Assessment of Two AI Approaches to Predict Mortality in Adult Intensive Care Units

by

Hui Li

A thesis submitted to

The Faculty of Graduate Studies and Research

In partial fulfillment of the requirements for the degree of

Master of Applied Science

in Electrical Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario, Canada

September 2005

Acknowledgements

There are a lot of people I would like to thank for a variety of reasons.

I would like to begin by thanking my thesis supervisor, Dr. Monique Frize for her guidance and support, as well as giving me the opportunity to be part of this very exciting research group. Her common-sense, knowledge and perceptiveness truly inspired me during this work.

Thank you as well to Prof. Tet Hin Yeap for the valuable suggestions and technical discussion on the RBF networks. I would like to acknowledge your enthusiasticsupport.

I'm also grateful to Christophe Herry and Doaa Swailum Ibrahim from the MIRG laboratory. Thank you very much for your help with experimental setup and general advice.

Finally, I would also like to thank all the rest of the academic and support staff of the Ottawa-Carleton Institute for Electrical and Computer Engineering for the assistance during my study in this wonderful institute.

To my dearly beloved daughter

Nan Nan

Abstract

Mortality prediction for ICU patients has been of particular interest in medical applications. Artificial Neural Networks has emerged as an important technique in the medical decision support field in recent years. Traditionally, the Multi-Layer Perceptron (MLP) network using a back-propagation learning algorithm has been the most popular model used in the medical domain, while the Radial Basis Function (RBF) network has been rarely used in this field. This thesis discusses several training algorithms of RBF networks, including the K-mean clustering method and the forward selection ROLS method. A new approach using K-mean clustering to initialize the centre candidates of the ROLS method is presented. Technologies including weight-elimination, regularisation and data analysis (PCA) are applied for optimisation purposes. The experimental results show that weight-elimination (or regularisation) is suitable for solving the over-fitting problem, thus improving the model's performance. The PCA method can improve the prediction quality of the RBF network, especially in the detection of non-survival cases which have a low occurrence. The performance of the MLP and RBF networks is compared and discussed. With respect to the problem studied in this thesis, among all the models under consideration, the MLP model performs well overall. The RBF model with K-mean clustering is unstable and performs relatively poorly. The advanced RBF model using ROLS with a K-mean centre candidate initialization plus a PCA approach performs the best.

Content:

Chapter 1 Introduction 1
1.1 Motivation 1
1.2 Problem identification and thesis objectives 3
1.3 Adult ICU medical environment 7
1.3.1 Diagnostic sensitivity and specificity 7
1.3.2 Traditional illness scoring systems in adult ICUs 9
1.3.3 Acute Physiology And Chronic Health Evaluation (APACHE) II 10
1.3.4 Some mortality prediction facts 11
1.4 A brief literature overview of mortality prediction 12
1.5 MIRG's medical software and application environment 14
Chapter 2 Background information 17
2.1 Artificial Neural Networks 17
2.1.1 Overview of Artificial Neural Networks 17
2.1.2 Multi-layer Perceptron Network 20
2.1.3 Weight-elimination/Weight-decay for performance optimisation 22
2.1.4 Parameters to specify a MLP network 24
2.2 Radial Basis Function networks 28
2.2.1 Design philosophy of RBF networks as a classifier 28
2.2.2 Parameters to specify an RBF network 31
2.2.3 A brief overview of training algorithms for RBF networks 31
2.2.4 Regularized Orthogonal Least Squares (ROLS) method 36
2.3 Cross-validation for error estimation 41
2.4 Principal Component Analysis (PCA) for dimension reduction 42
2.5 A brief comparison of the training algorithms of MLP network and the RBF network 45
Chapter 3 Methodology 48
3.1 Data pre-processing 48
3.1.1 Normalize variables of the database 52
3.1.2 Divide the whole dataset into training data and test data 52
3.1.3 Artificially increase the mortality rate of the training dataset 52
3.2 Measures of performance 53
3.2.1 Contigency table 54
3.2.2 Log-sensitivity index 55
3.2.3 Constant predictor 56
3.3 Implementing the MLP network system 56
3.3.1 Network structure 56
3.3.2 Automation of parameter computing 57
3.3.3 Parameter initialization 59
3.4 Implementing the RBF network system 60
3.4.1 A conventional K-mean approach 61
3.4.2 Regularized Orthogonal Least Squares (ROLS) approach 62
3.4.3 ROLS approach with a K-mean centre initialization method 63
Chapter 4 Results and discussion 65
4.1 Results of the MLP network method 66
4.1.1 Results of the 3-layer MLP model without weight-elimination (MLP1) 66
4.1.2 Results of 3-layer MLP model with weight-elimination (MLP2) 68
4.1.3 Results of dimension reduction with the MLP network (with weight-elimination) (MLP3) 70
4.2 Results of RBF network method 72
4.2.1 Results of the RBF network using a K-mean method (RBF1 & RBF2) 72
4.2.2 Results of the RBF network using an OLS method with a naïve centre initialization (RBF3, RBF4 & RBF5) 74
4.2.3 Results of the RBF network with centre initialization using K-mean method (RBF6) 77
4.2.4 Results of dimension reduction using PCA (RBF7) 80

4.3 The experimental results with the non-operative database 81
4.4 Discussion of the experimental results 83
Chapter 5 Conclusions **86**
5.1 Concluding remarks 86
5.2 Contributions to knowledge 87
5.3 Future work 89

Table:

Table 1-1 Scoring for the APACHE II model 11
Table 3-1 Statistics of the DECH ICU database 49
Table 3-2 Summary of variables used in the DECH ICU database 51
Table 3-3 Contingency (True) table 54
Table 3-4 Initial parameter values (MLP) 60
Table 4-1 The final parameter values of the best MLP model (without weight-elimination) 67
Table 4-2 Performance of MLP models without weight-elimination 67
Table 4-3 The final parameter values of the MLP model with weight-elimination 70
Table 4-4 Performance of MLP models with weight-elimination 70
Table 4-5 Performance of the MLP model (with weight-elimination) + PCA 72
Table 4-6 The results of the RBF network with a K-mean method 74
Table 4-7 The results of the RBF model with a K-mean+PCA method 74
Table 4-8 The performance of the RBF model using naïve OLS method 76
Table 4-9 The performance of the RBF network using ROLS method with a naïve centre initialization 76
Table 4-10 The performance of the RBF network using ROLS method with a naïve centre initialization +PCA 77
Table 4-11 The performance of the RBF network using ROLS method with a K-mean centre initialization 79
Table 4-12 The performance of the RBF network using ROLS method with a K-mean centre initialization + PCA. 80
Table 4-13 Summary of the experimental results with non-postoperative dataset 82
Table 4-14 A brief summary of the performance of the models with the post-operative dataset 84

Figures of experimental results:

Figure 4-1 A typical training procedure of the MLP network without weight-elimination 68
Figure 4-2 A typical training procedure of the MLP network with weight-elimination 69
Figure 4-3 A typical procedure of the MLP network with weight-elimination +PCA 71
Figure 4-4 A typical training procedure of the RBF network with a K-mean method 73
Figure 4-5 A typical training procedure of the RBF network using the naïve OLS method 75
Figure 4-6 A typical training procedure of the RBF network using ROLS method with a naïve centre initialization 76
Figure 4-7 A typical training procedure of the RBF network using ROLS method with a K-mean centre initialization 78
Figure 4-8 A typical training procedure of the RBF network using ROLS method with a K-mean centre initialization + PCA 81

Abbreviation:

AI: Artificial intelligence

APACHE: Acute Physiology and Chronic Health Evaluation

ANN: Artificial Neural Network

CCR: Correct Classification Rate

ECG: Electrocardiography

EM: Expectation maximization

ER: Emergency Room

DECH: Dr. E.Chalmers Hospital

GCS: Glasgow Coma Score

ICU: Intensive Care Unit

MIRG: Medical Intelligence Research Group

MLP: Multi-layer Perceptron

MPM: Mortality Probability Model

NICU: Neonatal Intensive Care Unit

Post-op: Post-operative

RBF: Radial Basis Function

ROLS: Regularized Orthogonal Least Squares

SAPS: Simplified Acute Physiology Score

SSE: Sum of Square Error

Chapter 1 Introduction

1.1 Motivation

In an Intensive Care Unit (noted hereafter as ICU), patients are usually very ill and clinical complications may occur at any moment. Fast-pace decision-making is required in many acute cases. Unfortunately, the medical data of an ICU patient are usually complex and require time-consuming pre-processing, making a rapid diagnosis difficult. Moreover, researches showed that the iatrogenic illness is more likely to happen in the ICU settings than in other medical units [58;59]. It seems that iatrogenic illness and other forms of errors in ICUs can be linked to limitations in human decision making [42;58;59]. Under such circumstances, if a medical aid system was able to quickly and accurately predict medical outcomes such as the risk of death for an ICU case, it could help health staff to plan proper treatment, saving more lives and significantly increasing patients' chances of survival. In this way, rapid clinical decision support systems become highly desirable.

In past years, a variety of techniques have been applied to predict medical outcomes. Significant effort has been devoted to identifying the risk factors influencing mortality rates and other medical outcomes [4;13;21;36;56;68]. Risk factors such as age, chronic disease, clinical parameters, surgical status and acute problems have been used to make medical diagnoses. However, how these factors affect medical outcomes is still uncertain. Scoring systems that index the severity of illness were one of the earliest tools used to aid decision-making, and are still widely used in clinical practice [34]. For example, the

APACHE score is used with adults in ICU [16;46-48;79] and the SNAPS score is used with neonates in NICU (neonatal ICU) [67].

A variety of mathematical and statistical methods for pattern classification, such as discriminant analysis, logistic regression and recursive partitioning, have also been applied to develop clinical decision support models [17;61]. These methods can potentially improve diagnostic accuracy. However, due to the lack of knowledge concerning the mathematical relationship between the medical variants, a discriminant function or decision boundary is not easy to determine, constraining the use of these methods [17;61].

