiers are better than random guess and their errors are independent, if more base classifiers are used, the less error the MCS will cause. In practice this model is not realistic since the statistically independent assumption on the errors of the base classifiers is nearly impossible to achieve.

4.1.3 Bias, Variance and Covariance Decomposition

The bias-variance decomposition [Geman et al 1992] states that the MSE can be broken down into two components: bias and variance. Let $f(x; D)$ denote the output of classifier f trained by a set D. Since D is selected randomly, the performance of f is certainly depending on D. The bias-variance decomposition is defined as follows:

$$\underset{Z}{E}\left(\left(f(x;D)-F(x)\right)^2\right)=\left(bias\left(f;D\right)\right)^2+\mathrm{var}\left(f;D\right), \tag{4.10}$$

where $bias\left(f;D\right)=\underset{Z}{E}\left(f(x;D)-F(x)\right)$, $\mathrm{var}\left(f;D\right)=\underset{Z}{E}\left(\left(f(x;D)-\underset{Z}{E}\left(f(x;D)\right)\right)^2\right)$ and Z is a set of D of fixed size N. The bias term measures how close the classifier is to its target while the variance term measures how stable the classifier is. It means if the given training set D changes slightly, a classifier with high variance will tend to exhibit wildly different performances when the dataset is changed.

For a MCS with an average fusion method, the variance term can be further broken down. It is known as the Bias-Variance-Covariance decomposition [Ueda et al 1996]:

$$\underset{Z}{E}\left(\left(f^{mcs}(x;D)-F(x)\right)^2\right)=(\mathrm{bias}^{\mathrm{MCS}})^2+\frac{1}{L}\mathrm{var}^{\mathrm{MCS}}+\left(1-\frac{1}{L}\right)\mathrm{covar}^{\mathrm{MCS}}, \tag{4.11}$$

where $\mathrm{bias}^{\mathrm{MCS}}=\frac{1}{L}\sum_{i=1}^{L}bias(f_i;D)$, $\mathrm{var}^{\mathrm{MCS}}=\frac{1}{L}\sum_{i=1}^{L}\mathrm{var}(f_i;D)$ and $\mathrm{covar}^{\mathrm{MCS}}=\frac{1}{L(L-1)}\sum_{i=1}^{L}\sum_{j\neq i}^{L}\underset{Z}{E}\left(\left(f_i(x;D)-\underset{Z}{E}\left(f_i(x;D)\right)\right)\left(f_j(x;D)-\underset{Z}{E}\left(f_j(x;D)\right)\right)\right)$. The $\mathrm{bias}^{\mathrm{MCS}}$ and $\mathrm{var}^{\mathrm{MCS}}$

terms are the average bias and variance of each base classifier. The covariance term is a new one. It measures the average covariance of each pair of base classifiers and it could be a negative value. If the base classifiers are negatively correlated, the covariance term contributes to a decrease in the MSE of the MCS. Conversely, if the base classifiers are positively correlated, the MSE of the MCS increases. The ideal situation is that the covariance is reduced without causing any increase in the bias and the variance terms. This error model describes the average error of a classifier on different datasets. It provides some general guideline for the design of a MCS in general, which is high bias, low variance and low covariance.

4.1.4 Ambiguity Decomposition

Korgh and Vedelsby [Krogh et al 1995] have shown that the MSE of a MCS is guaranteed to be less than or equal to the average MSE of its base classifiers:

$$\left(f^{mcs} - F\right)^2 = \sum_{l=1}^{L} w_l \left(f_l - F\right)^2 - \sum_{l=1}^{L} w_l \left(f_l - f^{mcs}\right)^2 , \tag{4.12}$$

where $f^{mcs}(x) = \sum_{l=1}^{L} w_l f_l(x)$.

The first term on the right hand side is the weighted average of the MSE of the base classifiers and the second term is called ambiguity. Since the ambiguity term is a non-negative number, the MSE of the MCS must not be larger than the weighted average of the MSE of its base classifiers from Equation (4.12).

The ambiguity term measures the variability of the base classifiers. It is a kind of diversity measure. Equation (4.12) shows that the MCS error can be reduced more when the ambiguity term is bigger. However, the increase in the ambiguity term may also cause an increase of the first term. Hence it is not enough to consider the diversity term alone. A

correct balance between the diversity and the accuracies of the base classifiers must be stressed. (4.12) helps us understand the combined effect of the base classifiers' errors and the diversity upon the error of the MCS. However, the model fails to identify the relationship between the generalization error (on unseen samples) and the error and the diversity on the training set.

Wolpert and Macready [Wolpert et al 1999] proposed several estimation methods on the error and ambiguity terms of the generalization error using the training set. For example:

$$Error = \sum_{l=1}^{L} \frac{w_l}{|D_l|} \left(\sum_{x \in D_l} \left(f_l(x) - F(x) \right)^2 \right), \tag{4.13}$$

$$Ambiguity = \sum_{l=1}^{L} \frac{w_l}{|D_l| - 1} \left(\sum_{x \in D_l} \left(f_l(x) - \sum_{k=1}^{L} w_k f_k(x) \right)^2 \right). \tag{4.14}$$

where D_l denotes the subset of D which is not used in the training of f_l and $|D_l|$ represents the number of training samples in D_l. Only D_l rather than D is used for the evaluation of f_l to reduce the over-fitting problem. However, there is no justification why this will work. In fact, if we sum up the estimated errors and the ambiguity terms, the result is roughly equal to the training error, but not necessarily the generalization error of the MCS.

4.1.5 Tumer and Ghosh Framework

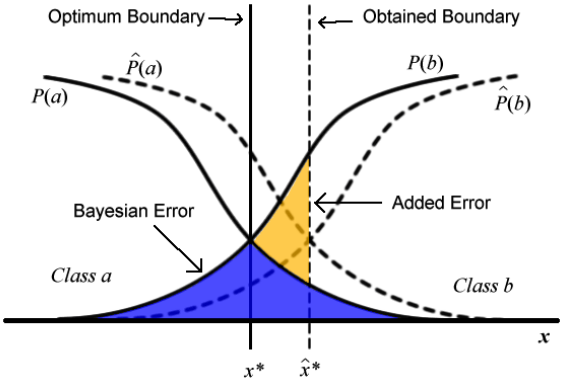


Figure 4.1 Decision boundaries and error associated with approximating the posteriori probabilities in Tumer and Ghosh's framework

A framework for analyzing the error of a MCS with the average fusion is proposed by Tumer and Ghosh [Tumer et al 1996a, 1996b]. A base classifier i approximates the posterior probability of the class ω as:

$$f_{i,\omega}(x) = \hat{P}_i(\omega \mid x) = P(\omega \mid x) + \eta_i(\omega \mid x), \tag{4.15}$$

where $P(\omega|x)$ is the true posteriori probability of class a and $\eta_i(\omega|x)$ is the posteriori probability estimation error of the base classifier i on the sample point x.

Figure (4.1) shows the posteriori probability obtained by a classifier, and the associated added error (E_{add}) region. The total error (E_{total}) of the classifier is defined as:

$$E_{total} = E_{add} + E_{Bayesian}, \tag{4.16}$$

where $E_{Bayesian}$ denotes the non-reducible Bayesian error. It means that this error cannot be reduced by any classifier.

This framework assumes the estimation errors on different classes are independent and identically distributed random variable [Tumer et al 1996a] with zero mean and variance $\sigma_{\eta_i}^2$. Also, the posteriori probabilities are assumed to be monotonic around the decision boundary. The added error of a classifier can be expressed as:

$$E_{add} = \frac{s\sigma_{\eta_i}^2}{2}. \tag{4.17}$$

where s is the difference between the derivatives of the two posteriors. When a MCS is considered with E_{add} and $\sigma_{\eta_i}^2$ being the same for all base classifiers, the added error of the MCS can be shown as:

$$E_{add}^{mcs} = \left(\frac{1+\delta(L+1)}{L}\right) E_{add}. \tag{4.18}$$

E_{add}^{mcs} denotes the added error of MSCs. δ is the correlation coefficient of the posteriori probability estimation error of the base classifiers. When $\delta = 1$, the outputs of all base classifiers are the same. The error of the MCS is equal to those of its base classifiers. If $\delta = 0$, the error of the base classifiers are statistically independent. E_{add}^{mcs} is L times smaller than E_{add}.

Roli and Fumera [Roli et al 2002] have extended this framework by easing the assumption on the same E_{add} and $\sigma_{\eta_i}^2$ for all base classifiers. The relation between a MCS with the average fusion and the weighted average fusion has been discussed. However, since the true posteriori probability is unknown for a classification problem, η_i, $\sigma_{\eta_i}^2$ and E_{add} cannot be calculated. This framework is for theoretical discussion only.

4.1.6 VC Dimension Model

Schapire et al [Freund et al 1996, 1997, Golea et al 1998] derives an upper bound for the generalization error of MCSs using weighted majority vote method for two class problems. There are two possible class IDs, $y \in \{-1, +1\}$. If the value of $f_i(x)$ is positive, x belongs to class +1. Otherwise, it is -1. A MCS consisting of binary base classifiers using weighted majority vote is defined as:

$$f^{mcs}(x) = \sum_{i=1}^{L} w_i sign\big(f_i(x)\big), \tag{4.19}$$

where $\sum_{i=1}^{L} w_i = 1$ and $w_i > 0$. The majority vote rule gives the wrong prediction only if $yf(x) < 0$. The margin in this case is defined as $yf(x)$. Assume that the base classifier space H is finite, and let $\delta > 0$. With probability at least $1 - \delta$ over the random choice of a given training set D, f^{mcs} satisfies the following bound for all $\theta > 0$:

$$P_{\Omega}(yf^{mcs}(x) \le 0) \le \left(P_{D}(yf^{mcs}(x) \le \theta) + O\left(\frac{1}{\sqrt{N}} \sqrt{\frac{\log N \log|H|}{\theta^2} + \ln\left(\frac{1}{\delta}\right)} \right) \right), \quad (4.20)$$

where N is the number of samples in D. It shows that the large margin of the MCS can reduce the upper bound of the MCS. This conclusion is applied to weight adjustment in Adaboost [Mason et al 1997, Reyzin et al 2006, Rudin et al 2004].

Golea et al [Golea et al 1998] improved this result with a tighter error bound. Different from Schapire's one which depends on the complexity of the most complex classifier, the generalization error proposed by Golea depends on the complexity term which involves the average VC Dimension of the classes of the base classifiers. Mason et al [Mason et al 1997] has applied the generalization error bound to the single hidden-layer threshold networks and decision trees. However, as indicated in [Freund et al 1996], this upper bound is very loose and can be infinite. Moreover, it is only meaningful when $N \geq$ 10000. Thus, this model is only good for theoretical discussion as well.

4.2 Derivation of L-GEM for MCSs (L-GEM^{MCS})

The existing error models for MCSs suffer from the weakness that either some components in the models are not computable or the error bound is very loose. The Localized Generalization Error Model for MCSs, abbreviated as L-GEM^{MCS}, which aims to overcome such a weakness, is proposed and derived in this session.

L-GEM^{MCS} is the extension of the L-GEM for single classifiers to the MCSs. The weighted average fusion is selected for the L-GEM^{MCS} since it is a commonly used fusion method. For instance, the Bagging and Boosting technique [Breiman 1994, 1996, Derbeko et al 2002, Fumera et al 2008, Shirai et al 2008, Skurichina et al 2000, Wolpert et al 1999], also uses the weighted average as its fusion method.

We recall here the definition of the Local Generalization Error Bound (R^*_Q) of a classifier f in Q neighborhood as given in Equation (1.7):

$$R_Q^* = \left(\begin{array}{l} \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f(x_i) - F(x_i))^2 \cdot p_Q(x) dx \right)} \\ + \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f(x_i) - f(x))^2 \cdot p_Q(x) dx \right)} + A \end{array} \right)^2 . \quad (4.21)$$

The first two terms are the training error and the sensitivity, and the last term is constant for a given dataset. For a MCS function, the error bound becomes:

$$R_Q^*$$

$$= \left(\begin{array}{l} \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f(x_i) - F(x_i))^2 \cdot p_Q(x) dx \right)} \\ + \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f(x_i) - f(x))^2 \cdot p_Q(x) dx \right)} + A \end{array} \right)^2$$

$$= \left(\begin{array}{l} \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f^{mcs}(x_i) - F(x_i))^2 \cdot p_Q(x) dx \right)} \\ + \sqrt{\sum_{i=1}^{N} \left(\int_{Q_i} (f^{mcs}(x_i) - f^{mcs}(x))^2 \cdot p_Q(x) dx \right)} + A \end{array} \right)^2 . \quad (4.22)$$

A MCS with the weighted average fusion is defined as:

$$f^{mcs}(x) = \sum_{l=1}^{L} w_l f_l(x), \quad (4.23)$$

where w_l is the weight assigned to the l^{th} base classifier, $\sum_{l=1}^{L} w_l = 1$ and $f_l(\ )$ is the output of the l^{th} base classifier.

By assuming x is uniformly distributed in the Q neighborhood and substituting Equation (4.23) to (4.22), the training error becomes:

$$\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}\left(f^{mcs}(x_i)-F(x_i)\right)^2 p_Q(x)dx\right)}$$

$$=\sqrt{\underset{D}{E}\left(\left(f^{mcs}(x_i)-F(x_i)\right)^2\right)}$$

$$=\sqrt{\underset{D}{E}\left(\left(\sum_{l=1}^{L} w_l f_l(x_i)-F(x_i)\right)^2\right)}$$

$$=\sqrt{\underset{D}{E}\left(\left(\sum_{l=1}^{L} w_l \left(f_l(x_i)-F(x_i)\right)\right)^2\right)}$$

$$=\sqrt{\underset{D}{E}\left(\left(\sum_{l=1}^{L}\sum_{m=1}^{L} w_l w_m \left(f_l(x_i)-F(x_i)\right)\left(f_m(x_i)-F(x_i)\right)\right)\right)}$$

$$=\sqrt{\begin{pmatrix}\sum_{l=1}^{L}\left[w_l^2 \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right)\right] \\ +2\sum_{l=1}^{L}\sum_{m=l+1}^{L}\left[w_l w_m \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)\left(f_m(x_i)-F(x_i)\right)\right)\right]\end{pmatrix}}. \quad (4.24)$$

Similarly, the sensitivity term can be expressed as:

$$\sqrt{\sum_{i=1}^{N}\left(\int_{Q_i}\left(\sum_{l=1}^{L} w_l\left(f_l(x)-f_l(x_i)\right)\right)^2 p_Q(x)dx\right)}$$

$$=\sqrt{\begin{pmatrix}\sum_{l=1}^{L}\left[w_l^2 \underset{D}{E}\left(\underset{Q_l}{E}\left(\left(f_l(x)-f_l(x_i)\right)^2\right)\right)\right] \\ +2\sum_{l=1}^{L}\sum_{m=l+1}^{L}\left[w_l w_m \underset{D}{E}\left(\underset{Q_l}{E}\left(\left(f_l(x)-f_l(x_i)\right)\left(f_m(x)-f_m(x_i)\right)\right)\right)\right]\end{pmatrix}}. \quad (4.25)$$

Finally, R^*_Q of the MCSs is given by:

$$R_Q^* = \left(\begin{array}{l} \sqrt{\sum_{l=1}^{L}\left[w_l^2 \underset{D}{E}\left((f_l(x_i)-F(x_i))^2\right)\right] + 2\sum_{l=1}^{L}\sum_{m=l+1}^{L}\left[w_l w_m \underset{D}{E}\left((f_l(x_i)-F(x_i))(f_m(x_i)-F(x_i))\right)\right]} \\ + \sqrt{\sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))^2\right)\right) + 2\sum_{l=1}^{L}\sum_{m=l+1}^{L} w_l w_m \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))(f_m(x)-f_m(x_i))\right)\right)} + A \end{array} \right)^2 . \quad (4.26)$$

Let

$$Err^{base} = \sum_{i=1}^{L} w_i^2 \underset{D}{E}\left((f_i(x_i)-F(x_i))^2\right),$$

$$Err^{div} = \sum_{l=1}^{L}\sum_{m=l+1}^{L}\left[w_l w_m \underset{D}{E}\left((f_l(x_i)-F(x_i))(f_m(x_i)-F(x_i))\right)\right],$$

$$Sen^{base} = \sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))^2\right)\right) \text{ and}$$

$$Sen^{div} = \sum_{l=1}^{L}\sum_{m=l+1}^{L} w_l w_m \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))(f_m(x)-f_m(x_i))\right)\right),$$

R^*_Q can be rewritten as:

$$R_Q^* = \left(\sqrt{Err^{base}+Err^{div}} + \sqrt{Sen^{base}+Sen^{div}} + A\right)^2 . \quad (4.27)$$

Similar to the L-GEM, L-GEM^{MCS} also contains three terms: training error, sensitivity and constant A. However, the training error and the sensitivity of L-GEM^{MCS} are more complex than similar terms in the L-GEM. The training error in L-GEM^{MCS} consists of two components. The first component, the base classifier training errors (Err^{base}), is the summation of the squared weight multiplied by the mean square training error of the base classifiers. Another component, the diversity of base classifier training errors (Err^{div}), is the summation of the product of the weighted training error of each pair of base classifiers.

Similarly, the sensitivity is also combined by two components: base classifier sensitivity (Sen^{base}) and diversity of base classifier sensitivity (Sen^{div}). Sen^{base} is the summation of squared weight multiplied by the mean square base classifiers' outputs differences between the unseen samples in Q_i and x_i. Sen^{div} is the summation of the product of the weighted outputs differences between the unseen samples in Q_i and x_i of each base classifier pair. The detail explanation of each component is discussed in the next section.

4.3 Components Discussion

The four terms that make up the L-GEM^{MCS} : Err^{base}, Err^{div}, Sen^{base} and Sen^{div}, are discussed in this section.

Base Classifier Training Error (Err^{base})

This term is defined as the summation of the squared weight multiplied by the mean square training error of base classifiers. The performance of the base classifiers will certainly affect the overall performance of the MCS. If a MCS is made up of very poorly performed base classifiers, it cannot expect a good performance in general. As a smaller R^*_Q value is preferred (to be discussed in Section 4.6), Err^{base} should be as small as possible. It should be noted that Err^{base} is a non-negative term. The optimal situation is when it is equal to zero. It means that each base classifier perfectly classifies each sample in the training set.