Artificial intelligence (AI) has been applied in the medical field for over 30 years. In 1984, Clancey and Shortliffe defined Medical Artificial Intelligence as follows:[15]

"Medical artificial intelligence is primarily concerned with the construction of Artificial Intelligence programs that perform diagnosis and make therapy recommendations. Unlike medical applications based on other programming methods, such as purely statistical and probabilistic methods, medical Artificial Intelligence programs are based on symbolic models of disease entities and their relationship to patient factors and clinical manifestations."

The ICU environment is particularly suitable to the use of AI tools because there is a large amount of available data and it may improve the quality of patient care [38;76;77]. Artificial Neural Networks (noted hereafter as ANNs) have attracted significant attention

due to their ability to learn and generalize in terms of self-learning, their fault-tolerance and predictive accuracy. Multi-layer perceptron (noted hereafter as MLP) networks using back-propagation training are the most commonly used neural network type in the medical field. This network is considered quite appropriate for complex problems that cannot be defined precisely, as is often the case in medical data analysis. MLP networks have been applied successfully in many areas of clinical decision support [15;19;34;38]. Another successful technique which originated from multivariable interpolation is the Radial Basis Function (noted hereafter as RBF) technique, which can also be interpreted as an ANN that uses radial symmetrical functions in the hidden layer to perform a nonlinear transformation on the input data. The RBF technique has performed well in pattern classification yet is not seen as much as MLP in the medical field. Generally speaking, RBF networks have high convergence speed, global minimum convergence and few parameters to be estimated [40]. These characteristics may well suit the requirements of clinical decision-making in an ICU, and it is expected to help make rapid mortality predictions that are critical for an ICU patient.

1.2 Problem identification and thesis objectives

Medical decision support systems are not a replacement for doctors but a decision tool to quickly provide a likely medical outcome and assist doctors in making informed decisions. These systems simulate the procedures conducted by medical personnel to draw knowledge from a large medical database and quickly make a recommendation to the doctor. The decision support problem consists of building a classification model using

attributes extracted from medical records, then evaluating the effectiveness of the model using new cases [17].

The basic procedures of implementing a medical decision aid system are illustrated in Figure 1-1. Here we use neural networks as an example. First, data with known outcomes must be collected from a representative population of cases and pre-processed for later use. Then, the neural network is trained with these known data. The architecture and parameters are adjusted during the training time until a pre-defined goal is achieved. After training, new data is fed to the network to test the performance of the trained network. During this period, the performance of the network is audited and appropriate modifications are done to try on different architectures (parameters) according to the audit's results. Experimental results are collected for different architectures of the network and assessments of the neural network are done by some criteria in the medical situation. The best model then is chosen from all the models of different architectures (parameters) according to this specific criterion.

An obvious challenge for a medical aid system is that the medical database is usually highly skewed because some medical outcomes such as mortality are usually rare. For example, in most patient population in ICUs, the mortality rate is below 10%. Such a low occurrence in a database is hard to detect but it is of interest in the medical field.

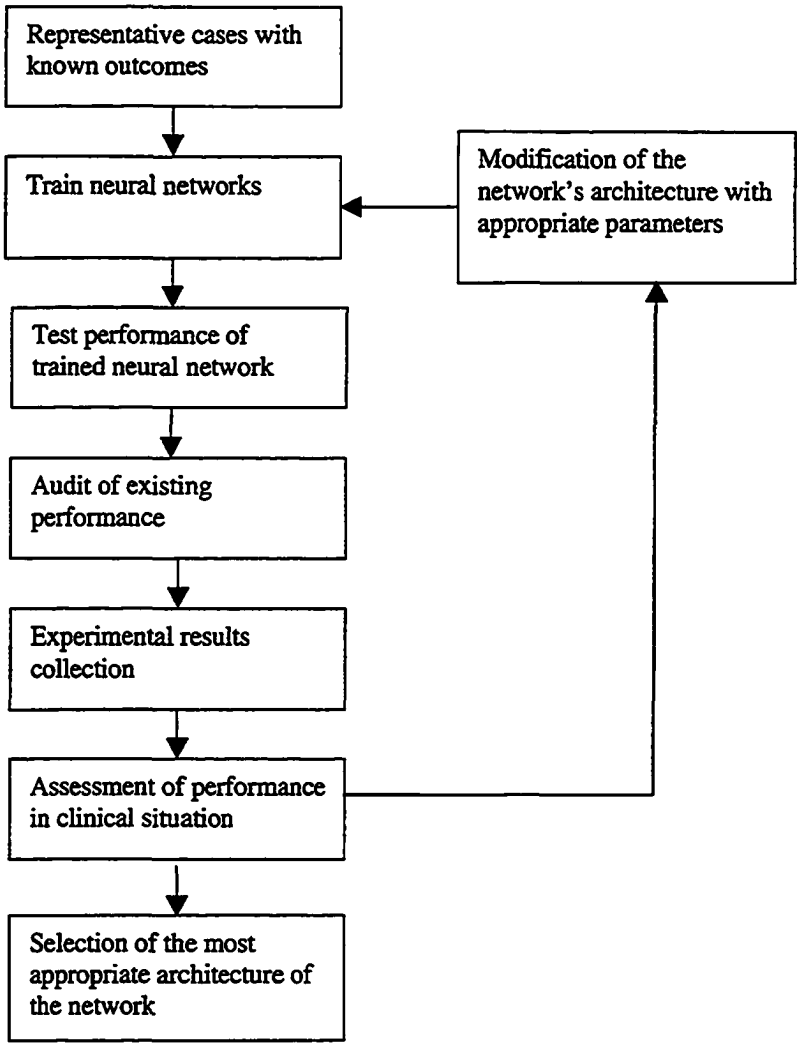

Figure 1-1 Flow chart of training and development processes for implementing a neural network medical decision aid system

Another challenge is how to train the medical system and how to evaluate its performance. Unlike other pattern classification problems in which classification performance can be measured by the errors between the expected outputs and the predicted outputs, an ideal mortality prediction system in ICU should have not only a high predicting accuracy but also a high sensitivity and specificity (see section 1.3.1).

Thus, the training of the medical aid system is a little different from that of ordinary pattern classification systems. Measures combining predictive accuracy, sensitivity and specificity are used to evaluate the performance of the medical support system, and the criteria for stopping training strategies vary depending on different measures.

Mortality prediction can be regarded as a typical binary pattern classification problem based on medical variables where the only output variable (mortality) has 2 values: yes or no. From this point of view, MLP networks and RBF networks are two good candidates for our problem. The efficiency of MLP networks has been demonstrated by many applications, so it is studied in this thesis as one approach to solve our problem. Meanwhile, although most previous medical research was focused mainly on MLP networks, RBF networks as another good pattern classifier is also of great interest to researchers. The second objective of this research is to apply the RBF network technique to determine whether it can efficiently predict mortality in the ICU, and to compare its behaviour with the commonly used MLP network. Some techniques for training RBF networks will be discussed to explore the best mechanism to apply to our problem. The performance of the two approaches will be evaluated based on experimental results with actual data from a comprehensive electronic medical record system. In addition, since ensembles of models are more accurate than single models in their predictive ability [55], a system assembling all the models will be implemented to predict mortality to benefit from the strengths of both techniques and provide comparable outcome estimations for the medical personnel. To evaluate performance, measures such as the correct

classification rate, sensitivity and specificity will be applied. Other comparisons will be done regarding the approximation quality of models and the speed of convergence.

1.3 Adult ICU medical environment

In order to better understand the medical background of our medical decision support system, a brief review of adult intensive care facts and terms is presented in this section.

1.3.1 Diagnostic sensitivity and specificity

In clinical practice, there are always some patients who are diagnosed as positive but do not have the disease, and some patients who are diagnosed as negative but actually have the disease. These cases are called false positive and false negative, respectively. Similarly, those who are diagnosed as positive and actually have the disease are called true positive, while those who are diagnosed as negative and actually don't have the disease are called true negative. A false positive diagnosis may result in unnecessary treatment, but a false negative diagnosis could be worse in cases where the proper treatment opportunity is missed, resulting in higher treatment costs, and in some cases death.

Medical studies usually use sensitivity, specificity and correct classification rate to measure the quality of a diagnosis. Sensitivity refers to the percentage of true positives among patients who have the disease. To be specific, it is the number of true positives divided by the total number of patients with the disease:

$$\text{Sensitivity} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{1-1}$$

The specificity is the percentage of true negatives among patients without the disease.

$$\text{Specificity} = \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}} \tag{1-2}$$

The correct classification rate (CCR) is the percentage of all patients who are correctly diagnosed.

$$\text{CCR} = \frac{\text{true positives} + \text{true negatives}}{\text{all cases}} \tag{1-3}$$

Although the above three parameters are all indices of system performance, it should be noticed that none of them can be used alone. They depend on the choice of the medical decision threshold value, and there is a "trade-off" between them. An extreme example of this would be setting the diagnosis threshold value at zero, so everyone would have a "negative" result. In spite of the high classification rate and the high specificity of 100%, the sensitivity could be as low as 0%.

1.3.2 Traditional illness scoring systems in adult ICUs

ICU patients are often at risk of dying and require a high level of intensive care. Multiple system dysfunctions and multiple medical problems can coexist, and medical conditions are usually complicated and severe. There is an interest in examining the risk factors influencing the mortality of ICU patients because these risk factors may help us understand individual differences in response to illness and help develop interventions intended to decrease mortality.

Previous analysis on identifying special risk factors influencing mortality in ICUs has mainly focused on both physiological and clinical characteristics. Usually, the following variables are recorded in intensive care: gender, age, duration of ICU stay, location before ICU admission, type of patient (surgical/medical), existing medical problems at admission, etc. These factors are intuitively related to ICU mortality [4;13;68].