Diversity of Base Classifier Training Error (Err^{div})

As discussed in Section 4.1.1, one of the major advantages of considering the MCSs is the base classifiers can complement each other. The measurement of the differences among the base classifiers is called diversity. In L-GEM^{MCS}, the diversity of base classifier training errors is a kind of pair-wise diversity. This is because it considers a pair of classifiers at each time. Err^{div} measures how much difference in the errors made by the base

classifiers on the training set. For example, if the l^{th} classifier estimates the output of x_i higher than it should be, $f_l(x_i)$-$F(x_i)$ will be positive. If the m^{th} classifier makes the same estimation, then $f_m(x_i)$-$F(x_i)$ is also positive. The product of $f_l(x_i)$-$F(x_i)$ and $f_m(x_i)$-$F(x_i)$ is positive. As a result, the value of the diversity of the base classifier training errors becomes bigger. This is not a good situation since both classifiers make the same mistake on the same sample. In the contrast, if the m^{th} classifier estimates the output of x_i lower than it should be, then the product of $f_l(x_i)$-$F(x_i)$ and $f_m(x_i)$-$F(x_i)$ becomes negative. This will be a preferable situation since it causes the Err^{div} to be smaller. Different from Err^{base}, Err^{div} can be negative. The value of R^*_Q can be reduced if Err^{div} is smaller than 0. As a result, it is desirable to have the base classifiers making different errors on the training set.

Base classifier sensitivity (Sen^{base})

Sen^{base} is the summation of squared weight multiplied by the mean square base classifiers' outputs differences between the unseen samples in Q_i and x_i. The sensitivity measure is defined as the classifier output differences between the training sample and the unseen samples in its Q-neighborhood. It measures how sensitive the classifier output is to the input change. Generally, a classifier is not expected to have a good performance if it is very sensitive. This is because in most cases, the outputs of two similar inputs should not be very different. Moreover, when a dataset is collected, the actual value of a sample may be slightly different from the one being collected. A sensitive classifier will have a poor performance in this noise situation. Similar to the Err^{base}, Sen^{base} is a non-negative term. It means that smaller Sen^{base} is preferable.

Diversity of base classifier sensitivity (Sen^{div})

Similar to the Err^{div}, the Sen^{div} measures the relationship among the sensitivities of the base classifiers. Sen^{div} is also a pair-wise diversity measure and considers two classifiers at each time. It is the summation of the product of the weighted outputs differences between

the unseen samples in Q_i and x_i of each base classifier pair. It will be negative if the sensitivities of base classifiers are different on unseen samples. For example, if the output of the l^{th} classifier on x is higher than its output on x_i, $f_l(x)$-$f_l(x_i)$ is a positive value. An undesired situation will occur if another classifier has the same behavior on x since it will cause the MCS much more fluctuation at that point. The preferable case is that another classifier outputs a smaller value, which means $f_m(x)$-$f_m(x_i)$ is negative. The fluctuation effect can then be canceled out. Therefore, a negative Sen^{div} is preferred.

These four terms are dependent on each other. The change in one term may affect the values of the others. For example, if the variability of the individual error increases (Err^{div} decreases), then Err^{base} will also increase. Therefore, the L-GEM^{MCS} indicates that a good MCS should not consider any term individually. A right balance between these four terms must be considered. In summary, a MCS will have a smaller R^*_Q if its base classifiers are accurate (small Err^{base}) and stable (small Sen^{base}). Under the unfortunate situation where the base classifiers are inaccurate and unstable, it is hoped that their errors will be complementary to each other.

4.4 Characteristics of L-GEM^{MCS}

4.4.1 An Extension of the L-GEM

The L-GEM^{MCS} is an extension of the L-GEM for single classifiers. When $L = 1$, the Err^{div} and Sen^{div} are equal to 0 (Equation (4.26)). Then R^*_Q of L-GEM^{MCS} becomes:

$$R_Q^* = \left(\begin{array}{l} \sqrt{\sum_{l=1}^{L}\left[w_l^2 \underset{D}{E}\left((f_l(x_i)-F(x_i))^2\right)\right] + 2\sum_{l=1}^{L}\sum_{m=l+1}^{L}\left[w_l w_m \underset{D}{E}\left((f_l(x_i)-F(x_i))(f_m(x_i)-F(x_i))\right)\right]} \\ + \sqrt{\sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))^2\right)\right) + 2\sum_{l=1}^{L}\sum_{m=l+1}^{L} w_l w_m \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))(f_m(x)-f_m(x_i))\right)\right)} + A \end{array} \right)^2$$

$$= \left(\sqrt{\left(\sum_{l=1}^{1} w_l^2 \underset{D}{E}\left((f_i(x_i)-F(x_i))^2\right)\right)} + \sqrt{\sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))^2\right)\right)} + A \right)^2$$

$$= \left(\sqrt{\underset{D}{E}\left((f_l(x_i)-F(x_i))^2\right)} + \sqrt{\underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x)-f_l(x_i))^2\right)\right)} + A \right)^2 . \quad (4.28)$$

This result is equal to Equation (1.7).

4.4.2 Relation between L-GEM^{MCS} and Existing MCS Error Models

Bias, Variance and Covariance Decomposition describes the average error of a classifier on different datasets. The three components, which are bias, variance and covariance, cannot be calculated. However, R_Q^* provides the error bound of a trained classifier on a particular dataset. Each term in R_Q^* is computable.

Tumer and Ghosh model describes the relationship between the added error of a MCS and its base classifiers. It does not provide any information on the relation between the generalization and training error. In contrast, L-GEM^{MCS} expresses the generalization error bound as the summation of training error, sensitivity and a constant. Moreover, the added error in Tumer and Ghosh model cannot be calculated.

VC Dimension Model provides an error upper bound. However, this bound is for classifiers with a particular architecture and for the entire input space. These are different

from L-GEM^{MCS} which provides an error upper bound of a trained MCS in the *Q*-neighborhood.

Ambiguity Decomposition does not provide any information on the relation between the generalization and training error. The error of the MCS is broken down into Bias and Ambiguity terms. The relationship between these two terms and L-GEM^{MCS} are shown as follows. Using Equation (4.12), the training error of a MCS can be expressed as:

$$\underset{D}{E}\left(\left(f^{mcs}(x_i)-F(x_i)\right)^2\right)=\sum_{l=1}^{L} w_l \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right)-\sum_{l=1}^{L} w_l \underset{D}{E}\left(\left(f_l(x_i)-f^{mcs}(x_i)\right)^2\right). \quad (4.29)$$

Let $Error=\sum_{l=1}^{L} w_l \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right)$ and $Ambi=\sum_{l=1}^{L} w_l \underset{D}{E}\left(\left(f_l(x_i)-f^{mcs}(x_i)\right)^2\right)$. We can show that:

$$Error - Ambi = \text{Err}^{\text{base}} + \text{Err}^{\text{div}}$$

$$(\text{Error} - \text{Err}^{\text{base}}) - \text{Ambi} = \text{Err}^{\text{div}}, \quad (4.30)$$

$$\begin{aligned}
&\text{Error} - \text{Err}^{\text{base}} \\
&= \sum_{l=1}^{L} w_i \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right)-\sum_{l=1}^{L} w_i^2 \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right) \\
&= \sum_{l=1}^{L} w_i(1-w_i)\underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)^2\right) \\
&\geq 0. \quad (4.31)
\end{aligned}$$

Equation (4.30) shows that a bigger *Ambi* implies a smaller Err^{div}. When *Ambi* is equal to (*Error* - Err^{base}), Err^{div} is equal to 0. When *Ambi* is bigger (smaller) than (*Error* - Err^{base}), Err^{div} will become negative (positive) respectively. In Bias Ambiguity Decomposition, a MCS with large ambiguity is preferred. It is in agreement with the

conclusion in 4.3 which prefers a MCS with smaller Err^{div} when the sensitivity terms are ignored.

4.4.3 The Limiting Cases

The parameter q in the L-GEM^{MCS} controls the size of the local neighborhood (Q). For the special case when $q = 0$, Q only contains the training samples ($Q = D$). It follows that both Sen^{base} and Sen^{div} become zero and R^*_Q will be determined by Err^{base} and Err^{div} only. R^*_Q becomes the training error (R_{emp}) of the MCS.

When q is bigger, the importance of the training error of the MCS decreases since the Q neighborhood will become larger. When $q \rightarrow \infty$, $Q \rightarrow \Omega$, where Ω denotes the entire input space. Sen^{base} and Sen^{div} would become the dominating factors of R^*_Q and the effect of Err^{base} and Err^{div} will be relatively smaller. When $Q \rightarrow \Omega$, R^*_Q becomes the upper bound of the MCS generalization error in the entire input space (R_{ture}).

4.4.4 Independent of Training Method

L-GEM^{MCS} does not depend on the training method of the base classifiers. R^*_Q can be calculated for any trained MCS. This property of the L-GEM^{MCS} makes it an useful characteristic of a MCS since the base classifiers may be trained by different training methods to increase the diversity.

4.4.5 Time Complexity of L-GEM^{MCS}

It can be easily shown that the time complexity of the L-GEM^{MCS} is $O(nNL(L+1)/2)$, where n is the number of features of a sample, L is the number of base classifiers and N is number of samples.

4.4.6 Complexity of the MCSs

When a MCS has a "non-smooth" separating surface, its output changes may be very sensitive to input changes. The two terms, Sen^{base} and Sen^{div} in the L-GEM^{MCS}, represent the fluctuations of the MCS outputs in a Q neighborhood. The value of the sum of these two terms is large when the output differences between unseen samples and the training samples in the Q neighborhood are large. Therefore, the sensitivity terms in the L-GEM^{MCS} could be used to describe the complexity of a MCS.

4.4.7 Limitation

L-GEM^{MCS} gives an estimate on the generalization error bound in a Q neighborhood of the training set. However, it suffers from the same problem as all other methods do, i.e., when the training set is not a good representation of the input population, its estimation may not be useful.

4.5 Sensitivity Measures (Sen^{base} and Sen^{div})

An effective way of computing the sensitivity measure is proposed in this section. When $||\Delta x||$ is small, the classifier output can be approximated by

$$f(x+\Delta x) \approx f(x) + \left(\frac{\partial f}{\partial x}\right)^T \Delta x + \frac{1}{2}\Delta x^T H \Delta x \quad , \tag{4.32}$$

where H denotes the Hessian matrix with element $h_{ij} = \partial^2 f / (\partial x_i \partial x_j)$ and H is assumed to be zero approximately for the surfaces with small curvature. Sen^{base} and Sen^{div} become:

$$\underset{Q_i}{E}\left(\left(f(x) - f(x_i)\right)^2\right)$$

$$= \underset{Q_i}{E}\left(\left(f(x_i + \Delta x) - f(x_i)\right)^2\right)$$

$$\approx \underset{Q_i}{E}\left(\left(f(x_i)+\left(\frac{\partial f}{\partial x_i}\right)^T \Delta x - f(x_i)\right)^2\right)$$

$$= \underset{Q_i}{E}\left(\left(\frac{\partial f}{\partial x_i}\right)^T \Delta x \Delta x^T \left(\frac{\partial f}{\partial x_i}\right)\right)$$

$$= \frac{q^2}{3}\left(\frac{\partial f}{\partial x_i}\right)^T \left(\frac{\partial f}{\partial x_i}\right), \tag{4.33}$$

$$\underset{Q_i}{E}\left(\left(f_l(x)-f_l(x_i)\right)\left(f_m(x)-f_m(x_i)\right)\right)$$

$$= \underset{Q_i}{E}\left(\left(f_l(x_i+\Delta x)-f_l(x_i)\right)\left(f_m(x_i+\Delta x)-f_m(x_i)\right)\right)$$

$$\approx \underset{Q_i}{E}\left(\left(f_l(x_i)+\left(\frac{\partial f_l}{\partial x_i}\right)^T \Delta x - f_l(x_i)\right)\left(f_m(x_i)+\left(\frac{\partial f_m}{\partial x_i}\right)^T \Delta x - f_m(x_i)\right)^2\right)$$

$$= \underset{Q_i}{E}\left(\left(\frac{\partial f_l}{\partial x_i}\right)^T \Delta x \Delta x^T \left(\frac{\partial f_m}{\partial x_i}\right)\right)$$

$$= \frac{q^2}{3}\left(\frac{\partial f_l}{\partial x_i}\right)^T \left(\frac{\partial f_m}{\partial x_i}\right). \tag{4.34}$$

Δx is assumed to be uniformly distributed in Q_i. Hence, Δx is zero mean and uncorrelated. The mean of $\Delta x \Delta x^T$ in Q_i is $\delta^2 I$, where δ^2 is equal to $q^2/3$.

One of the advantages of this proposed calculation is its low complexity. This advantage is especially important to the MCS since most, if not all, of its computational efforts increases with respect to its number of base classifiers. The complicated calculation makes the estimation method not practical. Equations (4.33) and (4.34) show that both Sen^{base} and Sen^{div} are in terms of $\partial f/\partial x$. A computation reduction could be realized if $\partial f/\partial x$ for

each base classifier can be calculated and stored first. For instance, when Sen^{div} is computed, there is no need to do any pairwise sensitivity computation. One only needs to retrieve the values of $\partial f/\partial x$.

Considering the special case of a MLP Neural Network with one hidden layer:

$$f^{MLP}(x_i) = \sum_{m=1}^{M} \left(\frac{\alpha_{H_m}}{1 + \exp\left(- \sum_{j}^{n} \alpha_{I_{jm}} x_{i_j} \right)} \right) \tag{4.35}$$

where M denotes the number of hidden neurons, α_{Hm} denotes the weight of the m^{th} neuron, α_{Ijm} denotes the weight between the j^{th} input feature and the m^{th} neuron, x_i denotes the sample i and x_{ij} is the j feature of x_i.

Thus, $\partial f^{MLP}/\partial x_i$ and df^{MLP}/dx_{it} are defined as:

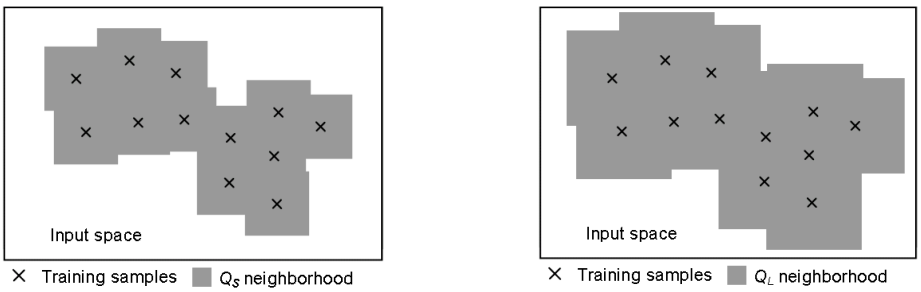
$$\frac{\partial f^{MLP}}{\partial x_i} = \left[\frac{df^{MLP}}{dx_{i_1}}, \frac{df^{MLP}}{dx_{i_2}}, ..., \frac{df^{MLP}}{dx_{i_n}} \right]^T, \tag{4.36}$$

$$\frac{df^{MLP}}{dx_{i_t}} = \sum_{m=1}^{M} \left(\frac{\alpha_{H_m} \alpha_{I_{tm}} \exp\left(- \sum_{j=1}^{n} \alpha_{I_{jm}} x_{i_j} \right)}{\left(1 + \exp\left(- \sum_{j}^{n} \alpha_{I_{jm}} x_{i_j} \right) \right)^2} \right). \tag{4.37}$$

The calculations of $\partial f^{RBF}/\partial x_i$ and df^{RBF}/dx_{it} of a RBF Network are given by Equations (3.15) and (3.16).

4.6 Comparing MCSs Using L-GEM^{MCS}

Given a pool of trained MCSs, it is important to know which MCS will have a better generalization capability? In this section we want to show that L-GEM^{MCS} can be used as a reasonable evaluation criteria to help answer this question.



a) small value of q b) large value of q

Figure 4.2 Q neighborhoods with different values of q on the same dataset

Given a training set (D) and two MCSs (f^{mcs1} and f^{mcs2}), and for the same Q neighborhood, if R^*_Q of f^{mcs1} is smaller than R^*_Q of f^{mcs2}, it is expected that f^{mcs1} has better generalization ability than f^{mcs2}. It should be noted that q is the same for the Q neighborhood of f^{mcs1} and f^{mcs2}. Otherwise, R^*_Q will be for different regions and the comparison will be meaningless. For example, Figure (4.2) shows two Q neighborhoods, Q_S and Q_L ,corresponding to different values of q on the same dataset. If R^*_{QS} of f^{mcs1} is smaller than R^*_{QL} of f^{mcs2}, no conclusion can be drawn since Q_S and Q_L covers unseen samples in different regions. In general, R^*_{QL} is bigger than R^*_{Qs} since more unseen samples are considered in R^*_{QL}.

MCSs can be compared in another way by using the L-GEM^{MCS}. We can fix the R^*_Q and compare the sizes of q of different MCSs. For any given value of R^*_Q, if f^{mcs1} can cover a larger region than f^{mcs2}, we can expect the generalization ability of f^{mcs1} is better than f^{mcs2}.

By using the definition of Sen^{base} and Sen^{div} in Equations (4.33) and (4.34), R^*_Q in L-GEM becomes:

$$R_Q^* = \left(\begin{array}{l} \sqrt{Err^{base} + Err^{div}} \\ + \sqrt{\begin{array}{l} \sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\frac{q^2}{3} \left(\frac{\partial f}{\partial x_i} \right)^T \left(\frac{\partial f}{\partial x_i} \right) \right) \\ + 2 \sum_{l=1}^{L} \sum_{m=l+1}^{L} w_l w_m \underset{D}{E}\left(\frac{q^2}{3} \left(\frac{\partial f_l}{\partial x_i} \right)^T \left(\frac{\partial f_m}{\partial x_i} \right) \right) \end{array}} + A \end{array} \right)^2 . \quad (4.38)$$

As A is a constant for a given set, it can be ignored in the MCS comparison. For any given R^*_Q, q can be found by:

$$q = \frac{\sqrt{R_Q^*} - \sqrt{Err^{base} + Err^{div}}}{\sqrt{\sum_{l=1}^{L} w_l^2 \underset{D}{E}\left(\frac{1}{3} \left(\frac{\partial f_l}{\partial x_i} \right)^T \left(\frac{\partial f_l}{\partial x_i} \right) \right) + 2 \sum_{l=1}^{L} \sum_{m=l+1}^{L} w_l w_m \underset{D}{E}\left(\frac{1}{3} \left(\frac{\partial f_l}{\partial x_i} \right)^T \left(\frac{\partial f_m}{\partial x_i} \right) \right)}} . \quad (4.39)$$

It should be noted (4.39) is valid when the square root of R^*_Q is larger than the square root of the sum of Err^{base} and Err^{div}. This condition is reasonable since the error of a MCS on unseen samples in the Q neighborhood should be larger than its training error, The denominator of Equation (4.39) must also be bigger than zero. The denominator is equal to zero when all $\partial f_l / \partial x_i$ are zero, where $l = 1..L$ and $i = 1..N$. In this situation, all base classifiers in a MCS are stable and both Sen^{base} and Sen^{div} are equal to 0. This ideal situation rarely happens in practice.

4.6.1 Experiments

In this section, the performance on MCS comparison using L-GEM^{MCS} is discussed experimentally. A pool of MCSs with different number of base classifiers is trained. The best MCS will be chosen according to different selection criteria. R^*_Q in L-GEM^{MCS} will be compared with other selection criteria: 1) Training MSE (T-MSE), 2) Training Classification Error (T-CE), 3) Training MSE of 5-CV (5CVT-CE), 4) Training

Classification Error of 5-CV (5CVT-MSE) and, 5) Bias Ambiguity Decomposition method (BAD) [Wolpert et al 1999].

MLP Neural Network and RBF Network are used as the base classifiers. For the MLP Neural Network, it has only one hidden layer and the number of hidden neurons is randomly selected from two to fifty. The weights are determined by gradient descent [Kiernan et al 1996]. For the RBF network, its number of neurons is selected randomly from two to fifty. The center and width of the neurons are determined by K-mean [Kiernan et al 1996] and the K-nearest-neighbor algorithm [Musavi et al 1992] respectively. The weight is calculated using the Singular Value Decomposition (SVD) method [Mak et al 1998]. To diversify the base classifiers in a MCS, Bagging method [Breiman 1996] is applied. Each base classifier is assigned a different training set which is randomly selected from the original training set with replacement.

Ten MCSs are trained for each number of base classifier, L = 1..50. The pool contains 500 MCSs in total. Each MCS consists of half RBF networks and half MLP Neural Networks. Average and weighted average are used as the fusion method. The weights of the weighted average fusion method are determined according to the training errors of the base classifiers. The parameter q in the L-GEM^{MCS} is determined by cross valuation.

4.6.1.1 Experiment on Benchmark Datasets

Table (4.4) shows twelve datasets selected from the UCI machine learning repository [MLR] and Intelligent Data Analysis Group [DAG]. They cover a wide range of applications involving two-class and multi-class problems. Each dataset is equally divided into two parts randomly: training and testing. The experiment generates twenty independent runs for each pair of dataset. Only samples in the training set are used during training. The

samples in the testing set are reserved to evaluate the performances of the trained classifiers. The inputs of all samples are normalized to [0, 1].

Table 4.4 Twelve Datasets

Dataset	Short Name	# Class	# Sample	# Feature
Breast Cancer Wisconsin	Canc	2	569	32
Car Evaluation	Car	4	1728	6
Connectionist	Conn	2	208	60
Credit Approval	Cred	2	690	15
Pima Indians Diabetes	Pima	2	768	8
Solar Flare	Solar	2	1066	9
Glass Identification	Glass	7	214	10
Heart	Heart	2	270	13
Ionosphere	Iono	2	351	33
Image Segmentation	Img	7	2310	19
Spambase	Spam	2	4601	57
Waveform	Wave	3	5000	21

The results are shown in Tables (4.5) and (4.6). Each dataset contains two rows. The first row represents the average percentage of classification accuracy and its variances of the testing sets over twenty independent runs. The average number of base classifiers in MCSs over twenty independent runs is shown on the second row. Each column represents a different selection method for MCS. The Student's t-test is applied to examine the statistical significance of the performance made by L-GEM^{MCS} against the other methods. When the absolute t-value is larger than 2.02 in each experiment, the result is significant at the 95% probability level. The value is bolded and underlined in the cell if the performance of L-GEM^{MCS} is better than that specific method at a 95% significance level.

Table (4.5) demonstrates the MCSs selected by L-GEM^{MCS} perform better than the ones selected by other methods when the average fusion is used. In the Car Evaluation, Connectionist and Credit Approval datasets, L-GEM^{MCS} is 2% better than other methods on average. The performance of the Cross Validation methods (5CVT-CE and 5CVT-MSE) is worse than Training Error based methods (T-CE and T-MSE). BAD performs the worst in

this experiment, especially in Car Evaluation and Image Segmentation dataset. The performance of MCSs using the weighted average fusion method is shown in Table (4.6). Similar to Table (4.5), L-GEM^{MCS} outperforms other methods about 2%. It should be noted that the performance of 5CVT-CE and 5CVT-MSE slightly improve comparing with Table (4.5). The accuracy of MCSs selected by the Cross Validation methods and similar to the one selected by T-CE and T-MSE. The MCS selected by BAD also has the lowest accuracy and it is about 1% lower than average accuracy.

Table 4.5 L-GEM^{MCS} VS Other Methods
Average Classification Accuracy, Variance of Testing Set and Number of Base classifiers of MCSs using average over Twenty Independent Runs

Average		**L-GEM^{MCS}**	**T-CE**	**T-MSE**	**BAD**	**5CVT-CE**	**5CVT-MSE**
Canc	Acc (%)	97.16±0.00	**96.68±0.00**	**96.59±0.01**	**96.33±0.01**	**96.33±0.00**	**96.68±0.00**
	L	25.32	28.03	42.22	45.55	18.04	28.05
Car	Acc (%)	92.54±0.01	**91.80±0.00**	**91.74±0.00**	**89.8±0.01**	**91.10±0.00**	**91.34±0.00**
	L	39.25	36.25	46.30	46.53	19.53	33.50
Conn	Acc (%)	84.08±0.19	**80.78±0.23**	83.94±0.14	**80.30±0.15**	**81.51±0.11**	**80.78±0.14**
	L	20.50	9.54	36.02	42.25	19.00	24.36
Cred	Acc (%)	86.97±0.03	**85.23±0.01**	**85.30±0.02**	**84.94±0.01**	**84.87±0.02**	**84.94±0.04**
	L	37.25	32.54	44.25	42.03	21.25	31.25
Pima	Acc (%)	79.13±0.14	**76.50±0.01**	**76.83±0.02**	**75.33±0.03**	**76.18±0.01**	**76.70±0.02**
	L	42.23	23.04	40.51	46.54	21.00	31.34
Solar	Acc (%)	67.59±0.01	**66.79±0.02**	67.16±0.01	66.60±0.04	**66.55±0.02**	**66.32±0.03**
	L	17.49	13.04	47.77	47.25	13.79	25.75
Glass	Acc (%)	89.49±0.12	88.06±0.06	88.77±0.08	88.53±0.04	**87.58±0.04**	87.82±0.04
	L	29.00	19.53	44.02	38.06	21.75	41.65
Heart	Acc (%)	82.24±0.06	81.50±0.03	81.5±0.02	**80.39±0.04**	**80.76±0.01**	81.13±0.01
	L	26.34	37.77	46.76	48.53	22.01	29.20
Iono	Acc (%)	89.86±0.02	**88.06±0.01**	88.91±0.04	**88.49±0.01**	**87.77±0.01**	**88.06±0.04**
	L	31.50	36.38	43.50	36.74	20.75	30.75
Img	Acc (%)	88.72±0.02	87.34±0.10	87.58±0.11	**84.49±0.12**	**87.10±0.07**	**87.34±0.06**
	L	21.75	14.56	44.53	44.26	15.03	23.32
Spam	Acc (%)	89.25±0.01	**88.15±0.01**	**87.69±0.01**	**87.71±0.02**	**87.56±0.01**	**87.54±0.01**
	L	43.50	43.77	40.24	48.51	34.23	32.75
Wave	Acc (%)	88.18±0.00	**86.73±0.00**	**86.69±0.00**	**86.52±0.00**	**86.71±0.00**	**86.65±0.00**
	L	41.62	21.53	46.02	46.75	12.4	21.01
Average	Acc (%)	86.27	84.80	85.23	84.12	84.50	84.61
	L	31.31	26.33	43.51	44.42	19.90	29.41

Generally, the methods using MSE (L-GEM^{MCS}, T-MSE, BAD and 5CVT-MSE) tends to select the MCS which contains a large number of base classifiers than the methods using classification error (T-CE and 5CVT-CE). The average *L* in L-GEM^{MCS}, T-MSE, BAD and 5CVT-MSE, which is about 37, is large than T-CE and 5CVT-CE, which is about 23. However, since the sensitivity terms (Sen^{base} and Sen^{div}) of R^*_Q in L-GEM^{MCS} discourage

to the selection of complex MCSs, the number of base classifiers in the MCSs selected by L-GEM^{MCS}, which is about 31 in average, is relatively small among the MSE based methods.

Table 4.6 L-GEM^{MCS} VS Other Methods
Average Classification Accuracy, Variance of Testing Set and Number of Base classifiers of MCSs using weighted average over Twenty Independent Runs

Weighted Average		**L-GEM^{MCS}**	**T-CE**	**T-MSE**	**BAD**	**5CVT-CE**	**5CVT-MSE**
Canc	Acc (%)	97.42±0.11	97.47±0.00	96.85±0.01	97.65±0.01	**95.49±0.00**	**95.60±0.00**
	L	23.12	26.64	41.76	30.40	28.69	31.14
Car	Acc (%)	92.54±0.01	**91.31±0.01**	**91.60±0.00**	**87.83±0.02**	**90.14±0.00**	**90.69±0.00**
	L	36.79	19.81	44.56	32.12	21.09	32.13
Conn	Acc (%)	83.42±0.22	**79.81±0.14**	83.94±0.18	**78.60±0.18**	81.89±0.06	82.01±0.05
	L	23.59	22.24	38.15	24.37	18.92	28.15
Cred	Acc (%)	87.10±0.02	**84.94±0.06**	**85.30±0.02**	**84.94±0.03**	85.99±0.05	86.24±0.03
	L	28.94	22.79	45.80	17.79	21.23	24.87
Pima	Acc (%)	80.39±0.04	**76.83±0.04**	**76.70±0.01**	**77.02±0.01**	**77.07±0.02**	**76.62±0.01**
	L	35.71	12.12	41.57	14.90	22.85	40.62
Solar	Acc (%)	68.69±0.01	**67.69±0.02**	**66.98±0.01**	**65.99±0.02**	68.53±0.00	**67.81±0.00**
	L	17.18	21.45	46.32	26.55	22.95	41.37
Glass	Acc (%)	91.11±0.06	**88.06±0.06**	**85.68±0.10**	**85.91±0.11**	**88.45±0.00**	**87.47±0.00**
	L	27.23	7.86	44.70	33.59	24.64	27.42
Heart	Acc (%)	83.68±0.03	**80.39±0.03**	**81.31±0.02**	82.79±0.04	**81.50±0.01**	**82.23±0.02**
	L	28.39	36.14	45.31	10.31	23.27	22.84
Iono	Acc (%)	89.63±0.02	**88.49±0.00**	88.91±0.04	**87.91±0.03**	**87.33±0.01**	**86.55±0.01**
	L	33.34	37.67	43.68	18.47	29.64	34.03
Img	Acc (%)	89.54±0.11	**87.10±0.06**	88.06±0.14	**85.68±0.04**	**87.88±0.00**	**87.92±0.00**
	L	20.94	19.11	44.96	33.15	18.81	29.52
Spam	Acc (%)	89.31±0.01	**88.07±0.01**	**87.61±0.01**	**86.97±0.02**	**87.67±0.00**	**87.83±0.00**
	L	43.49	36.36	41.72	16.63	25.23	29.41
Wave	Acc (%)	89.37±0.08	**86.90±0.00**	**86.66±0.01**	**86.53±0.00**	**86.78±0.00**	**87.88±0.00**
	L	31.36	13.90	48.13	24.34	21.51	45.12
Average	Acc (%)	86.93	84.76	84.97	83.99	84.89	84.90
	L	29.17	23.01	43.89	23.55	23.24	32.22

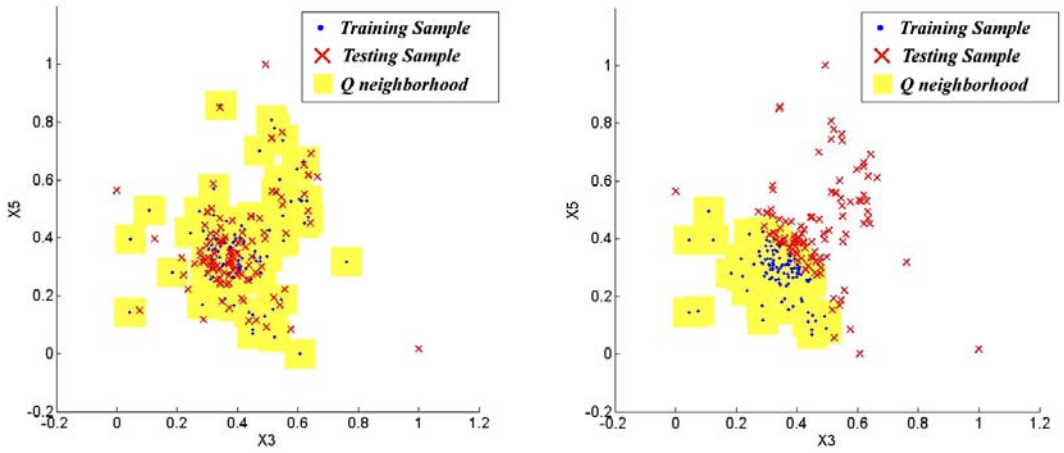
The average MCS selection times on twelve dataset over twenty independent runs are shown in Table (4.7). As expected, 5CVT-CE and 5CVT-MSE require the longest selection time since they need to train 5 times of MCSs. For example, if the MCS consists of 30 base classifiers, 150 base classifiers have to be trained. Definitely, cross validation is not a practical method for MCSs. The selection time of the L-GEM^{MCS} based method is longer than other methods since additional sensitivity terms are considered. The time required by the L-GEM^{MCS} is less than 2 times of T-CE and T-MSE.

Table 4.7 L-GEM^{MCS} VS Other Methods
Average Selection Time of MCSs on Twelve Datasets over Twenty Independent Runs

Method	**L-GEM^{MCS}**	**T-CE**	**T-MSE**	**BAD**	**5CVT-CE**	**5CVT-MSE**
Time	22.77	13.51	13.52	19.30	1331.77	1332.68

4.6.1.2 Experiment on a Biased Dataset

In general, a training set should be reasonable representative of the underlining classification problem. This means that the unseen samples should be similar to the training samples. Figure (4.3a) shows good representative training samples in the Glass dataset. The dots are the training samples and the crosses are the unseen samples. The shaded area denotes the Q neighborhood. In this case, even a small value of 0.1 for q, Q_S and Q_L near the training samples covers most of the testing samples. Figure (4.3b) shows training samples that are not good representatives of the problem. The training samples are all located on the left hand bottom corner of the input domain while the testing samples are mainly located in the upper right region. Hence most testing points cannot be covered by the Q neighborhood.



a) Random selection b) Biased selection

Figure 4.3 Distribution of Training and Testing sets divided by different selection for the glass Dataset

This section discusses the performances of different learning methods in their handling of a special situation when the training set is sampled poorly and cannot represent the classification problem. Glass dataset is again used in this experiment.

Tables (4.8) and (4.9) show the performance of MCSs selected by L-GEM^{MCS} and other selection methods when average and weighted average fusion method is used respectively. Refer to Tables (4.5) and (4.6), when the training and testing set are divided

randomly, the testing accuracy of all methods is about 88%. Unsurprisingly, the accuracies of all methods drop significantly, which is about 75% in average, for the biased dataset. Moreover, the variance for all methods for the biased dataset is larger due to the larger variations in the testing set. This result is in agreement with the findings in Section 2.4.3. When the training samples cannot represent the problem, no classifier is expected to perform well.

Table 4.8 L-GEM^{MCS} VS Other Methods
Average Classification Accuracy, Variance of Testing Set and Number of Base classifiers of MCSs using weighted average over Twenty Independent Runs

Average		**L-GEM^{MCS}**	**T-CE**	**T-MSE**	**BAD**	**5CVT-CE**	**5CVT-MSE**
Glass	Acc (%)	80.96±0.28	80.37±0.28	79.91±0.06	77.10±0.35	75.23±0.21	75.70±0.44
	L	19.34	13.54	44.43	46.03	20.52	27.53

Table 4.9 L-GEM^{MCS} VS Other Methods
Average Classification Accuracy, Variance of Testing Set and Number of Base classifiers of MCSs using weighted average over Twenty Independent Runs

Weighted Average		**L-GEM^{MCS}**	**T-CE**	**T-MSE**	**BAD**	**5CVT-CE**	**5CVT-MSE**
Glass	Acc (%)	79.83±0.11	79.84±0.35	79.91±0.07	77.10±0.53	64.06±0.34	62.79±0.31
	L	19.31	23.53	39.68	25.53	21.57	29.59

Conclusion

The experimental results show that the L-GEM^{MCS} still selects a better MCS than the other methods under the biased situation in term of testing accuracy. This indicates that the L-GEM^{MCS} can still pick up a MCS with relatively good generalization ability although most unseen samples are not located in the Q neighborhood. It should be noted that the performance of two 5-CV methods drastically drops 12% to 15%. Cross Validation methods try to estimate the generalization error of the MCSs using different subsets of the training samples. However, the bias training samples affect their estimations significantly.

4.7 Pilot Study on MCSs training using L-GEM^{MCS}

Chapter 2 shows that L-GEM can be applied as a single classifier training objective resulting in good experimental performances comparing with existing methods. Similarly, L-GEMMCS can be used as a training objective function for a MCS. In this situation, all base classifiers in a MCS will be trained at the same time to minimize the localized generalization error bound of the MCS.