Since the 1980s, scoring systems consisting these risk factors have been used to evaluate the severity of illness [34]. APACHE (Acute Physiology And Chronic Health Evaluation) [16;46-49;61;79] is one of the earliest and the most commonly used mortality score models in adult ICU. Following the invention of APACHE and its publication in the early 1980s, the SNAP (Score for Neonatal Acute Physiology) [67], TISS (Therapeutic Intervention Scoring System) [44] and MPM (Mortality Probability Models) [51]systems were developed for medical outcome prediction. In the later ten years, these instruments developed and new generations using sophisticated statistical techniques were found to outperform their

older counterparts. Next, we will introduce the widely-used APACHE II scoring system in detail.

1.3.3 Acute Physiology And Chronic Health Evaluation (APACHE) II

The APACHE II score is determined from points based upon the patient's chronic health history, age and acute physiology variables. All variables are collected from the worst values within 24 hours of ICU admission. The 12 acute physiology variables form a minimum set of key risk variables that must be measured to maintain statistical precision. Age and chronic medical histories are also included. Hence, these 12 acute physiology variables, combined with age and chronic medical problems, are scored to represent the degree of illness severity (Table 1-1). With the exception of Glasgow Coma Score (GCS), evaluated as 0-12, and serum creatinine, measured as 0-8, all other acute illness measures are evaluated using a 0-4 scale. Age and chronic health history are assigned 0 to 6 points according to the individual record. With this system, it has been possible to describe patients' characteristics despite enormous variance between individuals.

Many studies have shown that there is a significant correlation between the APACHE II score and ICU mortality. Patients with higher APACHE II scores have a greater chance of dying in the hospital [79]. APACHE II nowadays is one of the most widely used scoring models in clinical practice and validated in clinical trials.

Variable	No. of points
Acute physiology	
rectal temperature	0-4
mean arterial pressure	0-4
heart rate	0-4
respiratory rate	0-4
fraction of inspired oxygen	0-4
partial pressure of oxygen in the blood	0-4
arterial pH	0-4
serum sodium	0-4
serum potassium	0-4
white blood cell count.	0-4
serum creatinine	0-8
Glasgow Coma Score (GCS)	0-12
Age (year)	
<44	0
45-54	2
55-64	3
65-74	5
>75	6
Chronic health history	
Non-operative or emergency operative admission	5
Elective post-operative admission	2
Maximum possible total	71

Table 1-1 Scoring for the APACHE II model

1.3.4 Some mortality prediction facts

There are obvious differences in the behaviour of individual patients. Studies have shown that, compared with surgical (post-operative) patients, medical (non-operative) patients in critical condition have a higher death rate in ICUs [4;13]. The fact affected in APACHE II is that the medical patients are assigned more points than the surgical patients (Table 1-1). One possible reason for this fact is that medical patients may have more complex and severe coexisting problems on admission to ICU. Thus, it is very important to distinguish

these two groups of patients because they have different characteristics and must be analysed separately in medical studies.

Another fact revealed by a series of studies is that clinicians using scoring systems may overestimate mortality in high-risk patients but underestimate mortality in low- to moderate-risk groups [18]. The former situation happens more often in the ICU environment because most of the patients are high-risk. The reason clinicians might be prudent with their diagnoses is that there are always some discrepancies between current models and an ideal model. Overestimation of risk means more costly intensive care and transfer, according to empirical studies. A model with high specificity and reasonable sensitivity, meaning more survival patients are correctly classified, would be beneficial and complementary to clinicians' diagnoses, which are often overestimated. In practice, a model with a specificity of close to 100% and a sensitivity of over 50% is acceptable for medical support purposes [25].

1.4 A brief literature overview of mortality prediction

According to the literature, prognostic scoring systems which quantify the severity of a disease are still among the primary tools used in clinical practice to predict mortality [3;13;27;47]. Some evaluations of the mortality predictive accuracy of these scoring systems suggest that it may be necessary to adapt the scoring models to the patient population to increase the mortality prediction accuracy in particular medical settings [37;65].

Logistic regression and multivariate analysis are also widely used [3]. One example is the Bayes classifier. For instance, M. Ramoni *et al.* implement a so-called robust Bayes classifer and apply it to mortality prediction in ICUs [66]. This classifier, which is based on the naive Bayes classifier, is superior to the median imputation approach in that it does not assume the pattern of the missing data in the ICU setting. Some logistic regression models are used in combination with prognostic scores. For example, Le Gall JR *et al.* propose a SAPS-II logistic regression model which takes the SAPS-II score as a covariate [49;50]. Although logistic regression and multivariate analysis outperform the scoring system, they are still restricted by the limited medical statistical techniques available [3].

Recent efforts have focused on Artificial Intelligence. A number of typical AI technologies are used to predict mortality. Laboratory and clinical evaluations have shown that they can improve mortality prediction efficiently. Examples of these AI approaches include decision trees [2;45], Bayesian networks [43;60], and neural networks [19;22;34;80]. A typical neural network type commonly used in clinical practice is the MLP network with a back-propagation learning algorithm, which is applied to a broad spectrum of fields including intracerebral hemorrhage (ICH)[23], cancer[10;19], coronary heart disease[19], sepsis[27] and the ICU [14;33;61;80] etc..

Another AI approach, RBF networks, which can also be interpreted as a neural network, has also been applied successfully in medical domains such as medical decision support for traumatic brain injury patients [54], disease diagnosis from electrocardiography (ECG) [9;28;53;53], predicting heart rate [39] , analysis of the dynamics of the heart rate variability

(HRV) signal[7;63], multispectral brain MRI segmentation [75], spectroscopic detection of cervical pre-cancer [74] and estimation of evoked potentials [35] . However, they are less exploited in ICUs. E.Blanzieri *et al.* proposed an RBF architecture model in 1995 to predict the mortality of ICU patients [8]. This model was based on the Factorized Radial Basis Function Networks (F-RBFNs) and the fuzzy neural networks. As for their experiments, the MLP and RBF models had similar predictive accuracy.

1.5 MIRG's medical software and application environment

This thesis uses actual data from a large medical database of the ICU at the Dr. E. Chalmers Hospital (hereafter noted as DECH) in Fredericton [69]. The original DECH ICU database consists of over 3000 records of patients admitted to DECH ICU over 2.5 years (since 1998). There were 98 fields of clinical and administrative information including multiple procedural information, up to seven medical diagnoses, and auxiliary space for free form comments. Significant events and complications that occur in the ICU course were also recorded [69]. A subset of this database with the raw APACHE II variables was used in later research by the MIRG laboratory. Fifty-one input variables were extracted from the original database. Profiles with missing values or under the age of 12 were excluded, which resulted in a database with 1491 cases [32;72]. This database is used in this thesis.

The MIRG (Medical Intelligence Research Group) has previously implemented MLP network models to predict medical outcomes in ICU. In 1995 [32], they proposed the initial ANN model in MIRG and applied it to predict "Mortality", "Length of stay" and

"Duration of artificial ventilation". This study showed that ANN is an efficient method for medical decision support.

In 1997, Trigg *et al.* [72;73] tried two new approaches on the previous 2-layer and 3-layer MLPs to improve their performance. These two approaches are weight-elimination and high/low node representation. New models were applied to predict the "Duration of ventilation". The experiments using weight-elimination improved the performance of the previous model significantly while the high/low node representation performance showed not much difference from the regular data representation for that study.

Ennett [24] discussed how the data distribution and representation affect the classification performance of ANN in 1999. She obtained the best performance when training the network on a higher-than-normal prevalence approach, i.e., increasing the mortality rate of the training dataset artificially by randomly adding/copying some positive (non-survivor) cases into the training dataset. She applied the ANN model to predict coronary surgery mortality where the actual mortality rate in the database was quite low. Her study also used weight-elimination and performed well in mortality prediction. This model was later applied to predict mortality and ventilation hours in ICUs by the MIRG laboratory [33;76].

Later in 2001, Scales proposed a performance measurement tool which considers both sensitivity and specificity in a single index and balances them to favour sensitivity. The

equation of this measurement comes in a logarithm form and was named the "log-sensitivity index" (Section 3.2.2).

This study is conducted based on the above research achievements in MIRG. The data pre-processing is done following the methods previously used by MIRG, and the MLP model is implemented using the existing programs. The new RBF models are implemented based on some publicly available algorithms and codes. To provide a consistent standard to compare the different models, new functions are created to collect the experimental results and measure the performance. All the models are implemented into one system and use the same functions to pre-process data, collect experimental results and measure the performance. All programs are implemented and run in the environment of Matlab 6.0 in the MIRG laboratory.

Chapter 2 Background information

2.1 Artificial Neural Networks

An artificial neural network is a parallel, highly integrated system that simulates the learning ability of human brains.

2.1.1 Overview of Artificial Neural Networks

The nervous system of living organisms is a collection of about 10 billion connected elements (neurons) working in parallel. An artificial neuron works in a similar way as a biological neuron does. In an artificial neuron (Figure 2-1), weighted inputs (weights in ANN are similar to synaptic connections in the human brain) are sent to a node (similar to a neuron in the brain), are summed up and then transformed to a limited range by the neuron's transfer function. This function may come in three forms: a certain threshold level, a linear function, or a sigmoid function. The output from the transfer function is then sent to other nodes (neurons) connected with this node (neuron). This value can be calculated using the following expression:

$$A_O = \mathbf{f}\left(\sum_{1}^{p} W A_I + \mathbf{b}\right) \qquad (2\text{-}1)$$

where A_I is the p inputs of the neuron, A_O is the output of the neuron, b is the bias node, f is the transfer function.

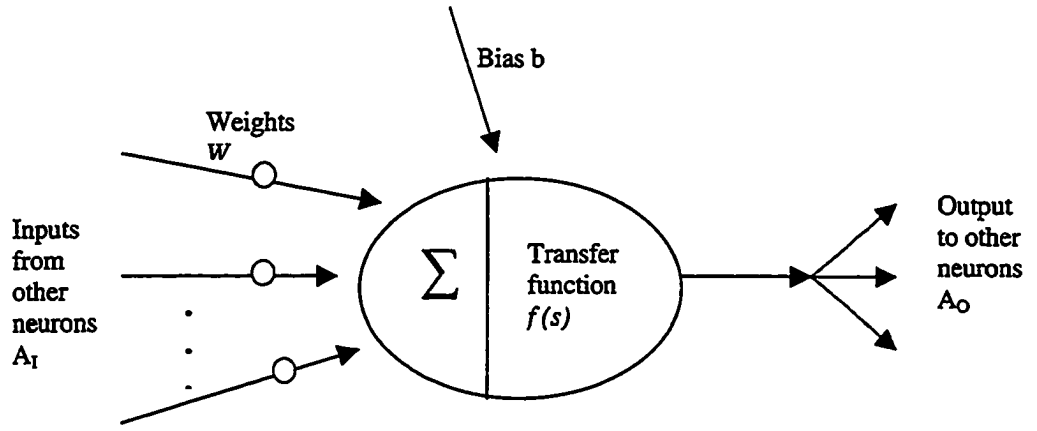

Figure 2-1 The structure of a neuron

An individual neuron is not very useful. However, when many neurons are highly interconnected, the collective power of the network as a whole is enormous. Every neuron in the network continuously evaluates its output according to instantaneous inputs from other neurons. The network continuously evolves until all neurons are in stable states. These procedures simulate the way that the human brain works. In another word, the highly interconnected neural network has the ability to learn. During the learning procedures, weights are adjusted repeatedly. The more important information is assigned a larger weight and less important information is assigned a smaller weight. Learning is accomplished with the evolvement of the whole network.

When an ANN is used in pattern classification, two phases are involved: the training/learning phase and the generalization/test phase. During the training phase, the network is trained by a specific algorithm which can either be supervised or unsupervised. If the expected output is already known before training and is needed to

help adjust the parameters of the network, the training is called supervised. Otherwise, it is called unsupervised.

Supervised ANNs are usually used to classify patterns into known categories. During the learning process, input patterns are propagated through the network from the input layer to the output layer. An output pattern is generated at the output layer. The difference (defined by some mathematical criteria) between actual output and expected output is then used to compute an error. This output error indicates the network's learning effectiveness and is used to adjust the parameters of the ANN to obtain smaller errors.

Neural networks that learn unsupervised do not have such expected outputs. The learning goal is to group similar patterns close together in certain areas of the value range. This can be used efficiently for pattern clustering purposes.

After the training, the network is in a stable status in which the weights will no longer change. We call this network a trained network. Then comes the generalization phase during which the trained network is used to generalize the information for new input patterns. Since the trained network has learned many patterns and is already stable, it can predict the pattern that the new data belongs to based on its learnt knowledge.

There are a variety of structures of ANNs. Figure 2-2 is a typical topological structure of one category of Artificial Neural Network: the Multi-layer feed-forward network. If we see the RBF network as an ANN, both MLP and RBF can be illustrated with this figure.

In this category of ANN, information is fed forward within the network and neurons are activated layer by layer. Neglecting the fact that the hidden layers of MLP and RBF work in different ways, the only difference in the structure is that an MLP may have multiple hidden layers while an RBF almost always has only one hidden layer, meaning three layers in total.

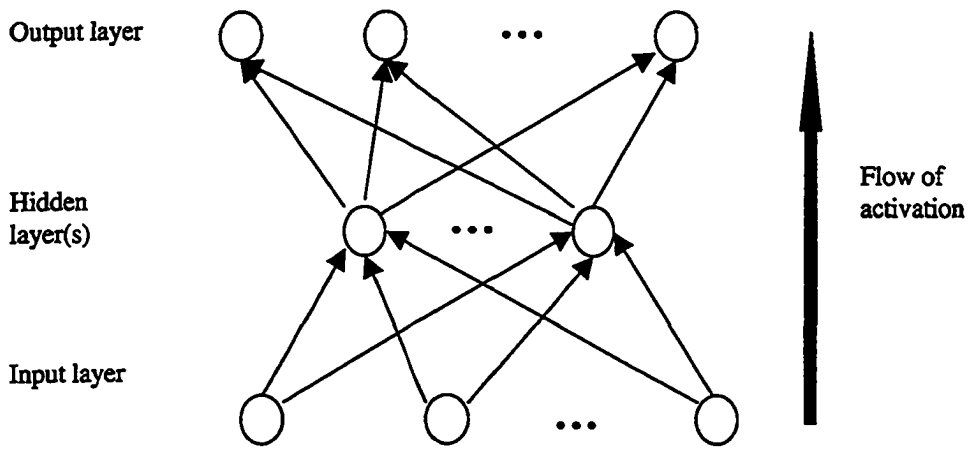

Figure 2-2 Topological structure of a feed-forward ANN

2.1.2 Multi-layer Perceptron Network

The MLP network is a multi-layer supervised network that has no less than one hidden layer. It is a typical feed forward network, in which the information is fed from the input layer to the first (and possibly only) hidden layer, then to the second hidden layer, and so on until the output layer.

MLP is a non-linear system. In a single-layer perceptron with no hidden layer, the output of the network is linear in the weights. However, when a hidden layer is added to the network, the system is no longer linear. Typically, the hidden layer(s) learns to recode the

inputs. With the hidden layer(s), there could be more mapping from the input data to output data thus the network is more powerful than a single-layer network.

MLP networks normally use a back-propagation learning algorithm to adjust weights. The adjustment is determined using an amount proportional to the error at each iteration. Specifically, each iteration of the learning phase consists of two passes: a forward pass and a backward pass. The forward pass refers to the process the input data is forwarded from the input layer to the hidden layer(s), and then to the output layer. The outputs of each unit are calculated and the errors at the output units are calculated in this pass. In the backward pass, the errors calculated at the output layer from the forward pass are propagated backward. The weights are altered so that the errors (expressed in the cost function) become smaller by performing the error gradient descent. This is repeated over and over again until the error is lower than a threshold, which signals the end of the learning phase.

Output of each layer in forward pass:

$$A_n = \tanh\left(\sum_{i=1}^{p} W_n[i]A_{n-1}[i] + b_n[i]\right),\ n \geq 1 \qquad (2\text{-}2)$$

Cost function at the output layer:

$$E(W) = \sum_{i=1}^{p} (E_0(W))^2 + \lambda \sum \frac{\frac{w_{ij}^2}{w_0^2}}{1+\frac{w_{ij}^2}{w_0^2}} \qquad (2\text{-}3)$$

Weight adjusted in backward pass:

$$dW_n = m*dW_n + (1\text{-}m)*lr*D_n*A_{n-1}^T - 2\lambda \left[\frac{w_0^2 W_{nij}\left(1+\frac{Wn_{ij}^2}{w_0^2}\right) - W_{ij}^3}{w_0^4\left(1+\frac{W_{nij}^2}{w_0^2}\right)^2} \right] \qquad (2\text{-}4)$$

Later in the test phase, new patterns are fed into the input layer and forwarded to the hidden layer(s), and then the output layer. The outputs of each layer are determined by the input to this layer and the weights between the previous layer and this layer. The final predicted results for the test data are achieved at the output layer.

2.1.3 Weight-elimination/Weight-decay for performance optimisation

Large weights can impair the generalization ability of ANNs since they can cause excessive variance in the output (Geman, Bienenstock and Doursat 1992). Small weights, on the other hand, can introduce "white noise" into the data [1]. According to Bartlett [6], the size of the weights is more important than the number of weights in determining a generalization. So, adjusting the weight size may significantly improve generalization performance.

Weight-elimination and weight-decay are two regularisation methods to minimize the generalization error of an ANN. Both are achieved by adding a penalty term to the cost function. Weight-elimination eliminates the small weights by forcing them to equal zero, while weight-decay penalizes large weights by causing the weights to converge to smaller absolute values. In a linear model, this form of weight decay is equivalent to ridge regression, which can be used to optimise the performance of our RBF models (Section 2.2). Using these two approaches, the cost function should be rewritten as following:

$$E(W) = \Sigma (E_0(W))^2 + \text{penalty} \tag{2-5}$$

Here, the penalty term for weight decay is:

$$\text{penalty} = \lambda * \Sigma_{ij} (w_{ij}^2) \tag{2-6}$$

where λ is the weight-decay constant and w_{ij} refers to the individual weights within the network.

Similarly the term for weight-elimination is:

$$\text{penalty} = \lambda \sum \frac{\frac{w_{ij}^2}{w_0^2}}{1 + \frac{w_{ij}^2}{w_0^2}} \tag{2-7}$$

where λ is the weight-decay constant, w_{ij} refers to the individual weights within the network and w_0 is the weight-decay scaling parameter.

When comparing the two penalty terms, it can be noticed that weight-decay tends to shrink the large weights more than the small ones, while weight-elimination tends to shrink the small weights more.

The decay constant λ plays an important role here. It indicates how strongly the weights are adjusted. One common approach to determine this constant is to try different amounts of decay for training the network and then choose the decay constant that minimizes the estimated generalization error. In 1991, Weigend *et al.* [78] proposed to update the decay constant iteratively during training. They initialised the weight decay to zero and then gradually increased its value until the generalization error became excessive.

A subtler optimisation approach using weight decay is to use different decay constants for each type of weights in the network. In other words, apply different decay constants for input-to-hidden, hidden-to-hidden, and hidden-to-output weights. This method may generalize better, but it often requires significant computation and is thus inappropriate for our problem.