Negative Correlation (NC) [Zanda et al 2007] is a well known MCS training method. In NC, the following objective function is used:

$$R_{NC} = \underset{D}{E}\left(\sum_{l=1}^{L} \left(f_l(x_i) - F(x_i) \right)^2 \right) + \lambda \underset{D}{E}\left(\sum_{l=1}^{L} \left(f_l(x_i) - f^{mcs}(x_i) \right) \left(\sum_{\substack{m=1 \\ m \neq l}}^{L} \left(f_m(x_i) - f^{mcs}(x_i) \right) \right) \right). \tag{4.40}$$

The first term in (4.40) is the sum of the training errors of the base classifiers and the second term is the diversity term which measures the differences between the errors of base classifiers. Therefore, NC minimizes the error and increases the error diversity of base classifiers of a MCS.

One possible choice for the L-GEM^{MCS} objective function is defined as:

$$R_Q^{'} = \left(\sqrt{Err^{base} + Err^{div}} + \sqrt{Sen^{base} + Sen^{div}} \right). \tag{4.41}$$

The terms under the first square root in (4.41) are similar to the terms in NC although the diversity terms are defined differently. The diversity in NC is defined intuitively while the diversity term in L-GEMMCS is closely related to the generalization error of the MCS. (4.41) has an additional term, namely, the sensitivity of MCSs.

The following training methods are compared experimentally: 1) single RBF network by $2PLR_Q$ (RBF $2PLR_Q$), 2) single RBF network by $3PLR_Q$ (RBF $3PLR_Q$), 3) K-Nearest Neighborhood (Single K-NN), 4) C4.5 Decision Tree (Single DT), 5) MCS with RBF network base classifier by L-GEM^{MCS} (RBF $2PLR_Q$), 6) MCS with RBF network base classifier by NC (RBF $2PLR_Q$), 7) MCS with K-Nearest Neighborhood (MCS K-NN) and 8) MCS with C4.5 Decision Tree (MCS DT). Each MCS contains 10 base classifiers. The base classifiers in MCS K-NN and MCS DT are trained by Bagging method [Breiman 1996]. Each base classifier is assigned a different training set randomly selected from the original training set with replacement. Parameters K, q and λ are determined by CV.

Four datasets are used from the UCI machine learning repository [MLR]. The experiment generates ten independent runs for each pair of dataset. Only samples in the training set are used during training. The samples in the testing set are reserved to evaluate the performances of the trained classifiers. The inputs of all samples are normalized to [0, 1].

Table (4.10) shows the experimental performances of different classifiers. Our first observation is that MCSs trained by L-GEM^{MCS} perform the best among all four MCS methods for all datasets. The second observation is that the performance of a single classifier could be improved by combining a number of them into a MCS. For example, in breast cancer dataset, the accuracy of a single K-NN improves by more than 4.5% when a number of K-NNs are combined.

Table 4.10 RBF L-GEM^{MCS} VS other classifier
Average Classification Accuracy of Testing Set over Ten Independent Runs

	Single Classifiers				**MCSs**			
	RBF $2PLR_Q$	**RBF $3PLR_Q$**	**Single K-NN**	**Single DT**	**RBF L-GEM^{MCS}**	**RBF NC**	**MCS K-NN**	**MCS DT**
Breast Cancer	96.03	96.72	91.24	92.23	97.65	96.01	95.71	96.48
Car	92.18	96.72	90.02	91.32	98.00	96.45	94.63	94.92
Connectionist	80.01	82.37	77.24	77.98	83.35	81.48	79.82	80.79
Credit Approval	87.24	87.32	85.78	87.34	88.35	86.90	88.14	87.51
Dermatology	97.99	98.19	94.64	96.57	98.23	97.88	96.65	97.56
Average	90.69	92.26	87.78	89.09	93.12	91.74	90.99	91.45

4.8 Conclusion

The L-GEM^{MCS} is proposed and derived in this chapter. L-GEM^{MCS} contains four terms: Err^{base}, Err^{div}, Sen^{base} and Sen^{div}. Err^{base} and Sen^{basev} consider the error and sensitivity for each base classifier, while Err^{div} and Sen^{div} measure the relation of error and sensitivity between each pair of base classifiers. Similarities and differences between L-GEM^{MCS} and the Bias Ambiguity Decomposition are presented. The relationship between Err^{base}, Err^{div} and the Ambiguity term is analyzed and the different characteristics of the L-GEM^{MCS} are discussed. A more effective computation of Sen^{base} and Sen^{div} is proposed and is applied to the RBF Network and the MLP Neural Network with one hidden layer.

L-GEM^{MCS} is applied as criterion to select the best one from a pool of trained MCSs. The experimental result shows that when the average and the weighted average fusions are used, the MCSs selected by L-GEM^{MCS} outperform the ones chosen by other five selection methods in terms of testing accuracy. Moreover, the number of base classifiers in MCSs selected by L-GEM^{MCS} is smaller than other MSE based selection methods, such as T-MSE, BAD and 5CVT-MSE. The experimental result also shows that L-GEM^{MCS} is competitive with other methods in terms of time complexity.

CHAPTER 5
BASE CLASSIFIER SUBSET SELECTION USING L-GEM^{MCS}

A long standing problem that exist in the study of the MCSs is the determination of τ, the number of base classifiers, to form the MCS, given that a pool of L base classifiers are available. Many ad-hoc rules or rules-of-thumb are available. In this chapter we will explore the use of L-GEM^{MCS} as an evaluation criteria for selecting a subset of τ base classifiers by adding one at a time, assuming that the additional base classifier will contribute to the performance improvement of the MCS.

The success of online classification problems is critically dependent on the value of τ [Margineantu et al 1997, Prodromidis et al 2001]. The selection of τ base classifiers out of a pool of L is referred to the "pruning" problem. Many studies show that a pruned MCS outperforms the original one in terms of accuracy [Aksela 2003, Caruana et al 2004, Margineantu et al 2009, Roli et al 2001, Sharkey et al 2000, Zhou et al 2002, 2003]. However, the optimal base classifier subset selection is a NP-hard problem. For instance, a MCS with L base classifiers, the number of non-empty base classifier subset is 2^L-1.

One practical approach to this problem is to use a greedy search to find a sub-optimal solution to this base classifier subset selection problem. A MCS is constructed by adding a base classifier one at a time. In each iteration, the base classifier which is expected to improve the generalization ability of the MCS the most is selected.

The selection criteria used in the current methods are either based on diversity measure [Giacinto et al 2001a, 2001b, Margineantu et al 1997] or intuitive thinking [Banfield et al 2005, Martınez-Munoz et al 2004]. These methods offer no theoretical justifications for their claims. Our proposed L-GEM^{MCS} ,which estimates the error bound of a MCS on unseen samples located within a neighborhood of the training samples, is intuitively logical and conceptually appealing to be used as a criteria for base classifier subset selection.

Section 5.1 presents a review of base classifier subset selection methods using a greedy search. The new subset selection method using L-GEM^{MCS} is presented and discussed in Section 5.2. Experimental results are shown and analyzed in Section 5.3 and Section 5.4 concludes this chapter.

5.1 Base Classifier Subset Selection Methods

The following notations are needed to facilitate discussion on the base classifier selection methods:

H	a set of trained base classifier
H_S	a set of selected trained base classifier
$C_H H_S$	a set of unselected trained base classifier
f_H^{mcs}	a MCS combined by base classifiers in H

Reduce Error (RE) method [Margineantu et al 1997, 2006] may be the simplest for base classifier subset selection. This method calculates the training error of $f_{H_s \cup f}^{mcs}$,

where $f \in C_H H_s$. The base classifier which yields the smallest training error of $f^{mcs}_{H_s \cup f}$ will be selected.

Concurrency Thinning (CT) [Banfield et al 2005] incorporates the base classifier which is the most complementary to the MCS. The complementary is represented by a score. A base classifier is given a high score for obtaining a correct decision, especially when the decision of the MCS is incorrect. The score of a classifier is deducted when both the ensemble and classifier are in correct. The detail algorithm is shown in Figure (5.1). The concept of Complementarily Measure (CM) [Martınez-Munoz et al 2004] is similar to Concurrency Thinning. However, Complementarily Measure only counts the number of times when the MCS is wrong but the base classifier is correct.

For each $f_l \in C_H H_S$
    For each $x_i \in D$
        If $f^{mcs}_{H_s}$ correct and f_l correct
            Score$_l$ = Score$_l$ + 1
        If $f^{mcs}_{H_s}$ incorrect and f_l correct
            Score$_l$ = Score$_l$ + 2
        If $f^{mcs}_{H_s}$ incorrect and f_l incorrect
            Score$_l$ = Score$_l$ - 2
    End
End
$H_S = H_S \cup f$, where f has the largest Score

Figure 5.1 Algorithm of Concurrency Thinning

The concept of *PCDM* diversity measure, which is introduced in Section 4.1.1, is applied in the Accuracy In Diversity (AID) method [Banfield et al 2005]. The uncertainty samples are identified in the training samples. The uncertainty samples are defined as those training samples for which the proportions of base classifiers of the MCS can correctly classify is between 10% to 90%. The Upper Bound (*UB*) and Lower Bound (*LB*) of the training samples are defined by the proportion of the uncertainty samples. The base classifier which has the highest accuracy on the samples between *UB* and *LB* would be selected. Figure (5.2) shows the algorithm of AID.

Find the uncertainty samples in D
$PCDM$ = proportion of uncertainty samples
$LB = m \times PCDM + \frac{1 - PCDM}{c}$ and $UB = 0.9 \times PCDM + m \times (1 - PCDM)$
where m is the mean of accuracy of $f \in H_S$
$H_S = H_S \cup f$, where f has the highest accuracy on the points between LB and UB

Figure 5.2 Algorithm of AID

Boosting-Based Ordering method is proposed in [Martınez-Munoz et al 2007]. This method is similar to the boosting method [Freund 1995, Freund et al 1997]. Each training sample is assigned a weight. All weights are initially set to $1/N$ at first and are adjusted in each iteration according to the performance of the MCS. The base classifier with the lowest weighted error is selected. The algorithm is shown in Figure (5.3).

$H_S = H_S \cup f$, where f has the lowest weighted error (ϵ)
Calculate ϵ of $f_{H_s}^{mcs}$
For each $x_i \in D$
    If $f_{H_s}^{mcs}$ incorrect, $w_i = w_i / 2\epsilon$
    Else $w_i = w_i / 2(1 - \epsilon)$
End

Figure 5.3 Algorithm of Boosting-Based Ordering Method

The Kappa (κ) measure is used for the base classifier selection in [Giacinto et al 2001a, Margineantu et al 1997]. κ measures the level of agreement of base classifiers while correcting for chance. It is defined as:

$$\kappa_{ij} = \frac{2(N^{11}N^{01} - N^{10}N^{00})}{(N^{11} + N^{01})(N^{10} + N^{00}) + (N^{11} + N^{10})(N^{01} + N^{00})}, \quad (5.1)$$

where N^{00}, N^{01}, N^{10} and N^{11} are defined in Table (4.3). Lower value of κ means higher disagreement and hence lower κ is preferred. The base classifier which has the lowest value of κ with the MCS is selected.

Giacinto and Roli [Giacinto et al 2001b] cluster the ensembles based on the Double-Fault diversity measure (DF). DF is defined as

$$DF_{ij} = \frac{N^{00}}{N^{11} + N^{10} + N^{01} + N^{00}}. \tag{5.2}$$

The rationale behind this method is that classifiers getting wrong at the same time are not preferable since they cannot help each other to reduce the error. *DF* measures the probability of two classifiers being wrong at the same time. The value of *DF* is from 0 to 1 and smaller value of *DF* is preferred. The base classifiers are clustered using *DF* as distance matrix and the average linkage clustering is applied. In each iteration, the base classifier with the highest training accuracy is selected from each cluster and be integrated to the MCS.

5.2 L-GEM^{MCS} Base Classifier Subset Selection (LCS)

The algorithm of the base classifier subset selection method using L-GEM^{MCS} is introduced in this section. The method is called L-GEM^{MCS} Base Classifier Subset Selection (LCS). The idea of the LCS is to use the L-GEM^{MCS} as a criterion to select the base classifier subset. In each iteration, the LCS calculates the local generalization error bound of the MCS when an additional base classifier is added to it. The added base classifier which contributes to the MCS with the smallest error bound is chosen.

Before presenting the algorithm of the LCS in detail, the selection criterion needs to be defined. Since A in R^*_Q is a constant when the training set is given, it does not affect the selection result. Therefore, A is ignored in the LCS and the selection criterion ($R'_Q(f)$) is defined as:

$$R'_Q(f) = \left(\sqrt{\underset{D}{E}\left((f(x_i) - F(x_i))^2\right)} + \sqrt{\underset{D}{E}\left(\underset{Q_i}{E}\left((f(x) - f(x_i))^2\right)\right)} \right)^2. \tag{5.3}$$

R'_Q of the MCS combined by H_S is:

$$R'_Q\left(f^{mcs}_{H_s}\right)=\left(\begin{array}{l}\sqrt{\sum_{f_l\in H_s}\sum_{f_m\in H_s} w_l w_m \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)\left(f_m(x_i)-F(x_i)\right)\right)} \\ +\sqrt{\sum_{f_l\in H_s}\sum_{f_m\in H_s} w_l w_m \underset{D}{E}\left(\underset{Q_i}{E}\left(\left(f_l(x)-f_l(x_i)\right)\left(f_m(x)-f_m(x_i)\right)\right)\right)}\end{array}\right)^2. \tag{5.4}$$

Let Err^{mcs} and Sen^{mcs} be:

$$Err^{mcs}=\sum_{f_l\in H_s}\sum_{f_m\in H_s} w_l w_m \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)\left(f_m(x_i)-F(x_i)\right)\right), \tag{5.5}$$

$$Sen^{mcs}=\sum_{f_l\in H_s}\sum_{f_m\in H_s} w_l w_m \underset{D}{E}\left(\underset{Q_i}{E}\left(\left(f_l(x)-f_l(x_i)\right)\left(f_m(x)-f_m(x_i)\right)\right)\right). \tag{5.6}$$

$R'_Q\left(f^{mcs}_{H_s}\right)$ becomes:

$$R'_Q\left(f^{mcs}_{H_s}\right)=\left(\sqrt{Err^{mcs}}+\sqrt{Sen^{mcs}}\right)^2. \tag{5.7}$$

Let $Err_{H_s}(f)$ and $Sen_{H_s}(f)$ be:

$$Err_{H_s}(f)=\left(\begin{array}{l}\sum_{f_l\in H_s} w_l w \underset{D}{E}\left(\left(f_l(x_i)-F(x_i)\right)\left(f(x_i)-F(x_i)\right)\right) \\ +w^2 \underset{D}{E}\left(\left(f(x_i)-F(x_i)\right)^2\right)\end{array}\right), \tag{5.8}$$

$$Sen_{H_s}(f)=\left(\begin{array}{l}\sum_{f_l\in H_s} w_l w \underset{D}{E}\left(\underset{Q_i}{E}\left(\left(f_l(x)-f_l(x_i)\right)\left(f(x)-f(x_i)\right)\right)\right) \\ +w^2 \underset{D}{E}\left(\underset{Q_i}{E}\left(\left(f(x)-f(x_i)\right)^2\right)\right)\end{array}\right). \tag{5.9}$$

Consider R'_Q of the MCS combined by H_S and an additional base classifier f_k:

$$R'_Q\left(f^{mcs}_{H_s\cup\{f_k\}}\right)$$

$$= \left(\begin{array}{l} \sqrt{Err^{mcs} + w_k^2 \underset{D}{E}\left((f_k(x_i) - F(x_i))^2\right) + \sum_{f_l \in H_s} w_l w_k \underset{D}{E}\left((f_l(x_i) - F(x_i))(f_k(x_i) - F(x_i))\right)} \\ + \sqrt{Sen^{mcs} + w_k^2 \underset{D}{E}\left(\underset{Q_i}{E}\left((f_k(x) - f_k(x_i))^2\right)\right) + \sum_{f_l \in H_s} w_l w_k \underset{D}{E}\left(\underset{Q_i}{E}\left((f_l(x) - f_l(x_i))(f_k(x) - f_k(x_i))\right)\right)} \end{array} \right)^2$$

$$= \sqrt{Err^{mcs} + Err_{H_s}(f)} + \sqrt{Sen^{mcs} + Sen_{H_s}(f)}, \quad (5.10)$$

Equation (5.7) shows that the value of $R_Q'\left(f_{H_s}^{mcs}\right)$ depends on two components: Err^{mcs} and Sen^{mcs}. When an additional base classifier (f_k) is added to H_S, $R_Q'\left(f_{H_s \cup f_k}^{mcs}\right)$ can be calculated by Err^{mcs}, Sen^{mcs}, $Err_{H_s}(f)$ and $Sen_{H_s}(f)$, which is shown in Equation (5.10). Comparing with Equations (5.7) and (5.10), only two components ($Err_{H_s}(f)$ and $Sen_{H_s}(f)$) are needed to compute $R_Q'\left(f_{H_s \cup f_k}^{mcs}\right)$. Therefore, there is no need to compute $R_Q'\left(f_{H_s \cup f_k}^{mcs}\right)$ again for each new subset of base classifiers. This reduces the computational complexity of the LCS. $Sen_{H_s}(f)$ can be computed by using Equations (4.33) and (4.34).

1. Initialization:

1.1 $Err^{mcs} = 0$, $Sen^{mcs} = 0$, and $H_s = \phi$

2. Add the best base classifier to H_S

2.1 $f_s = \arg\min_f \left(R_Q'\left(f_{H_s \cup f_k}^{mcs}\right)\right)$, where $f \in C_H H_S$

(2.2) if $R_Q'\left(f_{H_s}^{mcs}\right) - R_Q'\left(f_{H_s \cup f_k}^{mcs}\right) > \theta$ and $H_s \neq \phi$, Terminate the Loop

2.3 $Err^{mcs} = Err^{mcs} + Err_{H_s}(f_s)$ and $Sen^{mcs} = Sen^{mcs} + Sen_{H_s}(f_s)$

2.4 $H_S = H_S \cup \{f_s\}$

2.5 Goto 2.1 until $C_H H_s = \phi$ or $|H_S| = L'$

Figure 5.4 Algorithm of the base classifier subset selection using L-GEM^{MCS}

The algorithm of the LCS is shown in Figure (5.4). The parameters are initialized in step 1. Next, the base classifiers are selected one by one iteratively. In step 2.1, R'_Q is calculated for each classifier in $C_H H_S$. The classifier f_k which yields the lowest value of $R'_Q\left(f^{mcs}_{H_s \cup f_k}\right)$ is added to the MCS. It should be noted that H_S is empty in the first iteration. Therefore, the base classifier which has the lowest single localized generalization error bound is selected as the first classifier in the LCS. Step 2.2 is an optional stopping condition. A certain level of performance improvement should be expected after a new base classifier is added to the MCS. Otherwise, the base classifier should not be selected. θ denotes the expected localized generalization error bound improvement. In the special case when $\theta = 0$, the base classifier will be selected only if integrationist addition can make R'_Q of the MCS decrease, which is $R'_Q\left(f^{mcs}_{H_s}\right) > R'_Q\left(f^{mcs}_{H_s \cup f_k}\right)$. Finally, the selected classifier is added to the H_S. The process ends when all base classifiers are selected or the number of selected classifier reaches a predetermined number L'.