2.1.4 Parameters to specify a MLP network

In the following discussion, we will demonstrate in detail how an MLP network with weight-elimination works to see what parameters are needed to specify it. We take a

three-layer MLP as an example and assume that the transfer function is the hyperbolic tangent.

During training, the output of the *n*-th layer is:

$$A_n = \tanh\left(\Sigma W_n[i]A_{n-1}[i] + b_n[i]\right),\ n \geq 1 \qquad (2\text{-}8)$$

where A_0 represents the value of the input layer nodes, A_1 represents the output of the hidden layer nodes and A_2 represents the output of the output layer nodes.

The error at the output layer can be calculated by (2-9):

$$E = \tfrac{1}{2} * \Sigma_{p \in data} \Sigma_{j \in outputs} (T_{jp} - A_{2jp})^2 \qquad (2\text{-}9)$$

where T_{jp} is the expected output at the *j*-th output layer node and A_{2jp} is the actual output at this node.

Then, the error obtained at the output layer is back-propagated within the network. The weights are adjusted as below:

$$W = W + dW \qquad (2\text{-}10)$$

where dW is defined as:

$$dW_n = m * dW_n + (1-m) * lr * D_n * A_{n-1}^T - 2\lambda \left[\frac{w_0^2 W_{n_{ij}} \left(1 + \frac{W_{n_{ij}}^2}{w_0^2} \right) - W_{n_{ij}}^3}{w_0^4 \left(1 + \frac{W_{n_{ij}}^2}{w_0^2} \right)^2} \right] \quad (2\text{-}11)$$

Here, D_2 is proportional to the rate of change of the network output A_2 with respect to the error E, and, D_1 is proportional to the rate of change of the hidden layer A_1 with respect to D_2. D_1 and D_2 can be calculated by (2-12, 2-13):

$$D_2 = \frac{dA_2}{dE} \bullet E = \left[\frac{d\tanh(E)}{dE} \right] \bullet E \quad (2\text{-}12)$$

$$D_1 = \frac{dA_1}{dD_2} \bullet W_2 \bullet D_2 = \left[\frac{d\tanh(D_2)}{dD_2} \right] \bullet W_2 \bullet D_2 \quad (2\text{-}13)$$

The bias vector can be adjusted in a similar way:

$$B = B + dB \quad (2\text{-}14)$$

$$\text{Where } dB = m * dB_n + 1 - 2\lambda \left[\frac{w_0^2 B_{n_{ij}} \left(1 + \frac{B_{n_{ij}}^2}{w_0^2} \right) - B_{n_{ij}}^3}{w_0^4 \left(1 + \frac{B_{n_{ij}}^2}{w_0^2} \right)^2} \right] \quad (2\text{-}15)$$

Note here m is a momentum term used to escape a small local minimum in the error surface. lr is the learning rate

To summarize, the following parameters are needed to specify a 3-layer MLP network with weight-elimination:

- Weights from input-to-hidden W_1 and hidden-to-output W_2;
- Bias term b;
- Momentum term m
- Learning rate lr
- Weight decay constant λ

2.2 Radial Basis Function networks

2.2.1 Design philosophy of RBF networks as a classifier

In 1985, the Radial Basis Function method for multivariable interpolation was first introduced by M.J.D.Powell[40]. Studies have shown that this method can generate a unique solution for any choice of data interpolation [40]. In 1988, Broomhead and Lowe interpreted radial basis functions as a neural network model and applied it in a pattern classification problem. This was the first RBF neural network model in the world, and was later inspired by observations that biological neurons tune the response locally to stimuli in several parts of the nervous systems [30].

If we view pattern classification as a curve-fitting (approximation) problem in high-dimensional space, the procedure of the classification can be considered as below:[40]

- Training/learning phase: Construct a surface in a multidimensional space which best fits the training set.
- Test/generalization phase: Interpolate the test pattern properly in the multidimensional surface obtained in the training/learning phase.

The design of RBF networks as an ANN is based on the above statements and the fact that a complex pattern classification problem cast in a high-dimensional space is more likely to be linearly separable than one in a low-dimensional space (Cover's theorem) and the approximation will be more accurate [40]. A typical RBF network can be described as a

3-layer ANN (Figure 2-4) with one input layer, one hidden layer and one output layer. The input layer feeds the patterns into the network, and the hidden layer transforms the input patterns into the high dimensional space through a set of "functions" (i.e., *radial-basis functions*) that constitute an arbitrary "basis" for the input patterns. The output layer, which acts as a linear discriminant, then calculates the weighted sum of the radial basis functions, generating the response of the whole network to the input pattern.

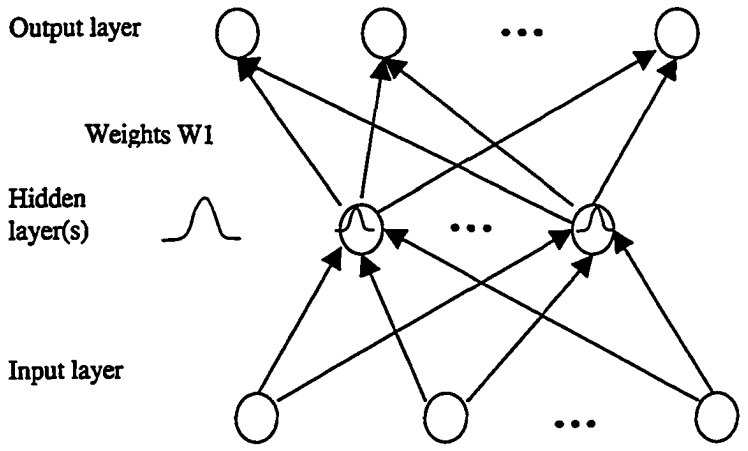

Figure 2-3The structure of an RBF network

Specificially, the hidden layer of an RBF has the following form:

$$h(\boldsymbol{x}) = \Phi((\boldsymbol{x}-\boldsymbol{c})^{T}\mathrm{R}^{-1}(\boldsymbol{x}-\boldsymbol{c})) \qquad (2\text{-}16)$$

where Φ is a set of radial-basis functions, the known vector $\boldsymbol{c}$ is an n-dimensional vector consisting of data points called the "centres" of the radial-basis functions, and $\boldsymbol{x}$ is the independent variable (input of the network). R is the metric.

Radial basis functions are a special class of functions that respond monotonically to the distance from a central point. The centre, c, and the distance scale σ are parameters of the function. There is a large class of radial-basis functions used in RBF networks. The most common ones include Gaussian functions and Multiquadrics functions.

Gaussian functions monotonically decrease with distance from the centre.

$$h(x) = \exp\left(-\frac{(x-c)^2}{\sigma^2}\right) \tag{2-17}$$

Multiquadrics functions monotonically increase with distance from the centre. This is a popular tool for scattered data fitting:

$$h(x) = \frac{\sqrt{\sigma^2 + (x-c)^2}}{\sigma} \tag{2-18}$$

The output layer of the RBF network accepts the results $h(x)$ from the radial basis functions at the hidden layer and combines them linearly by assigning corresponding weights w. As this procedure is linear, an RBF network is usually considered as a linear system:

$$f(x) = \sum_{j=1}^{m} w_j h_j(x) \tag{2-19}$$

Most Artificial Neural Networks, such as MLP networks with back-propagation algorithms, have many local minima in their error surface. RBF networks have a wonderful property in their capability [64]. Chen and Chen [12] addressed the problem of what kinds of radial functions have the property of universal approximation and found that a necessary and sufficient condition is that the radial basis function not be even polynomial.

2.2.2 Parameters to specify an RBF network

A naive RBF network is completely specified by the following parameters:

- The number n of radial basis functions, i.e., the number of nodes in the hidden layer;
- The centres c_i and the widths of each centre σ_i (i = 1..n);
- The weights w_{ij} between the hidden layer nodes and the output layer nodes.

Note here that RBF networks have a relatively simple parameter space. This property usually means that less training effort is necessary. The mapping from the hidden layer to the output layer is simply a linear mapping so the weight w_{ij} can be solved by linear optimisation technology. The only other concern is the computation of the radial basis functions of the hidden layer. In practice, there have been numerous fast algorithms aiming to solve this problem.

2.2.3 A brief overview of training algorithms for RBF networks

Unlike MLP, which often uses back-propagation algorithms for learning, there are many learning algorithm choices for RBF networks. Learning algorithms for RBF networks can consist of several different strategies originating from different fields. Some strategies can apply to the whole parameter space while others only apply to some of the parameters. Although the learning algorithms vary a lot, they are usually categorized as follows according to different category criteria:

- Static or dynamic learning: Since the number n of radial functions is critical to the performance of the network, and other parameters (centres, widths, weights) all depend on it, the manner in which it is determined is used as a criterion to distinguish between static or dynamic learning algorithms. The learning algorithm for RBF is called static if it starts with a fixed number n of radial functions or dynamic if it adds or deletes some basis functions step by step until an optimal value is achieved. Both learning methods will be discussed in this thesis.

- Online or offline learning: RBF learning algorithms can also be categorized as either online or offline learning. Online learning means that training data change during the learning process, with new data fed to the network set by set [29]. Offline learning means that all training data are completely available at the beginning of the learning process. Online learning algorithms are usually employed in recurrent RBF networks, which differ from the regular RBF structure in that each hidden layer neuron also has one feedback connection. Offline algorithms are not applicable to our problem, since we already have the whole database available before training.

- One-, two- and three- phase learning: Friedhelm Schwenker etc. [30] categorize the RBF Learning algorithms into one-, two and three-phase learning schemes. In practice, the two-phase learning procedure is most common: First, the hidden layer is determined by calculating the centres and the widths of the basis functions. Second, the weights between the hidden and output layers are calculated. The learning methods used in this thesis will be two-phased.

Actually, among all the parameters, methods of estimating the weights are consistent, although methods of determining centres/widths vary widely. The weights are usually estimated by computing the coefficient of the pseudo-inverse matrix or, alternatively, by performing the error gradient descent.

Since the output of an RBF network is linear (in the weights), a linear optimisation method that aims to minimise the cost function computed on the sample set can be used to calculate weights. Weights Linear Optimisation (WLO) tunes the weights between the output layer and the hidden layer by computing a pseudo-inverse matrix that has the following form:

$$\mathbf{W} = \mathbf{G}^{+} d \tag{2-20}$$

where $\mathbf{G}^{+} = (\mathbf{G}^{T}\mathbf{G})^{-1}\mathbf{G}^{T}$ is the pseudo-inverse matrix, G is the output of the hidden layer (radial basis functions) and d is the expected output.

Gradient descent is another popular method of computing the weights. Since this method also applies to other parameters of RBF networks, it will be discussed later.

Below, our discussion of some typical RBF learning strategies will focus on how to calculate the centres/widths.

K-mean is a typical statistical clustering algorithm that determines the centres/widths associated to the basis functions. The training data are clustered into K subsets based on their similarities/distances defined by certain mathematical criteria. Each centre represents a cluster of nearby data points. During the clustering process, the centres/widths are updated iteratively among the recent data points of each cluster. K-mean is a static learning algorithm. It performs well when the correct value of K is chosen or known. Unfortunately, in practice, the value of K must be set by some heuristic process and is hard to determine. This weakness impairs the performance of the K-mean method.

The Gradient Descent method is used to improve the performance of RBF networks. This is achieved by iteratively updating the means and standard deviations of the radial basis functions. This can be applied to part or all the parameter space of RBFs after these parameters have been initialised and the learning rule can be expressed as: $A \leftarrow A - \eta\frac{\partial E}{\partial A}$,

where η is the learning rate. E is the error evaluated and A indicates a generic parameter, which could be a centre, width or weight.

The main disadvantage of gradient descent is the slow convergence speed and unstable performance due to reliance on the choice of the initial parameters (centres). Although it may not be a good candidate for our problem due to its slow speed, one of its advanced versions, the Expectation Maximization (EM) algorithm [71], is still worth introducing here, as this is currently one of the most efficient learning methods for RBFs.

The Expectation Maximization (EM) algorithm [20] is a data clustering method that can be seen as a generalized version of K-mean clustering. In the K-mean algorithm, an input data pattern is assigned to one cluster only. In the EM algorithm, the membership of each input pattern can be distributed over multiple clusters. The EM algorithm computes a maximum likelihood solution using Gradient Descent to model the input-output distribution. The input density is modelled as a mixture of components by Gaussian distributions. The estimated parameters of the mixture density are then transplanted into the RBF network, after which supervised learning of the network takes place [71].

Another advanced learning algorithm, the Orthogonal Least Squares (OLS) method, originated from linear regression technology, and is a quick learning algorithm. In 1991, Chen *et al.* [11] proposed an algorithm for forward stepwise regression for training RBF networks. Instead of adjusting the RBF centres in continuous space like most other algorithms do, OLS selects discrete subsets that usually provide good training error from

a large set of candidate centres via a modified Gram-Schmidt orthonormalisation. This way, the problem turns into a linear problem. As we know, linear models are simpler to analyse mathematically than nonlinear models, since there are ways to derive and solve a set of equations for linear model optimization. Nonlinear models, such as MLPs, require iterative numerical procedures for their optimization, so computation is usually more expensive than linear models. Due to its special mechanism, OLS is supposed to be a fast learning algorithm.

There are many revised versions of OLS aimed at improvements in centre selection and the over-fitting problem. One famous version is the Regularized Orthogonal Least Squares (ROLS) method proposed in 1995 by Chen *et al.* [11] and later revised by Orr [62] by combining ridge regression (also known as weight decay in ANN) with forward subset selection for RBF training.

2.2.4 Regularized Orthogonal Least Squares (ROLS) method

Since OLS is based on linear regression, this section begins with a brief introduction of linear regression.

In general, a linear model can be expressed in the following form:

$$y = f(x) = \sum_{j=1}^{m} w_j h_j(x) \qquad (2\text{-}21)$$

where the regressors $h(\cdot)$ are fixed functions of input x, and only the coefficients w are unknown. The goal of linear regression is to find the best fit on the training set $\{(x_i, y_i)\}_1^p$, i.e., to solve the best w for the system of equations $\mathbf{y} = \mathbf{H}\,\mathbf{w} + \mathbf{e}$ which minimizes the defined error:

$$E = \mathbf{e}^T \mathbf{e} \tag{2-22}$$

Here, the design matrix $\mathbf{H}$ ($p \times m$) are the responses of the m regressors to the p training patterns, and output $y = [y_1, y_2, \ldots y_p]^T$ is the p-dimensional vector responses to p patterns (as in our problem, y should be a scalar). The vector e contains p unknown errors between the expected outputs and the true values. The solution of the above proposition is:

$$w = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T y \tag{2-23}$$

When comparing an RBF network and a linear regression model, each centre (c_j, r_j) of an RBF can be considered as corresponding to a regressor $h_j(x)$ in a linear regression model, and the weights from the hidden units to the output unit as corresponding to the coefficients w. A regressor is determined by a point (centre) in the input space and a scale factor (radii). The radii can be fixed since studies show that such networks are still universal approximations [70]. From this point of view, selecting centres of RBFs is the same as selecting regressors in a linear regression model.

Suppose at the beginning we have a set of centre candidates, we are going to choose a rational number of centres from the candidates set. Usually we take all training data as the candidate centres. Then the design matrix is $p \times p$, hereafter noted as F (called the full design matrix). The centre selection strategy must avoid the over-fitting problem. If too many centres are chosen from the candidates, this regression may overfit, since the training set is reproduced (memorized) exactly by the network.

Generally speaking, there are two ways to avoid the overfitting problem. Tikhonov's regularisation theory solves the poor generalization problem through minimizing the Tikhonov function[40]. The Tikhonov function involves a penalty term besides the standard error term [40]. The standard error term measures the standard error (distance) between the desired (target) pattern and the actual pattern for training examples. For the penalty, one may embed prior information to make the regularisation smoother. The regularisation (ridge regression in statistics) minimizes the following energy:

$$E = \mathbf{e}^T\mathbf{e} + \lambda \mathbf{w}^T\mathbf{w} \tag{2-24}$$

From the above equation we get:

$$w = (\mathbf{F}^T\mathbf{F} + \lambda \mathbf{I}_p)^{-1}\mathbf{F}^T y \tag{2-25}$$

where $\mathbf{I}_p$ is a p×p identity matrix and λ is the regularisation parameter that is chosen *a priori* or estimated from the data.

Another way to avoid the overfitting problem is to choose only part of the candidates as the network centres. Subset selection methods of regression analysis are supposed to be suitable for solving this problem since by these methods we can choose those subsets that best explain the variation in the dependant variables (i.e., mortality in this thesis). OLS is a quick forward selection method which selects the centres using a Gram-Schmidt orthogonalisation process. In this process, each new column added to the design matrix of the growing subset is orthogonal to all previous columns. This simplifies the equation for the change in sum-squared-error and results in a more efficient algorithm. Below we will show how regularized OLS solves the linear regression problem.

Any matrix can be factored into the product of a matrix with orthogonal columns and a matrix which is upper triangular. In particular, the design matrix H_m obtained at the m-th step, can be factored into

$$\mathbf{H}_m = \tilde{\mathbf{H}}_m \mathbf{U}_m \tag{2-26}$$

where $\tilde{\mathbf{H}}_m$ is a $p \times m$ matrix that has orthogonal columns and $\mathbf{U}_m$ is an $m \times m$ matrix that is upper triangular.

Then the regression equation can be rewritten as follows:

$$\mathbf{y} = \tilde{\mathbf{H}}_m \tilde{\mathbf{w}}_m + \mathbf{e}_m \tag{2-27}$$

where

$$\tilde{\mathbf{w}}_m = \mathbf{U}_m \mathbf{w}_m \tag{2-28}$$

At the mth step, $\tilde{H}_{m-1}$ is augmented by a new column $\tilde{f}_i$ which is orthogonal to the m-1 existing and already orthogonal columns,

$$\tilde{H}_m = \left[\tilde{H}_{m-1} \quad \tilde{f}_i \right] \tag{2-29}$$

where

$$\tilde{f}_i = f_i - \sum_{j=1}^{m-1} \frac{f_i^T \tilde{h}_j}{\tilde{h}_j^T \tilde{h}_j} \tilde{h}_j \tag{2-30}$$

and f_i is selected from the columns of the full design matrix F.

In the case of a regularized network, "orthogonalisation is only possible if the roughness penalty term depends on the orthogonalised weights $\tilde{w}_m$ and not on the ordinary weights w_m" [11]. Then, the minimized energy is

$$\tilde{E}_m^{(i)} = e_m^T e_m + \lambda \tilde{w}_m^T \tilde{w}_m = y^T \tilde{P}_m y \tag{2-31}$$

where

$$\begin{aligned} \tilde{P}_m &= I_p - \tilde{H}_m (\tilde{H}_m^T \tilde{H}_m + \lambda I_m)^{-1} \tilde{H}_m^T \\ &= I_p - \sum_{j=1}^{m-1} \frac{\tilde{h}_j \tilde{h}_j^T}{\lambda + \tilde{h}_j^T \tilde{h}_j} - \frac{\tilde{f}_i \tilde{f}_i^T}{\lambda + \tilde{f}_i^T \tilde{f}_i} \\ &= P_{m-1} - \frac{\tilde{f}_i \tilde{f}_i^T}{\lambda + \tilde{f}_i^T \tilde{f}_i} \end{aligned} \tag{2-32}$$

The $\tilde{f}_i$ which maximizes

$$\tilde{E}_{m-1} - \tilde{E}_m^{(i)} = \frac{(\mathbf{y}^T \tilde{\mathbf{f}}_i)^2}{\lambda + \tilde{\mathbf{f}}_i^T \tilde{\mathbf{f}}_i} \tag{2-33}$$

is selected and becomes the last column of $\tilde{\mathbf{H}}_m$ noted as $\tilde{\mathbf{h}}_m$.