The time complexity of the LCS is higher than existing selection methods since $Err_{H_s}(f)$ and $Sen_{H_s}(f)$ require additional computational effort. The time complexity of one iteration for the LCS is $O(nN\ (|H_S|+1)\ |C_H H_S|)$, where n is the number of features of a sample, N is number of samples and $|H_S|$ is the number of classifiers in H_S. The time complexity is highest when $|H_S|$ is equal to $|C_H H_S|$. The time complexity of select all base classifiers from H using the LCS is $O(nN\sum_{i=1}^{|H|-1} i(|H|+1-i))$, where $|H|$ is the number of classifiers in H.

5.3 Experiments

Table (5.1) shows thirteen datasets selected from the UCI machine learning repository [MLR] and Intelligent Data Analysis Group [DAG]. They cover a wide range of

applications involving two-class and multi-class problems. Each dataset is equally divided into two parts randomly: training and testing. The experiment generates thirty independent runs for each pair of datasets. Only samples in the training set are used during training. The samples in the testing set are reserved to evaluate the performances of the trained classifiers. The inputs of all samples are normalized to [0, 1].

Table 5.1 Twelve Datasets

Dataset	Short Name	# Class	# Sample	# Feature
Breast Cancer Wisconsin	Canc	2	569	32
Car Evaluation	Car	4	1728	6
Credit Approval	Cred	2	690	15
Dermatology	Derm	6	366	34
Solar Flare	Solar	2	1066	9
German Credit Data	Germ	2	1000	24
Hepatitis	Hepa	2	80	19
Spambase	Spam	2	4601	57
Thyroid	Thy	2	215	5
Tic-Tac-Toe Endgame	TTT	2	958	9
Titanic	Tit	2	2201	3
Waveform	Wave	3	5000	21
Wine	Wine	3	178	13

In this section, the performance of the LCS is discussed and compared with Reduce Error (RE), Kappa (Kappa), Complementarily Measure (CM), Concurrency Thinning (CC), Accuracy in Diversity (AID), Boosting-Based (Boost) and Clustering (Cluster) method experimentally. A random base classifier selection method (Ran) is also included in the comparison to show the performance of random guessing. The size of q in the LCS is determined by cross validation.

A pool of trained base classifiers (H) is formed by 50 trained classifiers. Half of them are MLP Neural Network and the rests are RBF Network. The base classifiers are trained by Bagging [Breiman 1996]. It means each base classifier is assigned a different training set which is randomly selected from the original training set with replacement. For the MLP Neural Network, it has only one hidden layer. The number of hidden neurons is randomly selected from two to fifty. Gradient descent [Kiernan et al 1996] is applied to train

the weights. For the RBF network, its number of neurons is also selected randomly from two to fifty. The center and width of the neurons are determined by K-means [Kiernan et al 1996] and the K-nearest-neighbor algorithm [Musavi et al 1992] respectively. The weight is calculated using the Singular Value Decomposition (SVD) method [Mak et al 1998]. The base classifiers are combined by the average fusion method.

All experiments are given the same set of base classifiers. Each base classifier subset selection method chooses the most suitable base classifier to be added to the MCS based on certain selection criteria.

The average percentage of classification accuracy and the variance of the testing sets of the MCSs combined by the best base classifiers subsets over 30 independent runs are shown in Table (5.2). A column represents a fusion method while a dataset is represented by a row. The Student's t-test is applied to examine the statistical significance of the improvement made by the LCS. When the absolute t-value is larger than 2.00 in each experiment, a difference between two means is significant at the 95% probability level. The value is bolded and underlined in the cell if the performance of the MCSs with the LCS is significantly better than the one using other methods.

Table (5.2) shows the best MCS out of the 50 totally created by using the LCS is more accurate, which is about 0.89% more in average, than the best ones created by other methods. For example, for the Hepatitis and Waveform datasets, LCS is 89.63% and 87.36% while the others are only 87.58% and 86.01% on average respectively. Comparing the RE with the LCS, RE relies entirely on the training error while the LCS measures both the training error and the sensitivity. The experimental results show the best subset of base classifiers selected by the RE has a lower accuracy than the LCS. It shows that the sensitivity term may be useful for generalization capability estimation. It should also be

noted that the Ran may perform better than some selection methods. For example, for the Wine dataset, Ran performs second best to the LCS.

Table 5.2 LCS VS Other Methods
Average Classification Accuracy and Variance of Testing Set of the Best Base Classifier Subsets over Thirty Independent Runs

	LCS	Ran	CM	Kappa	RE	Boost	AID	CC	Cluster
Canc	97.18 ±0.01	**96.61 ±0.01**	96.83 ±0.01	97.10 ±0.02	97.05 ±0.01	96.92 ±0.01	97.01 ±0.01	96.79 ±0.01	96.96 ±0.02
Car	93.85 ±0.01	**91.63 ±0.01**	**92.47 ±0.01**	**92.47 ±0.01**	**93.05 ±0.01**	**92.47 ±0.01**	**92.47 ±0.01**	**92.47 ±0.01**	**92.42 ±0.01**
Cred	86.90 ±0.01	86.26 ±0.01	86.41 ±0.01	86.41 ±0.01	86.19 ±0.01	86.70 ±0.01	86.37 ±0.01	86.56 ±0.01	86.41 ±0.01
Derm	97.75 ±0.01	97.32 ±0.01	97.25 ±0.01	97.25 ±0.01	**96.98 ±0.01**	97.25 ±0.01	97.25 ±0.01	97.25 ±0.01	**97.12 ±0.01**
Solar	68.21 ±0.02	**67.34 ±0.02**	**67.55 ±0.01**	**67.01 ±0.01**	**66.89 ±0.01**	**67.55 ±0.01**	**66.92 ±0.01**	67.83 ±0.02	**67.03 ±0.01**
Germ	75.92 ±0.04	75.35 ±0.04	75.55 ±0.03	**74.85 ±0.03**	75.30 ±0.05	75.42 ±0.04	75.25 ±0.03	75.75 ±0.02	**74.88 ±0.03**
Hepa	89.63 ±0.10	87.82 ±0.18	88.14 ±0.19	**87.50 ±0.10**	**87.50 ±0.20**	**86.86 ±0.25**	**86.86 ±0.21**	88.46 ±0.15	**87.50 ±0.12**
Spam	89.13 ±0.01	**87.67 ±0.01**	88.72 ±0.01	88.89 ±0.01	88.99 ±0.01	88.83 ±0.01	88.95 ±0.01	88.72 ±0.01	**88.03 ±0.01**
Thy	97.22 ±0.01	**95.91 ±0.03**	**96.61 ±0.01**	96.85 ±0.01	96.73 ±0.02	96.96 ±0.01	96.96 ±0.01	**96.61 ±0.01**	**96.50 ±0.02**
TTT	87.74 ±0.05	**85.99 ±0.02**	**84.97 ±0.04**	86.93 ±0.02	87.42 ±0.02	87.19 ±0.03	87.34 ±0.02	**84.97 ±0.04**	**86.61 ±0.02**
Tit	79.71 ±0.01	**78.66 ±0.01**	**78.66 ±0.01**	**78.62 ±0.01**	**78.66 ±0.01**	**78.66 ±0.01**	**78.66 ±0.02**	**78.66 ±0.01**	**78.62 ±0.01**
Wave	87.36 ±0.01	**86.70 ±0.01**	**86.82 ±0.01**	**86.82 ±0.01**	**86.80 ±0.01**	**86.82 ±0.01**	**86.82 ±0.01**	**86.82 ±0.01**	86.85 ±0.01
Wine	98.98 ±0.02	**98.01 ±0.02**	**97.87 ±0.03**	**97.87 ±0.03**	**97.73 ±0.03**	**97.87 ±0.03**	**97.87 ±0.03**	**97.87 ±0.03**	**97.44 ±0.04**

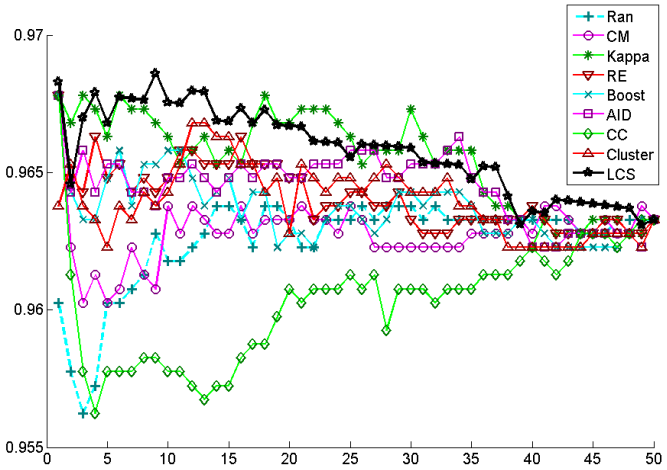
Table 5.3 LCS VS Other Methods
Average Size of the Best Base Classifiers Subset over Thirty Independent Runs

	LCS	Ran	CM	Kappa	RE	Boost	AID	CC	Cluster
Canc	3.63	4.00	3.00	3.63	3.63	2.38	2.13	6.38	4.50
Car	5.75	11.63	4.63	4.63	4.13	4.63	4.63	4.63	7.63
Cred	7.75	5.25	10.50	13.63	5.38	10.13	17.13	15.88	11.25
Derm	7.63	3.75	5.38	5.38	6.13	5.38	5.38	5.38	6.25
Solar	23.00	10.25	9.13	30.63	29.00	14.75	26.38	15.38	22.25
Germ	12.75	17.00	19.88	24.75	20.75	16.50	24.63	11.50	24.75
Hepa	5.00	5.50	5.50	5.38	1.88	3.38	3.13	4.13	2.50
Spam	3.75	11.25	1.00	4.13	2.50	1.50	4.88	1.00	24.00
Thy	8.75	3.25	5.38	3.25	5.38	8.63	8.25	7.50	5.63
TTT	21.88	27.75	43.25	23.13	22.88	18.00	22.00	48.25	25.38
Tit	1.80	1.25	1.13	1.23	1.38	1.38	1.25	1.13	12.38
Wave	31.38	19.25	13.88	13.88	14.25	13.88	13.88	13.88	19.00
Wine	7.25	4.25	2.25	2.25	2.50	2.25	2.25	2.25	2.75

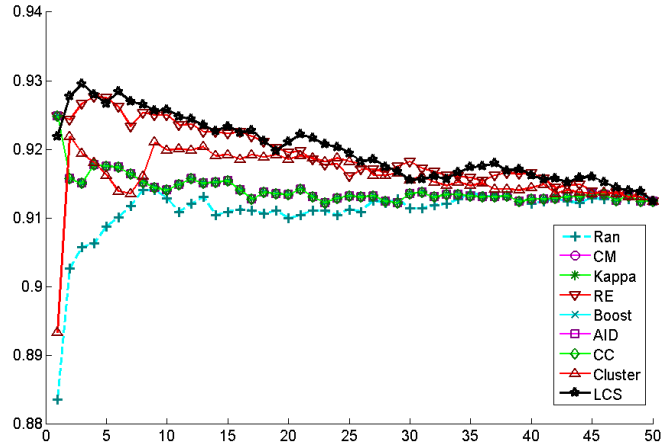
The average size of the best subset of the base classifiers is shown in Table (5.3). On average, for all selection methods, the size of the best subset of the base classifiers is 10.16.

This number is relative small since the MLP Neural Network and the RBF Network are considered to be good classifiers. Combining too larger a set of this kind of good classifiers may not help improve the generalization ability of the MCS. Table (5.3) also indicates that, with the exception of the waveform dataset, the size of the best subset selected by LCS is relatively small comparing with other selection methods.

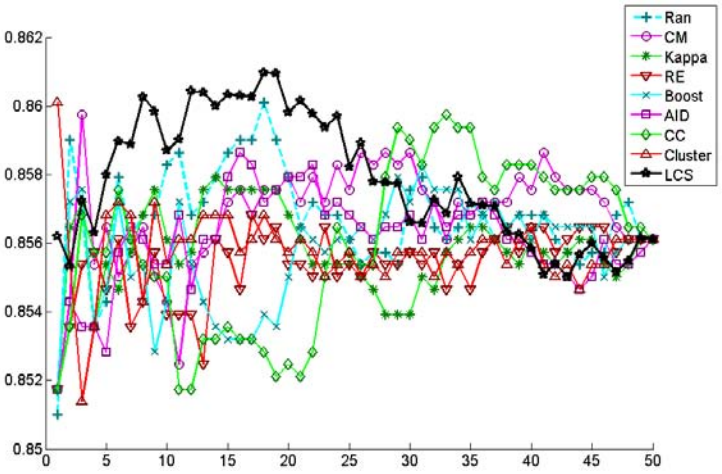
Figure (5.5) shows the testing classification accuracy of the 50 MCSs created with different number of base classifiers over 30 independent runs. The X-axis represents the number of base classifiers included in the MCSs and the Y-axis is the testing accuracy. The performance of the LCS is denoted by a star-line in Figure (5.5). The experimental results show that the MCSs created by using the LCS achieve the highest average accuracy in all datasets comparing with other base classifier subset selection methods. The performance of the MCSs created by the LCS is relatively stable. It is also significant to observe that the MCSs created by the LCS show consistently good performances for all datasets.