Orr proved that the computational cost (number of floating point operations) required to select one centre from M candidates with p patterns in the training set is proportional to Mp with orthogonalisation. Without orthogonalisation, the cost is roughly proportional to Mp^2. If the whole input training points are used as centre candidatess, then $M = p$ and the corresponding costs are p^2 and p^3 respectively [62]. Obviously, the computation complexity is decreased with orthogonalisation.

To summarize, a forward selection ROLS method picks centres among all the regressors of the initial model to determine how each regressor will contribute to the modelling of the desired system. The regressor that most decreases the regularized error with the regressors already selected will be added to the subset. The selection process is repeated until the maximum number of centres is reached or if adding new centres does not make any difference to the regularized error ratio according to a user-defined criterion.

2.3 Cross-validation for error estimation

For most artificial intelligence systems, a simple way to stop a learning process is to set a fixed threshold on the calculated error. Unfortunately, such a fixed threshold is liable to result in an overfit. Cross validation is a technique for calculating generalization errors by

repeatedly re-sampling the training data. In k-fold cross-validation, the whole training data set is partitioned into k subsets of equal size. The system is trained k times, each time with k-1 subsets for training and the omitted 1 subset for validating the trained system by computing the error on it. The general error is calculated on the errors attained in the k time trainings. If only one sample is left for validation, this is called "leave-one-out" (LOO) cross-validation. For non-linear regression problems with long training times, e.g., MLP networks with a back-propagation algorithm, cross-validation is usually too expensive to compute. However, for linear regressions that can be solved using parsimonious linear optimisation, such as in RBF networks, cross-validation can be very helpful.

2.4 Principal Component Analysis (PCA) for dimension reduction

Many practical applications of RBF networks use the Principal Component Analysis (PCA) approach to reduce the input variable dimension for the RBF layer. It is actually used with some MLP network as well but much more often seen in RBF networks, so we put it in this chapter for discussion. Via this approach, data represented by a set of correlated variables are transformed into a set of uncorrelated variables by a linear transformation. The new variables are linear combinations of the original variables and are ordered by reduced importance. As a result, representing the data without the last several variables of the new combined variables may not result in the loss of significant information.

The main use of PCA is to reduce the dimensionality of a dataset while retaining as much relevant information as possible [41]. However, in practice, while summarizing the important parts, the noise is also simultaneously filtered [52]. This is why in some applications, the new compact representation of the dataset resulting from PCA performs better than the original one.

PCA is based on the statistical representation of the multivariate vector. Suppose we have a random vector population x of n dimensions which is noted as:

$$\mathbf{X} = (x_1, x_2, \ldots, x_n)^T \tag{2-34}$$

Then, the mean and the covariance matrix of that population are:

$$\mu_x = E\{\mathbf{X}\} \tag{2-35}$$

$$C_x = E\{(x - \mu_x)(x - \mu_x)^T\} \tag{2-36}$$

Now consider the components of C_x. From a statistical perspective, if two variables x_i and x_j are uncorrelated, their covariance is zero, i.e., the corresponding component $c_{ij} = c_{ji} = 0$. If the variables are highly related, the covariance component will be large.

Given C_x, we can compute the value of the eigenvalues λ_i and eigenvectors e_i of the covariance matrix by the following relationship:

$$| C_x - \lambda I | = 0 \quad (2\text{-}37)$$

$$C_x e_i = \lambda_i e_i \quad (2\text{-}38)$$

where the λ_i are assumed to be distinct, I is the identity matrix having the same order as C_x and $|.|$ denotes the determinant of the matrix.

Let A be a matrix consisting of eigenvectors of the covariance matrix as the row vectors e_i. By transforming a data vector X, we get:

$$Y=A(X- \mu_x) \quad (2\text{-}39)$$

The components of Y can be seen as the coordinates in the orthogonal base. Now the original data vector X can be reconstructed as:

$$X = A^T Y + \mu_x \quad (2\text{-}40)$$

As mentioned before, the last several components have little significance for the vector. So, we may rewrite the above expression in terms of only a few basis vectors of the orthogonal basis. If we denote the matrix having the K first eigenvectors as rows by A_k, we can create a similar transformation as seen above:

$$Y= A_k (X- \mu_x) \quad (2\text{-}41)$$

$$X = A_k{}^T Y + \mu_x \quad (2\text{-}42)$$

Note that the original data vector has now been projected from the original n dimension space to the new k dimension space, and the new vector is a linear combination of the basis vectors. In theory, by picking the eigenvectors having the largest eigenvalues, we lose as little information as possible in the mean square sense. In actual application, the number of eigenvectors and their respective eigenvalues can be tried out experimentally to meet the particular requirement of the relevant problem.

PCA provides a convenient way to control the trade-off between retaining more information and reducing the computation complexity, so it has been frequently used as a dimensionality reduction method for RBF networks. A number of studies have shown that PCA performs well when combined in RBF networks [1]. In a combined RBF+PCA system, a typical procedure is as follows: First, a standard PCA is performed to reduce the dimensionality of the original data; then, an RBF network accomplishes the system task with the new pattern vectors obtained from the PCA as its input data.

2.5 A brief comparison of the training algorithms of MLP network and the RBF network

In general, there are two significant differences between the RBF and MLP networks. First, RBF networks can use unsupervised clustering methods to cluster the data without presupposed class labels based on their similarity. This usually increases the approximation quality. Unfortunately, the proper number of clusters is not easy to determine in most RBF algorithms, and even for algorithms that can automatically

compute the cluster, it is restricted by the limited initialization techniques available. As a result, the predictive ability of the network is weakened.

Second, RBF networks can employ some quick training algorithms other than the traditional gradient descent method, which requires a large amount of training iterations. The ROLS algorithm used in this thesis is one successful example of these rapid training algorithms. Given good parameter initialization, RBF networks may need less training time than do MLP networks.

As for the RBF networks, there are many training algorithms and they have different characteristics. The K-mean clustering algorithm and the OLS algorithm are two of them. As mentioned earlier, the nodes number of hidden layer is hard to determine for K-mean clustering algorithm in real applications. If we choose too few clusters, the network will not be able to efficiently separate the data; if there are too many, over-fitting will happen, resulting in poor generalization ability. In actual application, a naïve method to determine the K value is to first pre-define a range of K values heuristically, and then try every single value of K within this range to obtain the best performance.

Another weakness of the K-mean method is its sensitivity to parameter initialization. The data distribution which determines the centres/widths initialization can affect the clustering results, and thus the final outputs of the whole network. Usually, a simple but very common way to initialize centres is to randomly create K centre points in the data

space. Unfortunately, with this method, the output results fluctuate significantly depending on the different values of the initialized centres/widths.

The forward selection OLS method is an advanced approach to calculate radial basis functions. Unlike other simple methods such as the K-mean method, forward selection OLS does not require a pre-defined number of clusters. Instead, the proper cluster number is determined automatically by selecting the most suitable data point every step until the optimal result is achieved. Another advantage of forward selection OLS over K-mean is its robust results regardless of the data distribution in the dataset.

Chapter 3 Methodology

This chapter outlines the procedures to set up and analyse the MLP/RBF mortality prediction models. The methodology consists of the following parts:

- Data pre-processing of a large clinical database;
- Measurement of performance of mortality prediction systems;
- Implementation and validation of MLP network models with back-propagation algorithm and weight-elimination method;
- Implementation and validation of RBF network models with K-mean algorithm, or, Regularized Orthogonal Least Squares (ROLS) algorithm.

All models can come with a PCA process before the data is fed to the network. The RBF models also use the cross-validation technique to compute the errors. A detailed discussion of the PCA process and cross-validation technique can be found in section 2.4 and 2.3. We do not repeat them in this chapter.

3.1 Data pre-processing

Data quality is critical to the efficiency of a medical decision support system. Therefore, it is necessary to pre-process data to improve the data quality and make the extracted features more reliable. The original database used in this thesis contains 1491 complete cases collected over years from the ICU at the Dr. E.Chalmers Hospital. All cases in this

database have no missing variables and are of patients over 12 years old. Since surgical patients and medical patients exhibit different behaviours (see section 1.3.4), we divide the database into two major parts: 883 surgical cases (patients who were admitted to the ICU post-operative) and 608 medical cases (non-postoperative). Previous research by MIRG discovered that the non-postoperative data set is much harder to analyze due to its non-homogeneity, this study will mainly focus on post-operative patients (hereafter noted as Post-op). In addition, for the purpose of testing the designed system on more datasets, 3 groups of datasets extracted from the Post-op database are considered in this thesis. Each group consists of the same training set but different test set. Non-postoperative data set will also be test for comparison purpose. Table 3-1 summarizes the mortality rate distribution of each data set.

Database	No. of cases	No. of non-survival	Mortality Rate
DECH* ICU (whole)	1491	104	6.97%
L post-operative	883	28	3.17%
L training set	589	18	3.06%
L test set A	294	9	3.06%
test set B	294	9	3.06%
test set C	294	9	3.06%
L non-postoperative	608	76	12.50%
L training set	406	51	12.56%
L test set D	202	25	12.38%

Table 3-1 Statistics of the DECH ICU database

The original DECH ICU database contains a large amount of medical and administrative variables. To make it more consistent and compact, MIRG members have pruned the database through consultation with the medical staff. The dataset used in this study contains 52 variables including some key risk factors along with APACHE II variables, of which 51 variables are used as system inputs and 1 as output. The final variables selected in this study are listed in Table 3-2.