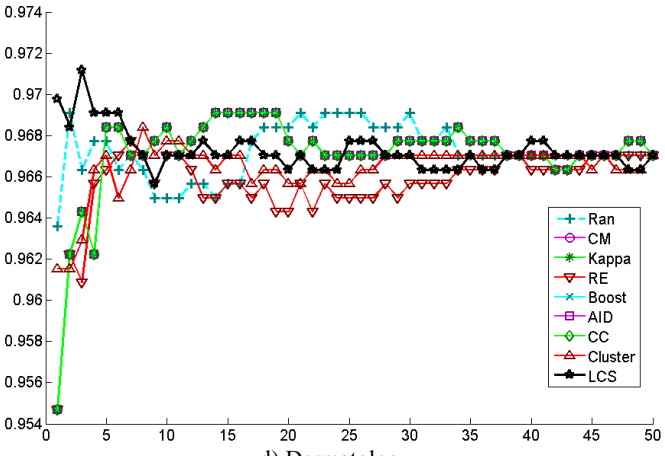
a) Breast Cancer Wisconsin



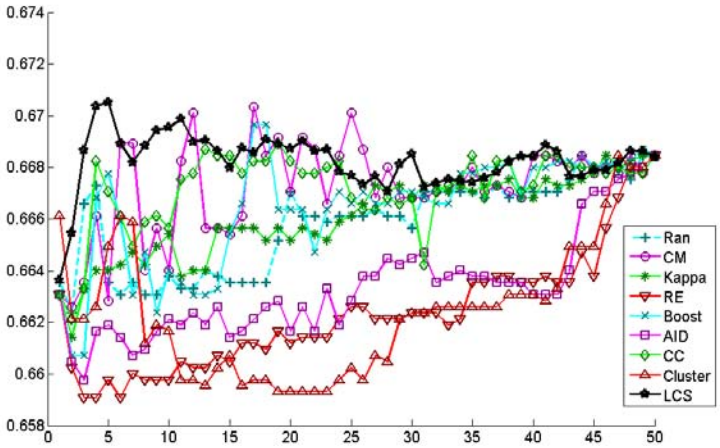
b) Car Evaluation



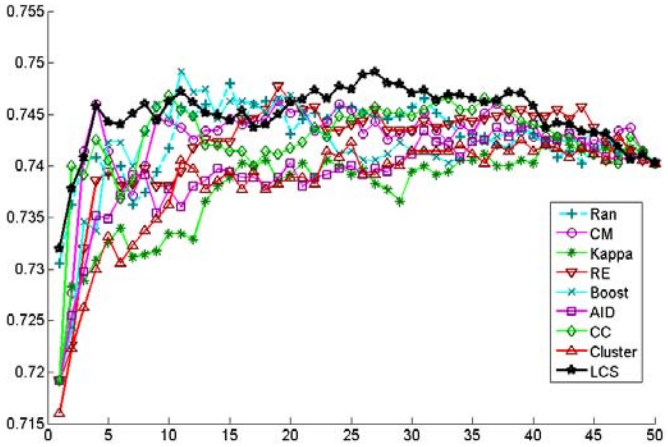
c) Credit Approval



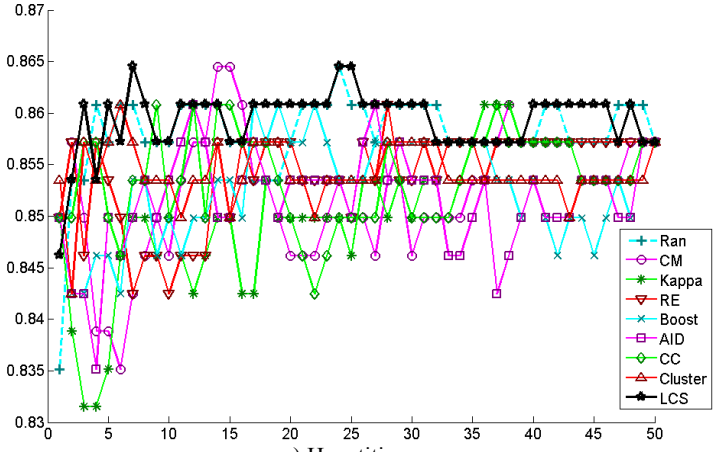
d) Dermatology



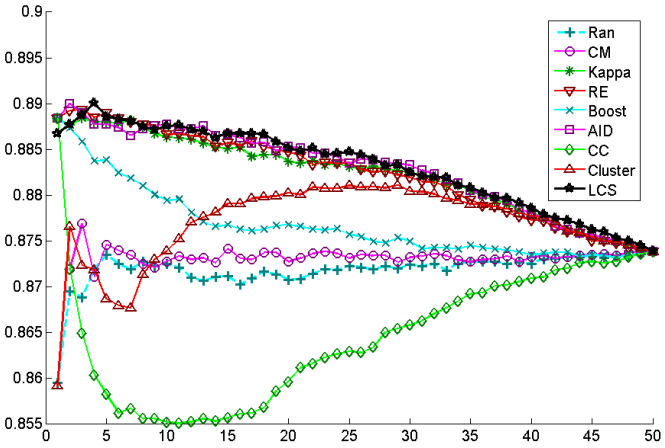
e) Solar Flare



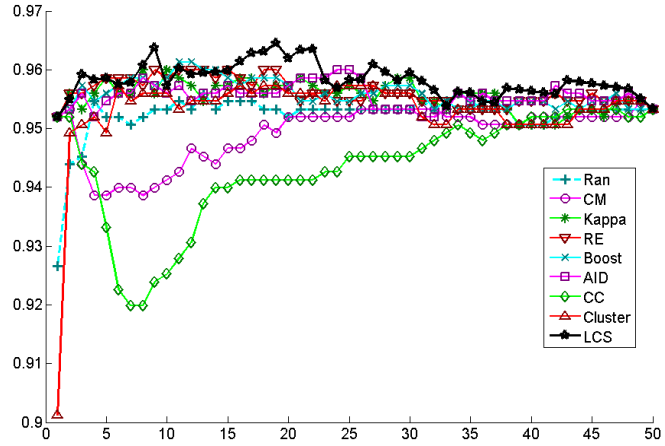
f) German Credit Data



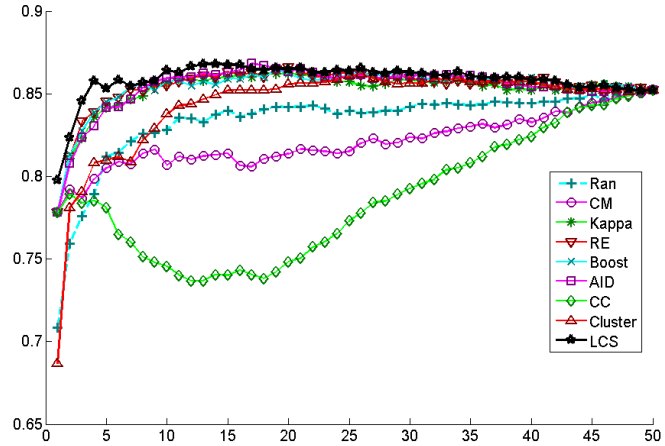
g) Hepatitis



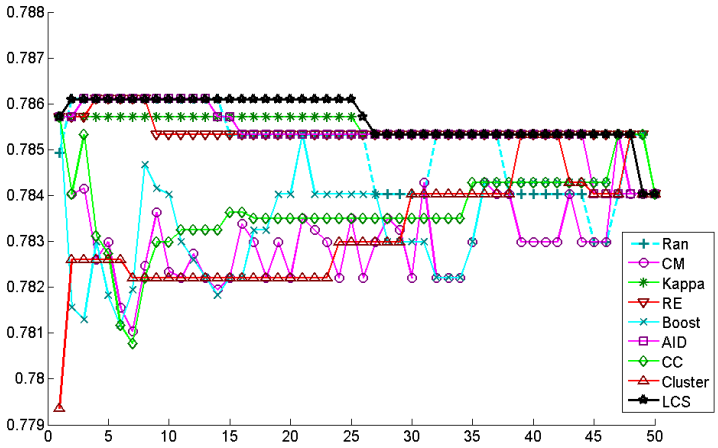
h) Spambase



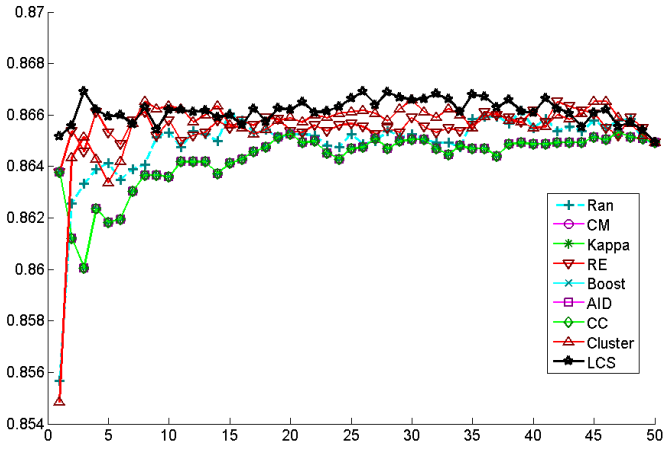
i) Thyroid



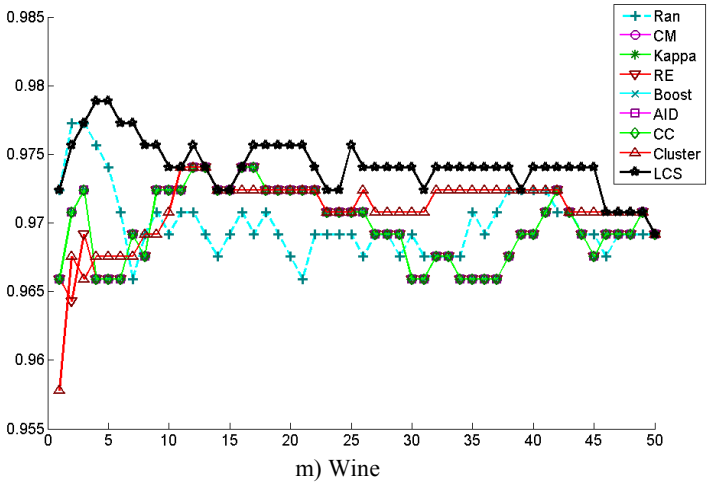
j) Tic-Tac-Toe Endgame



k) Titanic



l) Waveform



m) Wine

Figure 5.5 LCS VS Other Methods
Classification Accuracy of MCSs with Different Number of Base Classifiers
over Thirty Independent Runs

Table 5.4 LCS VS Other Fusion Methods
Win-Tie-Loss Comparison over Twenty Datasets

	Ran	CM	Kappa	RE	Boost	AID	CC	Cluster
Canc	**47-1-2**	**46-1-3**	31-1-18	**46-1-3**	**47-1-2**	**44-1-5**	**48-1-1**	**48-1-1**
Car	**49-1-0**	**48-1-1**	**48-1-1**	**38-1-11**	**48-1-1**	**48-1-1**	**48-1-1**	**47-1-2**
Cred	**37-0-13**	31-0-19	**44-0-6**	**40-0-10**	**32-0-18**	**44-0-6**	28-0-22	**43-0-7**
Derm	17-11-22	14-11-25	14-11-25	**35-13-2**	14-11-25	14-11-25	14-11-25	23-16-11
Solar	**45-0-5**	**33-0-17**	**47-0-3**	**49-0-1**	**37-0-13**	**49-0-1**	**39-0-11**	**47-0-3**
Germ	**42-0-8**	**40-0-10**	**48-0-2**	**40-0-10**	**36-0-14**	**49-0-1**	**43-0-7**	**48-0-2**
Hepa	**44-1-5**	**43-1-6**	**45-1-4**	**46-1-3**	**48-1-1**	**48-1-1**	**44-1-5**	**47-1-2**
Spam	**50-0-0**	**49-0-1**	**46-0-4**	**43-0-7**	**49-0-1**	**32-0-18**	**49-0-1**	**50-0-0**
Thy	**49-0-1**	**49-1-0**	**43-1-6**	**39-1-10**	**40-1-9**	**42-1-7**	**49-1-0**	**49-0-1**
TTT	**50-0-0**	**50-0-0**	**45-0-5**	**42-0-8**	**43-0-7**	**44-0-6**	**50-0-0**	**49-0-1**
Tit	28-22-0	**46-4-0**	25-25-0	20-30-0	**45-5-0**	18-32-0	**45-5-0**	**43-7-0**
Wave	**45-0-5**	**50-0-0**	**50-0-0**	**41-0-9**	**50-0-0**	**50-0-0**	**50-0-0**	**40-0-10**
Wine	**45-4-1**	**48-2-0**	**48-2-0**	**48-2-0**	**48-2-0**	**48-2-0**	**48-2-0**	**40-10-0**
Average	42.2-3.1-4.8	42.1-1.6-6.3	41.1-3.2-5.7	40.5-3.8-5.7	41.3-1.7-7.0	40.8-3.8-5.5	42.7-1.7-5.6	44.2-2.8-3.1

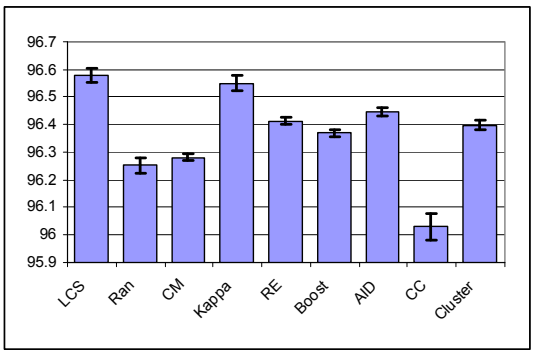
In Table (5.4), the Win-Tie-Loss gives the number of times for which the base classifiers subset selected by the LCS perform better/same/worse in comparison with the ones selected by other selection methods. For example, 47-1-2 is shown in the first cell. It means out of 50 subsets, the MCS created by the LCS performs better in 47, the same in 1 and worse in 2 times comparing with the ones by using Ran on the average of 30 independent runs. Each column represents a selection method. Each row represents a dataset. Let T denote the number of comparison times. If the number of wins is bigger than or equal to $T/2+1.96\sqrt{T}/2$, then the LCS claims to perform significantly better at the 95% probability level. In this experiment, as T = 50, the LCS performs significantly better at the 95% probability level when it wins more than 31 datasets. The value is bolded and underlined in the cell if the performance of the MCS created by the LCS is better significantly. The last row gives the average of the numbers in each column.

The table shows the MCSs created by the LCS outperform other selection methods in most situations and win more than 40 times on the average. Besides the Breast Cancer Wisconsin, Credit Approval, Dermatology and Titanic datasets, LCS is significantly better than all selection methods. In Titanic dataset, from Figure (5.5k), LCS always selects the most accurate subsets. However, the other methods, such as AID, RE, Kappa and Ran, also

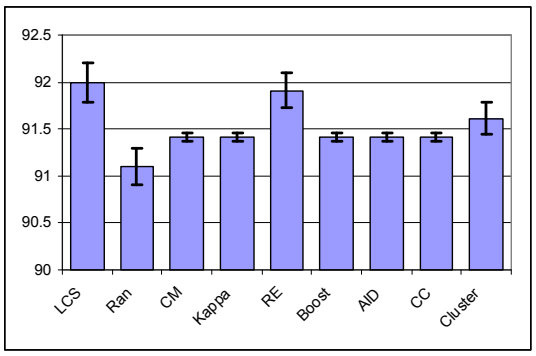
have a good performance. It should be noted that the performance of LCS in the Dermatology dataset is not as good as in other datasets. Figure (5.5d) shows the best subset created by the LCS is formed by three base classifiers. By adding more base classifiers apparently would degrade the performance of the MCS. Moreover, although the Ran, CM, Kappa, Boost and CC win 24.5 times on average over the LCS for the Dermatology dataset, they fail to claim a better performance at the 95% significance level.

Table 5.5 LCS VS Other Methods
Average Classification Accuracy and Variance of Testing Set of the MCSs over Thirty Independent Runs

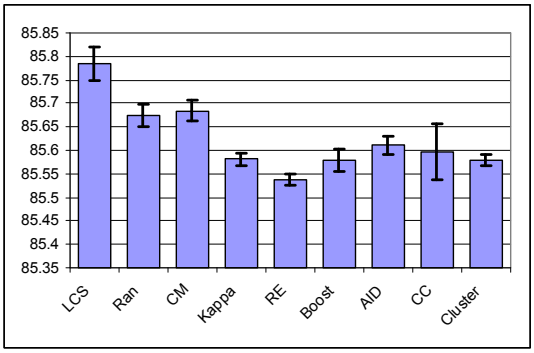
	LCS	Ran	CM	Kappa	RE	Boost	AID	CC	Cluster
Canc	96.58 ±0.01	**96.25 ±0.01**	**96.28 ±0.01**	96.55 ±0.01	**96.41 ±0.04**	**96.37 ±0.01**	**96.45 ±0.01**	**96.03 ±0.01**	**96.40 ±0.01**
Car	92.00 ±0.01	**91.10 ±0.01**	**91.41 ±0.01**	**91.41 ±0.01**	**91.91 ±0.01**	**91.41 ±0.01**	**91.41 ±0.01**	**91.41 ±0.01**	**91.61 ±0.01**
Cred	85.78 ±0.01	**85.67 ±0.02**	**85.68 ±0.01**	**85.58 ±0.02**	**85.54 ±0.01**	**85.58 ±0.01**	**85.61 ±0.01**	**85.60 ±0.01**	**85.58 ±0.02**
Derm	96.73 ±0.01	96.72 ±0.01	96.71 ±0.01	96.71 ±0.01	**96.56 ±0.01**	96.71 ±0.01	96.71 ±0.01	96.71 ±0.01	**96.64 ±0.01**
Solar	66.83 ±0.01	**66.58 ±0.01**	**66.72 ±0.01**	**66.61 ±0.01**	**66.23 ±0.01**	**66.64 ±0.01**	**66.34 ±0.01**	**66.72 ±0.01**	**66.25 ±0.02**
Germ	74.49 ±0.02	**74.27 ±0.01**	**74.25 ±0.01**	**73.78 ±0.01**	**74.20 ±0.01**	**74.19 ±0.02**	**73.92 ±0.01**	**74.26 ±0.01**	**73.83 ±0.01**
Hepa	85.93 ±0.01	**85.81 ±0.01**	**85.21 ±0.01**	**85.07 ±0.01**	**85.39 ±0.01**	**85.27 ±0.01**	**85.16 ±0.01**	**85.37 ±0.01**	**85.41 ±0.01**
Spam	88.31 ±0.01	**87.19 ±0.02**	**87.36 ±0.01**	**88.23 ±0.02**	**88.23 ±0.01**	**87.67 ±0.01**	88.28 ±0.01	**86.48 ±0.01**	**87.68 ±0.01**
Thy	95.83 ±0.01	**95.23 ±0.01**	**94.93 ±0.01**	**95.55 ±0.01**	**95.61 ±0.01**	**95.59 ±0.01**	**95.55 ±0.01**	**94.36 ±0.01**	**95.30 ±0.01**
TTT	85.83 ±0.01	**83.33 ±0.06**	**82.09 ±0.03**	**85.23 ±0.02**	**85.41 ±0.02**	**85.38 ±0.02**	**85.39 ±0.02**	**78.73 ±0.14**	**84.34 ±0.09**
Tit	78.57 ±0.01	**78.51 ±0.01**	**78.31 ±0.01**	78.55 ±0.01	78.54 ±0.01	**78.35 ±0.01**	78.54 ±0.01	**78.37 ±0.01**	**78.32 ±0.01**
Wave	86.62 ±0.01	**86.49 ±0.01**	**86.43 ±0.01**	**86.43 ±0.01**	**86.56 ±0.01**	**86.43 ±0.02**	**86.43 ±0.01**	**86.43 ±0.02**	**86.55 ±0.01**
Wine	97.42 ±0.01	**96.99 ±0.01**	**96.98 ±0.01**	**96.98 ±0.01**	**96.96 ±0.01**	**96.98 ±0.01**	**96.98 ±0.01**	**96.98 ±0.01**	**97.09 ±0.01**



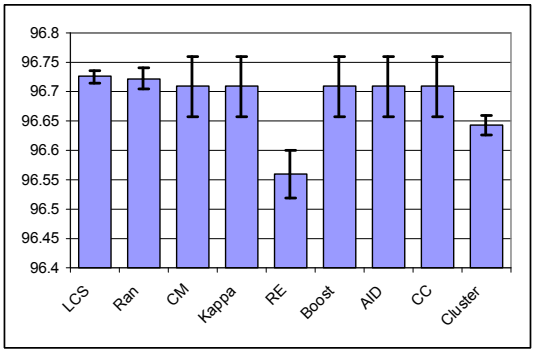
a) Breast Cancer Wisconsin



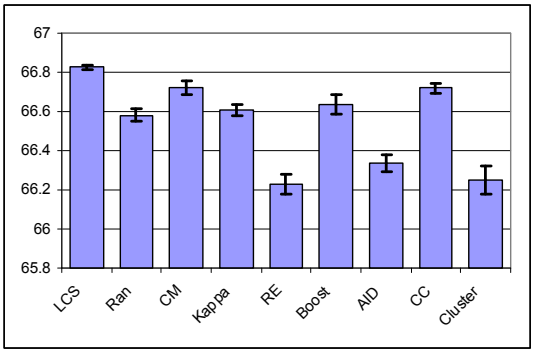
b) Car Evaluation



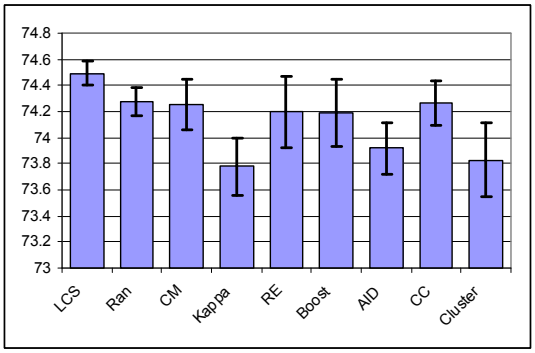
c) Credit Approval



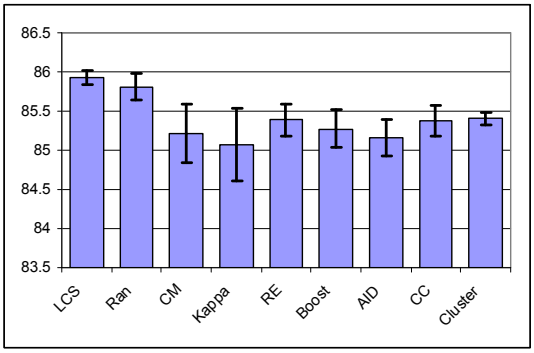
d) Dermatology



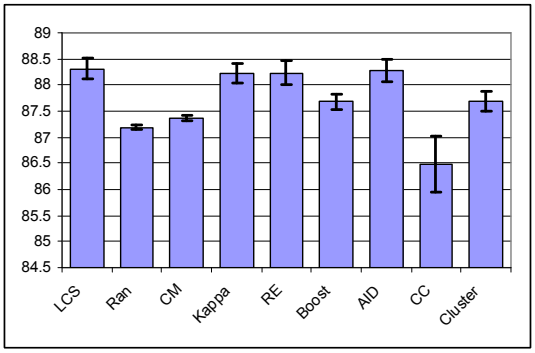
e) Solar Flare



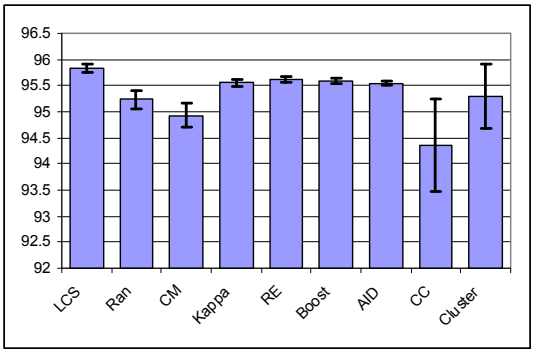
f) German Credit Data



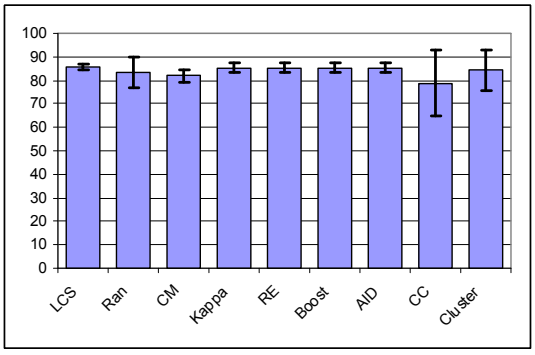
g) Hepatitis



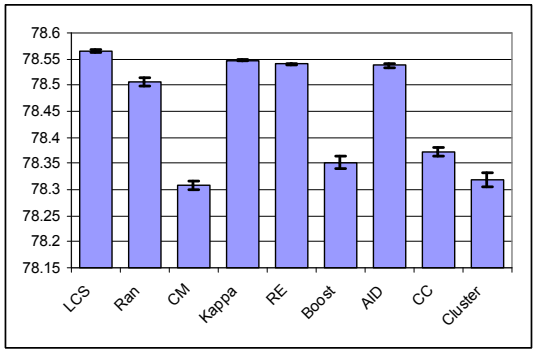
h) Spambase



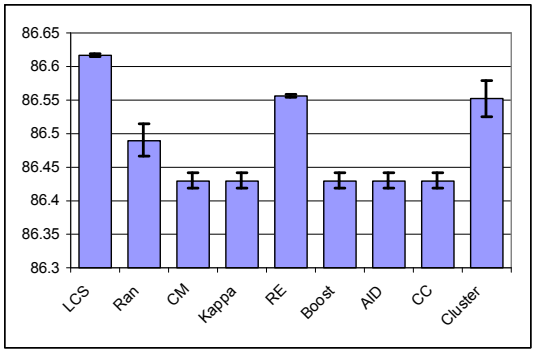
i) Thyroid



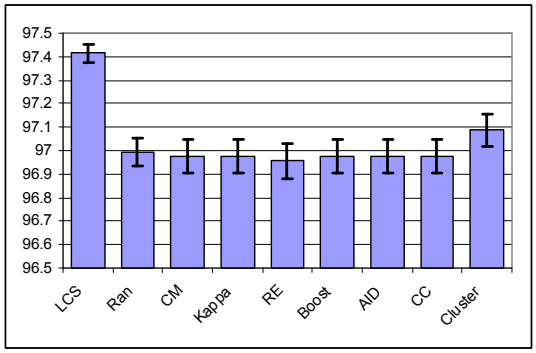
j) Tic-Tac-Toe Endgame



k) Titanic



l) Waveform



m) Wine

Figure 5.6 LCS VS Other Methods
Eror Bar with variances of Testing Set of the MCSs over Thirty Independent Runs

The average percentage of classification accuracy and the variance of the testing sets of the MCSs using base classifier subset selection methods over 30 independent runs are shown in Table (5.5) and Figure (5.6). The value is bolded and underlined in the cell if the performance of MCSs with LCS is significantly better than the one using other methods. This experimental results show the average performance of MCSs combined with the base classifiers selected by different selection methods. LCS has the highest testing accuracy on average comparing with others in all datasets. In most cases, the improvements are significant at 95% level. It indicates that in general, the subset of base classifiers selected by LCS is better than other methods in terms of accuracy. Similar to Table (5.2), the performance of the Ran may not be the worst. For example, for the Credit Approval, the Ran is 85.67% and it is better than all methods except the CM and the LCS.

The selection time of different selection methods is shown in Table (5.5). As no calculation is required by the Ran method, it has the shortest selection time. Except the LCS, the time required by all other methods are similar and the average is 5.7. LCS requires much longer time. This is due to the additional calculations of the sensitivity terms (Sen^{base} and Sen^{div}) in R'_Q. The average training time of 50 base classifiers in all datasets over 30 independent runs is 43.5, which is independent of the selection methods used. The selection time of LCS is about two times of this training time. Morever, in average, LCS required 15.77 minutes to select the best base classifier subset in the experiments while 1 – 2 minutes are used by other methods. Although LCS is much slower than other methods, the selection time used by LCS is still reasonable. Futhermore, since the classifier selection is done prior to any classification application task, a longer time for the creation of the MCS may not be a critical consideration.

Table 5.6 LCS VS Other Methods
Average Selection Time of Testing Set over Thirty Independent Runs

Method	LCS	Ran	RE	Kappa	CM	CC	AID	Boost	Cluster
Time	87.67	0.3	5.6	5.9	5.5	6.2	5.5	5.5	5.7

5.4 Conclusion

In this chapter, a base classifier subset selection method for MCSs, named L-GEM base classifier selection (LCS), is proposed. Current selection methods only use the information provided by the training set as the selection criteria, without any theoretical justification. Howeevr, the LCS makes use of the generalization error bound as the selection criterion.

The experimental results show that for the same set of trained base classifiers, the highest as well as the average testing accuracies of the MCSs created by the LCS are higher than those using other selection methods. This indicates that LCS can select the best base classifiers from a pool of given classifiers to form a MCS. One possible reason is that R'_Q

can indeed provide a useful estimation of the generalization ability of the MCS. Although the LCS takes a longer time for the selection of the base classifiers, this could be done prior to any real classification task.

CHAPTER 6
CONCLUSION AND FUTURE WORK

This research attempts to find a generalization error bound model for a MCS. The L-GEM for a single classifier system has been extended to a MCS, called L-GEM^{MCS}, for supervised classification problems using the Mean Square Error function as its cost function. The L-GEM^{MCS} has advantages over the existing models for MCSs in at least two aspects.

Conceptually speaking, unlike the other models which mainly rely on the training error, it takes into consideration both the training error and the output differences between unseen samples and the training samples in a region that contains all training samples. In other words it also considers the "local smoothness" of the discriminant function (versus the "global smoothness considered by the Regularization technique). The L-GEM^{MCS} is composed of four terms. Err^{base} (Sen^{base}) measures the error (sensitivity) for each base classifier, while Err^{div} (Sen^{div}) measures the degree of difference between errors (sensitivities) of each pair of base classifiers.

Practically speaking, it provides a easy-to-compute quantitative measure on the generalization error bound of the MCS with low computational complexity. Both the L-GEM and the L-GEM^{MCS} are found to have a wide range of applications. For instance, the L-GEM can be used as a training objective function for neural network training by minimizing the R^*_Q, with significantly better results than using wither the training error or the regularization function as an objective function. Another application is weight assignment in dynamic fusion methods. L-GEM proves to be a better way to assign weights to classifiers than other existing techniques. The problem of choosing the best one from a

given set of trained MCSs according to certain criteria is a practical yet fundamental problem for MCSs. Again L-GEM^{MCS} could be used as the selection criteria. Experimental results show that the MCSs selected by L-GEM^{MCS} have higher accuracy than those selected by other methods. The final application is the determination of the number of base classifiers used to form a MCS, given that a pool of trained base classifiers are available. Experimental results demonstrate the superiority of using L-GEM^{MCS} as an evaluation criteria for base classifiers subset selecting over other methods.

Until now the L-GEM and the L-GEM^{MCS} have been successfully applied to a wide range of pattern classification problems. However, many interesting research problems are wide open and a few ones will be mentioned here:

- Relationship between Localized Generalization Error Bound (in MSE) and Classification Error. $\boldsymbol{R_Q^*}$ **in L-GEMMCS is the upper bound on MSE of MCSs in** $\boldsymbol{Q}$ **neighborhood. The relationship between MSE bound and classification error is not clear.**

- **Determination of parameter** $\boldsymbol{q}$**. The parameter** $\boldsymbol{q}$ **is a critical value in L-GEM^{MCS} as it directly affects the computation of the generalization error bound. Parameter q has a geometric meaning in the input space and determines the size of Q neighborhood. The distribution of the training samples may be helpful to determine a proper q.**

- **L-GEMMCS Extension to other Fusion Methods. The localized concept in L-GEM is very general and could be applied to other fusion methods, for example, the fusion methods of label output.**

- **Objective Function of Genetic Algorithm for Sub-Optimal Base Classifiers Selection. The result found by genetic algorithm should be better than the greedy search in terms of the testing accuracy. The major problem is time complexity.**

- **Training MCSs by using R*Q in L-GEMMCS. A pilot study is given in Section 4.7 which shows that the MCSs trained by L-GEMMCS have good performance. This problem can be studied more extensively by analyzing the performance on different datasets and reducing the computational complexity.**

- **Data Analysis. The most informative samples in a training set for a classifier can be found by L-GEM. After a classifier is trained by a training set, L-GEM can be applied to estimate the sensitivity of Qi neighborhood of each training samples. If the value of sensitivity of a sample is high relatively, it means its classifier's outputs are not stable. This sample may be near to the decision boundary and its sensitivity value could be useful information for this classifier. Such samples are called Sensitivity Vectors (SVs). SVs in a training set can be selected by using L-GEM and can be analyzed in detail. For instance, the outliers in a training set can be detected by this method. If the outlier is one of the SVs, this means it is located near to the decision boundary and it may affect the shape of the boundary. So, we have to analyze it and see if it should be removed. If an outlier is not one of the SVs, it is not important since it does not affect the shape of the decision boundary so it can be ignored.**

- **Application to Unsupervised Learning Problems. An unsupervised learning problem could be regarded as a supervised learning problem after the learning samples are labeled according to the result of a clustering method.**

- **Time Complexity. The calculation of additional sensitivity term is time consuming especially when the size of base classifier is large. This is because one needs to evaluate all possible pairs of base classifiers in the MCS.**

REFERENCES

Abraham, A. and Grosan, C. (2008). Pharmaceutical Drug Design Using Dynamic Connectionist Ensemble Networks. *Iwata S. et al (Eds.), Communications and Discoveries from Multidisciplinary Data, Studies in Computational Intelligence, Springer Verlag, pp. 221-231*.

Aksela, M. (2003). Comparison of classifier selection methods for improving committee performance. In *Proc. 4th Int'l Workshop on Multiple Classifier Systems, LNCS 2709, pp. 84-93*.

Banfield, R.E., Hall, L.O., Bowyer, K.W. and Kegelmeyer, W.P. (2003). A New Ensemble Diversity Measure Applied to Thinning Ensembles. In *Proc. 4th Int'l Workshop on Multiple Classifier Systems, LNCS 2709, pp. 306-316*.

Banfield, R.E., Hall, L.O., Bowyer, K.W. and Kegelmeyer, W.P. (2005). Ensemble Diversity Measures and Their Application to Thinning. *Information Fusion, vol. 6(1), pp. 49-62*.

Battiti, R. and Colla, A.M. (1994). Democracy in neural nets: Voting schemes for classification. *Neural Networks, vol. 7, pp. 691-707*.

Bertolini, D., Oliveira, L.S., Justino, E. and Sabourin, R. (2009). Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers. To appear in *Pattern Recognition*.

Bhadoria, R., Das, S., Misra, M. and Sarje, A.K. (2005). Performance Comparison of Majority Voting with ROWA Replication Method over PlanetLab. In *Proc. 7th Int'l Workshop Distributed Computing, LNCS 3741, pp. 147-152.*

Bishop, C. (1995). Neural Networks for Pattern Recognition. *Clarendon Press.*

Breiman, L. (1994). Bagging Predictors. *Technical Report 421, Department of Statistics, University of California, Berkeley*

Breiman, L. (1996). Bagging Predictor. *Machine Learning, vol. 26, pp. 123-140.*

Brizzotti, M.M. and Carvalho, A.C.P.L.F. (1999). The Influence of Clustering Techniques in the RBF Networks Generalization. In *Proc. of the 7th Int'l Conference on Image Processing And Its Applications, vol. 1, pp. 87-92.*

Broomhead, D. and Lowe, D. (1988). Multivariable Functional Interpolation and Adaptive Networks. *Complex Systems, vol. 2, pp. 321-355.*

Brown, G. (2004). Diversity in Neural Network Ensembles. PhD thesis, University of Birmingham.

Bruzzone, L. and Prieto, D.F. (1999). A Technique for the Selection of Kernel-Function Parameters in RBF Neural Networks for Classification of Remote-sensing Images. *IEEE Trans. on Volume Geoscience and Remote Sensing, vol. 37(2), pp. 1179-1184.*

Caruana, R., Lawrence, S. and Giles, C.L. (2000). Overfitting in Neural Networks: Backpropagation, Conjugate Gradient, and Early Stopping, *Advance in Neural Information Processing Systems, vol. 13, pp. 402-408.*

Caruana, R., Niculescu-Mizil A., Crew, G. and Ksikes, A. (2004). Ensemble Selection from Libraries of Models. In *Proc. 21st Intl Conf. Machine Learning, pp. 18-26.*

Chan, P.P.K., Ng, W.W.Y. and Yeung, D.S. (2005). Active Learning Using Localized Generalization Error of Candidate Sample as Criterion. In *Proc. Int'l Conf. on Systems, Man and Cybernetics, vol. 4, pp. 3604-3609.*

Chen, L. and Kamel, M.S. (2009). A Generalized Adaptive Ensemble Generation and Aggregation Approach for Multiple Classifier Systems. *Pattern Recognition, vol 42 (5), pp. 629-644.*

Chen, Z. and Haykin, S. (2000). On Different Facets of Regularization Theory. *Neural Computation, vol. 14(12), pp. 2791-2846.*

Cherkassky, V. and Mulier, F. (1998). Learning From Data. *Wiley.*

Cherkassky, V., Shao, V., Mulier, F. M. and Vapnik, V. N. (1999). Model Complexity Control for Regression Using VC Generalization Bounds. *IEEE Trans. Neural Network, vol. 10(5), pp. 1075-1089.*

DAG: http://ida.first.fraunhofer.de/homepages/ida/

De Castro, M.C.F., De Castro, F.C.C. and Arantes, D.S. (1999). RBF neural networks with centers assignment via Karhunen-Loeve transform. In *Proc. Int'l Joint Conf. on Neural Networks, vol. 2, pp. 1265-1270.*

Derbeko, P., Ran, E.Y. and Meir, R. (2002). Variance Optimized Bagging. In *Proc. 13th European Conf. on Machine Learning, LNCS 2430, pp. 60-72.*

Dietterich, T.G. (1997). Machine Learning Research: Four Current Directions. *The AI Magazine, vol. 18(4), pp. 97-136.*

Dietterich, T.G. (2000a). Ensemble methods in machine learning. In *Proc. 1st Int'l Workshop on Multiple Classifier Systems, LNCS 1857, pp. 1-15.*

Dietterich, T.G. (2000b). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning, vol. 40(2), pp. 139-157.*

Duda, R.O., Hart, P.E. and Stork, D.G. (2000). Pattern Recognition, second edition. *Willey.*

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation, vol. 121(2), pp. 256-285.*

Freund, Y. and Schapire, R.E. (1996). Experiments with a new boosting algorithm. In *Proc. 13th Int'l Conf. on Machine Learning, pp. 148-156.*

Freund, Y. and Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, vol. 55(1), pp. 119-139.*

Fumera, G., Roli, F. and Serrau, A. (2008). A Theoretical Analysis of Bagging as a Linear Combination of Classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 30(7), pp. 1293-1299.*

Geman, S., Bienenstock, E. and Cooper, L.N. (1992). Neural networks and the Bias/Variance Dilemma. *Neural Computation, vol. 4(1), pp. 1-58.*

Giacinto, G. and Roli, F. (2001a). Design of Effective Neural Network Ensembles for Image Classification Processes. *Image Vision and Computing, vol. 19(9-10), pp. 699-707.*

Giacinto, G. and Roli, F. (2001b). An Approach to the Automatic Design of Multiple Classifier Systems. *Pattern Recognition Letters, vol. 22, pp. 25–33*.

Giacinto, G. and Roli, F. (1999). Methods for Dynamic Classifier Selection. In *Proc. 10th Int'l Conf. on Image Analysis and Processing, pp. 659-664.*

Golea, M., Bartlett, P.L., Lee, W.S. and Mason, L. (1998). Generalization in Decision Trees and DNF : Does Size Matter? *Advances in Neural Information Processing Systems, vol. 10, pp. 259-265*.

Hansen, L. and Salamon, P. (1990). Neural Network Ensembles. *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 12, pp. 993-1001*.

Hastie,T., Tibshirani, R. and Friedman, J. (2001). The Element of Statistical Learning. *Springer-Verlag*.

Haykin, S. (1994). Neural networks a comprehensive foundation. *Macmillan*.

Ho, T.K. (1998). The Random Subspace Method for Constructing Decision Forests. *IEEE Trans. on Pattern Analysis and Machines Intelligence, vol. 20(8), pp. 832-844*.

Ho, T.K., Hull, J.J. and Srihari, S.N. (1994). Decision Combination in Multiple Classifier System, *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 16(1), pp. 66-75.*

Islam, M.M., Xin, Y., Shahriar Non, S.M., Islam, M.A. and Murase, K. (2008). Bagging and Boosting Negatively Correlated Neural Networks. *IEEE Trans. on Systems, Man, and Cybernetics, Part B, vol. 38(3), pp. 771-784* .

Jacobs, R.A., Jordan, M.I., Nowlan, S.J. and Hinton, G.E. (1991). Adaptive Mixture of Local Experts. *Neural Computation, vol. 3, 79-87*.

Ji, C. and Ma, S. (1997). Combinations of weak classifiers. *IEEE Tran. Neural Networks, vol. 8 (1), pp. 32-42*.

Kiernan, L., Mason, J.D. and Warwick, K. (1996). Robust Initialization of Gaussian Radial Basis Function Networks Using Partitioned k-means Clustering. *Electronics Letters, vol. 32(7), pp. 671-673.*

Kim, E. and Ko, J. (2005). Dynamic Classifier Integration Method, In *Proc. 6th Int'l Workshop on Multiple Classifier Systems, LNCS 3541, pp. 97-107*.

Kittler, J., Hatef, M., Duin, R.P.W. and Matas, J. (1998). On Combining Classifiers. *IEEE Tran. on Pattern Analysis and Machine Intelligence, vol. 20(3), pp. 226-239*.

Ko, A. H. R., Sabourin, R. and Britto Jr. A.S. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition, vol. 41(5), pp. 1735-1748*.

Kohonen, T. (1990). The Self-Organizing Map. In *Proc. of the IEEE, vol. 78 (9), pp. 1464-1480*.

Kon, M.A. and Plaskota, L. (2001). Complexity of Regularization RBF Networks. In *Proc. Int'l Joint Conf. on Neural Networks, vol. 1, pp. 342 - 346*.

Koppel, M. and Engelson, S.P. (1996). Integrating Multiple Classifiers by Finding their Areas of Expertise. In *Proc. AAAI Workshop On Integrating Multiple Learning Models, pp. 53-58.*

Krogh, A. and Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems, vol. 7, pp. 231-238*.

Krzyiak, B. and Linder, T. (1996). Radial Basis Function Networks and Nonparametric Classification: Complexity Regularization and the Rates of Convergence. In *Proc. Int'l Conf. on Pattern Recognition, vol. 4, pp. 650-653*.

Kubat, M. (1998). Decision Trees Can Initialize Radial-Basis Function Networks. *IEEE Trans. on Neural Networks, vol. 9 (5), pp. 813 - 821*.

Kuncheva, L.I. (2004). Combining Pattern Classifiers: Methods and Algorithms. *John Wiley & Sons*.

Kuncheva, L.I. and Whitaker, C.J. (2003a). Measures of Diversity in Classifier Ensembles. *Machine Learning, vol. 51, pp. 181-207.*

Kuncheva, L.I., Whitaker, C.J., Shipp, C.A. and Duin, R.P.W. (2003b). Limits on the Majority Vote Accuracy in Classifier Fusion. *Pattern Analysis and Applications, vol. 6(1), pp. 22-31.*

Kuncheva L.I. (2001). Combining classifiers: Soft computing solutions. *Pattern Recognition: From Classical to Modern Approaches*, World Scientific Publishing Co., Singapore, 427-452

Lam, L. and Suen, C.Y. (1996). Majority Vote of Even and Odd Experts in a Polychotomous Choice Situation. *Theory and Decision, vol. 41(1), pp. 13-36*.

Lam, L. and Suen, C.Y. (1997). Application of Majority Voting to Pattern Recognition: An Analysis of its Behavior and Performance. *IEEE Tran. on Systems, Man, and Cybernetics, vol. 27(5), pp. 553-568.*

Liu, Y. and Yao, X. (1998). Negatively correlated neural networks for classification.Proc. 3d International Symposium on Artificial Life and Robotics, pp. 736-739.

Lu, L., Zeng, X.Q., Wu, X.L. and Zhong, S. (2006). A Novel Ensemble Approach for Improving Generalization Ability of Neural Networks. In *Proc. 9th Int'l Conf. on Intelligent Data Engineering and Automated Learning, LNCS 5326, pp. 164-171.*

Ma, L., Wahab, A. and Quek, C. (2001). A Modified Generalized RBF Model with EM-based Learning Algorithm for Medical Applications. In *Proc. 19th IEEE Symposium on Computer-Based Medical Systems, pp. 291 - 296.*

Mak, M.W. and Cho, K.W. (1998). Genetic Evolution of Radial Basis Function Centers for Pattern Classification. In *Proc. of the IEEE Int'l Joint Conf. on Neural Networks, vol. 1, pp. 669-673.*

Margineantu, D.D. and Dietterich, T.G. (1997). Pruning Adaptive Boosting. In *Proc. 14th Int'l Conf. Machine Learning, pp. 211-218.*

Martınez-Munoz, G. and Suarez, A. (2007). Using Boosting to Prune Bagging Ensembles. *Pattern Recognition Letters, vol. 28(1), pp. 156-165*.

Martınez-Munoz, G. and Suarez, A. (2006). Pruning in Ordered Bagging Ensembles. In *Proc. 23rd Int'l Conf. Machine Learning, pp. 609-616.*

Martınez-Munoz, G. and Suarez, A. (2004). Aggregation Ordering in Bagging. In *Proc. Int'l Conf. Artificial Intelligence and Applications, pp. 258-263*.

Martınez-Munoz, G., Hernandez-Lobato, D. and Suarez A. (2009). An Analysis of Ensemble Pruning Techniques Based on Ordered Aggregation, *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 31 (2), pp. 245-259*.

Mason, L. Bartlett, P.L. and Golea, M. (1997). Generalization Error of Combined Classifiers. *Computer and System Sciences, vol. 65(2), pp. 415-438*.

MCS Workshop: http://www.diee.unica.it/mcs/

Meghabghab, G. and Kandel, A. (2004). Stochastic simulations of web search engines: RBF versus second-order regression models. *Information Sciences, vol. 159(1), pp. 1-28*.

Mendoza, A., Melin, P. and Licea, G. (2009). A Hybrid Approach for Image Recognition Combining Type-2 Fuzzy Logic, Modular Neural Networks and the Sugeno Integral. *Information Sciences, vol. 179(13), pp. 2078-2101*.

Merz, C. J. (1998). Combining Classifiers Using Correspondence Analysis. In *Proc. Conf. on Advances in neural information processing systems, pp. 591 - 597.*

MLR: http://archive.ics.uci.edu/ml/

Montazer, G. A., Sabzevari, R. and Ghorbani, R. (2009). Three-Phase Strategy for the OSD Learning Method in RBF Neural Networks. *Neurocomputing, vol. 72(7-9), pp. 1797-1802.*

Moody, J. E. (1991). Note on Generalization, Regularization and Architecture Selection in Nonlinear Learning Systems. In *Proc. 1991 IEEE Workshop Neural Networks for Signal Processing, pp. 1-10.*

Moody, J. E. and Darken, C. (1989). Fast Learning Networks of Locally-Tuned Processing Units. *Neural Computation, vol. 1, pp. 289-303*.

Musavi, M.T., Ahmed, W., Chan, K.H., Faris, K.B. and Hummels, D.M. (1992). On the training of radial basis function classifiers. *Neural Network, vol. 5, pp. 595-603*.

Ng, W.W.Y., Yeung, D.S., Firth, M., Tsang, E.C.C. and Wang, X.Z. (2008). Feature Selection Using Localized Generalization Error for Supervised Classification Problems using RBFNN. *Pattern Recognition, vol. 41(12), pp. 3706-3719.*

Ng, W.W.Y., Yeung, D.S., Wang, D., Tsang, E.C.C. and Wang, X.Z. (2007). Localized Generalization Error of Gaussian-based Classifiers and Visualization of Decision Boundaries. *Soft Computing, vol. 11(4), pp. 375-381.*

Ng, Y. (2004). Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In *Proc. 21st Int'l Conf. on Machine learning, pp. 78-85*.

Paradeda, R., Xavier Jr., J.C. and Canuto, A. M. P. (2008). Applying Static and Dynamic Weight Measures in Ensemble Systems. In *Proc. 10th Brazilian Symposium on Neural Networks, pp. 45 - 50.*

Poggio, T. and Girosi, F. (1989). A Theory of Networks for Approximation and Learning. *A. I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.*

Poggio, T. and Girosi, F. (1990). Networks for Approximation and Learning. In *Proc. of the IEEE, vol. 78 (9), pp. 1481 - 1497.*

Polikar, R., Krause, S. and Burd, L. (2003). Ensemble of Classifiers Based Incremental Learning with Dynamic Voting Weight Update. In *Proc. Int'l Joint Conf. on Neural Networks, vol.4, pp. 2770-2775.*

Prodromidis, A.L. and Stolfo, S.J. (2001). Cost Complexity-Based Pruning of Ensemble Classifiers. *Knowledge and Information Systems, vol. 3(4), pp. 449-469.*

Puuronen, S. and Tsymbal, A. (2008). Local Feature Selection with Dynamic Integration of Classifiers. In *Proc. Int'l Symposium on Foundations of Intelligent Systems, LNCS 1932, pp. 363-375.*

Puuronen, S., Terziyan, V.Y. and Tsymbal, A. (1999). A Dynamic Integration Algorithm for an Ensemble of Classifiers. In *Proc. 11th Int'l Symposium on Foundations of Intelligent Systems, LNCS 1609, pp. 592-600.*

Puuronen, S. and Tsymbal, A. (2001). Local Feature Selection with Dynamic Integration of Classifiers. *Fundamenta Informaticae, vol. 47(1-2), pp. 91-117.*

Rasheed, S., Stashuk, D.W. and Kamel, M.S. (2008). Diversity-based Combination of Non-Parametric Classifiers for EMG Signal Decomposition. *Pattern Analysis and Applications, vol. 11(3-4), pp. 385-408.*

Reyzin, L. and Schapire, R.E. (2006). How boosting the margin can also boost classifier complexity. In *Proc. 23rd Int'l Conf. on Machine Learning, pp. 753-760*.

Roli, F. and Fumera, G. (2002). Analysis of linear and order statistics combiners for fusion of imbalanced classifier. In *Proc. 3rd Int'l Workshop on Multiple Classifier Systems, LNCS 2364, pp. 252-261*.

Roli, F., Giacinto, G. and Vernazza, G. (2001). Methods for Designing Multiple Classifier Systems. In *Proc. 2nd Int'l Workshop on Multiple Classifier Systems, LNCS 2096, pp. 78–87.*

Rudin, C. and Schapire, R.E. (2004). Boosting Based on a Smooth Margin. In *Proc. 17th Annual Conf. on Learning Theory, LNCS 3120, pp. 502-517*

Ruta, D., and Gabrys, B. (2001). Application of the Evolutionary Algorithms for Classifier Selection in Multiple Classifier Systems with Majority Voting. In *Proc. 2nd Int'l Workshop on Multiple Classifier Systems, LNCS 2096, pp. 399–408.*

Santos, E.M.D., Sabourin, R. and Maupin, P. (2008). A Dynamic Overproduce-and-Choose Strategy for the Selection of Classifier Ensembles. *Pattern Recognition, vol. 41, pp. 2993-3009*.

Schwenker, F., Kestler, H.A. and Palm, G. (2001). Three Learning Phases for Radial-Basis-Function Networks. *Neural Networks, vol. 14(4-5), pp. 439-458*.

Sharkey, A.J.C., Sharkey, N.E., Gerecke, U. and Chandroth, G.O. (2000). The test-and-select approach to ensemble combination. In *Proc. 1st Int'l Workshop on Multiple Classifier Systems, LNCS 1857, pp. 30–44.*

Shirai, S., Kudo, M. and Nakamura, A. (2008). Bagging, Random Subspace Method and Biding. In *Proc. 2008 Joint IAPR Int'l Workshop on Structural, Syntactic, and Statistical Pattern Recognition, LNCS 5342, pp. 801-810.*

Skurichina, M. and Duin, R.P.W. (2000). The Role of Combining Rules in Bagging and Boosting. In *Proc. Joint IAPR Int'l Workshops SSPR 2000 and SPR 2000, LNCS 1876, pp. 631-640.*

Sohn, I. and Ansari, N. (1997). A Novel Algorithm to Configure RBF Networks. In *Proc. Int'l Conf. on Neural Networks, vol. 3, pp. 1809 - 1814.*

Sun, Y., Kamel, M.S. and Wong, A.K.C. (2005). Empirical Study on Weighted Voting Multiple Classifiers. In *Proc. 3rd Int'l Conf. on Advances in Pattern Recognition, LNCS 3686, pp. 335-344.*

Tikhonov, N. and Arsenin, V.A. (1977). Solutions of Ill-posed Problems. *Winston & Sons*.

Tsymbal, A. (2000). Decision Committee Learning with Dynamic Integration of Classifiers. In *Proc. East-European Conf. on Advances in Databases and Information Systems Held Jointly with Int'l Conf. on Database Systems for Advanced Applications, pp. 265-278*

Tsymbal, A., Pechenizkiy, M. and Cunningham, P. (2006). Dynamic Integration with Random Forests. In *Proc. 17th European Conf. on Machine Learning, pp. 801-808*.

Tsymbal, A., Pechenizkiy, M., Puuronen, S. and Patterson, D.W. (2003). Dynamic Integration of Classifiers in the Space of Principal Components. *Advances in Databases and Information Systems, pp. 278 - 292*.

Tsymbal, A. and Puuronen, S. (2000a). Dynamic Integration of Decision Committees. In *Proc. 7th Int'l Conf. on High Performance Computing, pp. 537-546.*

Tsymbal, A. and Puuronen, S. (2000b). Bagging and Boosting with Dynamic Integration of Classifiers. In *Proc. European Conf. on Principles of Data Mining and Knowledge Discovery, pp. 116 - 125*.

Tsymbal, A., Puuronen, S. and Terziyan, V. (1998). A Technique for Advanced Dynamic Integration of Multiple Classifiers. In *Proc. 5th Conf. on Artificial Intelligence, pp. 71-79*.

Tumer, K. and Ghosh, J. (1995). Order statistics combiners for neural classifiers. In *Proc. of the World Congress on Neural Networks, vol. 1, pp. 31-34.*

Tumer, K. and Ghosh, J. (1996a). Analysis of Decision Boundaries in Linearly Combined Neural Classifiers. *Pattern Recognition, vol. 29(2), 341–348*.

Tumer, K. and Ghosh, J. (1996b). Error Correlation and Error Reduction in Ensemble Classifiers. *Connection Science, vol. 8(3-4), pp. 385-403.*

Tumer, K. and Oza, N.C. (2003). Input Decimated Ensembles. *Pattern Analysis and Applications, vol. 6, pp. 65-77.*

Ueda, N. and Nakano, R. (1996). Generalization Error of Ensemble Estimators. In *Proc. of Int'l Conf. on Neural Networks, pp. 90-95.*

Vogt, M. (1993). Combination of Radial Basis Function Neural Networks with Optimized Learning Vector Quantization. In *Proc. Int'l Conf. on Neural Networks, vol. 3, 1841-1846.*

Watanabe, S. (2001). Algebraic Analysis for Non-Identifiable Learning Machines. *Neural Computation, vol. 13, pp. 899-933*.

West, D. and Dellana, S. (2009). Diversity of Ability and Cognitive Style for Group Decision Processes. *Information Sciences, vol. 179(5), pp. 542-558*.

Woloszynski, T. and Kurzynski, M. (2006). Application of Combining Classifiers Using Dynamic Weights to the Protein Secondary Structure Prediction - Comparative Analysis of Fusion Methods. In *Proc. 7th Int'l Symposium Biological and Medical Data Analysis, pp. 83-91*.

Wolpert, D.H. (1992). Stacked Generalization. *Neural Networks, vol 5, pp. 241-259.*

Wolpert, D.H. and Macready, W.G. (1999). An Efficient Method To Estimate Bagging's Generalization Error. *Machine Learning, vol. 35(1), pp. 41-55*.

Woods, K., Kegelmeyer Jr., W.P. and Bowyer, K. (1997). Combination of Multiple Classifiers Using Local Accuracy Estimates. *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 19(4), pp. 405-410*.

Xue, H., Chen, S. and Zeng, X. (2008). Classifier Learning with a New Locality Regularization Method. *Pattern Recognition, vol. 41(5), pp. 1496-1507*.

Yeung, D.S. and Ng, W.W.Y. (2005). Feature Selection Using Generalization Error for Supervised Classification Problem. In *Proc. Int'l Conf. on Instrumentation, Control and Information Technology, pp. 41-46*.

Yeung, D.S., Ng, W.W.Y., Wang, D., Tsang, E.C.C. and Wang, X.Z. (2007). Localized Generalization Error Model and Its Application to Architecture Selection for Radial Basis Function Neural Network. *IEEE Trans. on Neural Networks, vol. 18(5), pp. 1294-1305*.

Yoo, J.H. and Sethi, I.K. (1995). Design of radial basis function networks using decision trees. In *Proc. IEEE Int'l Conf. on Neural Networks, vol. 3, pp. 1269-1272*.

Young, P.M. and Trevor, H. (2007). L1-Regularization Path Algorithm for Generalized Linear Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 69(4), pp. 659-677*.

Yule, G. (1900). On the Association of Attributes in Statistics. *Phil. Trans., A, 194, pp. 257–319*.

Zanda, M., Brown, G., Fumera, G. and Roli, F. (2007). Ensemble Learning in Linearly Combined Classifiers Via Negative Correlation. In Proc. 7the Int'l Workshop on Multiple Classifier Systems, LNCS 4472, pp. 440-449.

Zhou, Z.H. and Tang, W. (2003). Selective Ensemble of Decision Trees. In *Proc. 9th Int'l Conf. Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, pp. 476-483*.

Zhou, Z.H., Wu, J. and Tang, W. (2002). Ensembling Neural Networks: Many Could Be Better than All. *Artificial Intelligence, vol. 137(1-2), pp. 239-263*.