In order to become available for neural networks, the medical data should be pre-processed before they are sent to the networks. The following pre-processing has been done with the data in this study:

Step 1: Normalize variables of the database by some criteria;
Step 2: Shuffle the database randomly and divide it into training data and test data by the ratio of 2:1;
Step 3: Copy more positive (non-survivor) cases into training data, replacing the negative data to artificially increase the mortality rate of the training dataset by some degree (50% in this thesis).

These three steps are described in detail below:

Variable	Normalization method
Demographics and Administrative Information Assigned chronic health points in APACHE II scoring Emergency surgery prior to ICU admission Surgery prior to admission Patient gender	Method 1
Position in data sequence Patient age (years)	Method 2
APACHE II (Admission Information) Rectal temperature (°C) Mean arterial pressure (mmHg) Heart rate Respiratory rate Fraction of inspired oxygen Partial pressure of oxygen in the blood Arterial pH Serum sodium (mmol/l) Serum potassium (mmol/l) Serum creatinine (μmol/l) Hematocrit White blood cell count (total/mm^3 in 1000s) Glasgow Coma Score	
Admission Source Emergency Room 4SW 4E 4W 4NE 3W 4NW 3SW Coronary Care Unit 3E Admission from another location	Method 1
Admission Diagnosis #1 Postoperative Acute hypercapnic respiratory failure Trauma Drug overdose Ketoacidosis Sudden cession of heart or lungs Other diagnosis #1 **Admission Diagnosis #2** Carotid endarectomy Nothing filled in Abdominal aortic aneurysm repair Motor vehicle accident Lobectomy Aortobifemoral bypass Pneumonia Acute pulmonary edema Other diagnosis #2 **Admission Diagnosis #3** Nothing filled in Lung cancer Postoperative Ischemic foot Other diagnosis #3	Method 1
Output: Death	

Table 3-2 Summary of variables used in the DECH ICU database

3.1.1 Normalize variables of the database

The 52 variables used in this study include two types of data: text variables (admission diagnosis and admission location) and physiological variables, whose possible value range can be very wide. In this thesis, the variables are normalized in the same way the MIRG lab has done [33]. To be more specific, the following methods are included (see Table 3-2):

Method 1 (for all text variables): Variables are categorized into Boolean values so that only 1 (true) or –1 (false) are used;
Method 2 (for physiological variables): Variable value minus the expected 'normal' value and is then divided by three standard deviations. The expected 'normal' values of each variable were determined by MIRG members in consultation with experienced physicians at DECH.

3.1.2 Divide the whole dataset into training data and test data

After normalization, the database is shuffled randomly and divided into a training dataset and a test dataset so that two-thirds of the database is in the training dataset and one-third is in the test dataset. Out of the whole post-operative database, the training dataset contains 589 cases and the test set contains 294 cases.

3.1.3 Artificially increase the mortality rate of the training dataset

The actual mortality rate in the databases used in this study is as low as 3.17% (28 non-survivors out of 883 patients in the whole Post-op dataset). Such a low occurrence can harm the learning ability of a prediction system and make the identification of low occurrence patterns almost impossible. To deal with highly skewed data, studies suggest resizing the training set [81] so that more sample cases representing the characteristics of low occurrence patterns are included. There are two methods of artificially altering the database, over-sampling minority class examples and down-sizing the majority class [81]. In mortality prediction, they mean:

• Over-sampling minority class examples: Randomly copy non-survivor cases into the database until an expected percentage of non-survivors is reached;
• Down-sizing the majority class: Randomly delete survivor cases from the database until an expected percentage of non-survivors is reached;

In this thesis, the first method is selected, i.e., a certain number of positive cases are randomly selected and copied into the training dataset to replace negative cases until a desired artificial mortality rate (50% as in this thesis) for the training dataset is achieved [33]. This way, the altered database does not become too large and the critical valuable non-survival cases are kept.

3.2 Measures of performance

Unlike other pattern classification problems, using errors alone between the expected values and approximated predicted values is not a good way to evaluate the performance of a medical outcome prediction system. To evaluate the performance of a mortality prediction system, measurements combining sensitivity, specificity, correct classification rate and constant predictor are commonly used [57]. Among these measures, sensitivity, specificity and correct classification rate can be calculated using a contingency table.

3.2.1 Contigency table

Correct classification rate (CCR), sensitivity and specificity can be expressed in one table: The contingency table.

		Correct output		
		Not present ↓	Present ↓	
Model output:	Not present →	True negative (TN)	False negative (FN)	←*TN+FN*
	Present →	False positive (FP)	True positive (TP)	←*FP+TP*
		↑ *TN+FP*	↑ *FN+TP*	

Table 3-3 Contingency (True) table

As we mentioned earlier in section 1.3, sensitivity is the percentage of the positive cases (deceased cases in this research) that are correctly classified. Specificity is the percentage of the negative cases that are correctly classified. Correct classification rate is the percentage of cases that are correctly classified. From Table 3-3, we can easily calculate the CCR, sensitivity and specificity using the following equations:

$$\text{CCR} = \frac{TP + TN}{TN + FN + FP + TP} \tag{3-1}$$

$$\text{Sensitivity} = \frac{TP}{FN + TP} \tag{3-2}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{3-3}$$

3.2.2 Log-sensitivity index

For mortality prediction systems, the misclassification of a patient may result in various medical costs. Unlike in ordinary pattern classification problems, the classification rate is not necessarily the best measure for the mortality prediction problem. In practice, sensitivity and specificity, together with classification rate, are all used to measure a medical support system. A satisfactory trade-off is to balance sensitivity and specificity, slightly favouring sensitivity.

For the sake of simplicity, the MIRG developed one single index to measure the performance of the medical decision aid system. This index is called the log-sensitivity index and has the following form:

Log-sensitivity = -Sensitivity^n*log (1-Sensitivity*Specificity) (3-4)

Usually we choose n=1. This index can be used as a stopping criterion to select an optimal model when training with different model architectures. The higher the index is, the better the model is supposed to be.

3.2.3 Constant predictor

The constant predictor is usually used as a reference to compare the performance of the prediction system with some extreme case which classifies all cases as the category with the highest a priori probability. When applied to mortality prediction, a constant predictor classifies all patients in ICU as survival. This simple predictor can reach a high classification rate because survival cases always occur frequently in the ICU database. Unfortunately, with this method, none of the death cases are correctly classified. This means that the specificity of a constant predictor is 100% but the sensitivity is 0%. So, a constant predictor is usually used as a comparative standard for a designed system since it does not alone have any further value except for its statistical predictions. An ideal mortality prediction is supposed to have a high sensitivity and a reasonable classification rate higher than the rate achieved by the constant predictor.

3.3 Implementing the MLP network system

3.3.1 Network structure

Much research has shown that multi-layer networks outperform single-layer networks since they can develop models with nonlinear characteristics that are suitable for complicated problems [33]. On the other hand, researchers also agree that for most applications, a simple 3-layer network with 1 hidden layer is sufficient. Moreover, a 3-layer network has a relatively low complexity due to its simple structure, so it is usually a first choice for many applications. Previous research by MIRG has shown that a 3-layer MLP with weight-elimination performs better than other MLP architectures when used to predict medical outcomes [32;33]. Therefore, an MLP network with such architecture is chosen for the problem under study.

Based on previous experience in MIRG, the transfer function used in the network is the hyperbolic tangent function. This type of function has an output range of -1 to 1 with inputs scaled to have zero mean and unit variance. Networks that use this transfer function are usually expected to learn more quickly. This function is chosen for these experiments in the hope that it may help to better distinguish between survivors and non-survivors.

To compare the efficiencies of structures with and without the weight-elimination cost function, both structures are implemented and tested in this thesis. The training algorithm used for these models is the typical MLP learning algorithm: the back-propagation learning algorithm.

3.3.2 Automation of parameter computing

As mentioned earlier, there are many stopping criteria for training a medical prediction system, depending on the different measurement criteria needed. As a result of this, the training of the medical prediction system is also unique. Based on a decade of experience in research and experimentation in the medical domain, MIRG has implemented an automation program to set up and train/test MLP models. This program has been successfully validated with a variety of medical databases [26;31;33]. It was implemented based on a standard simple MLP network which is trained by the back-propagation algorithm and several useful functions were added to it to help obtain the best model automatically. Below are features of this automation program [25]:

- Standard back-propagation algorithm with/without the weight-elimination cost function;
- Two- or three-layer architectures;
- Adjustable number of nodes in the hidden layer;
- Multiple stopping criteria: Highest test set classification rate, lowest test set average squared error, exact epoch stop, highest test set log-sensitivity index;
- Selection of the initial values and ranges of the network parameters;
- Automated or manual program to operate the network with the selected parameters.

In this automation program, an MLP network is specified by the following parameters: learning rate (*lr*) and its adaptive parameters (*lr_inc*, *lr_dec*), momentum (*m*), weight-elimination scale factor (*lambda*) and its adaptive parameters (*lambda_inc*, *lambda_dec*), weight-decay constant (w_0) and error ratio (*error_ratio*). A set of experiential value

ranges of those parameters is defined before iteratively computing the optimal parameters. This automation program uses a divide-and-conquer method to determine the optimal (defined by different criteria) parameter values of an MLP network [25]. After each epoch of training, the range of parameters is divided into two and the new iteration is executed in the new range to obtain the best parameters among them.

If TSSE is the Training Sum Squared Error from the current epoch and SSE is the error from the previous epoch, then the parameters are adjusted as follows (pseudocode):

If TSSE >SSE*err_ratio

lr ← *lr*lr_dec*;

lambda ←*lambda** *lambda_dec;*

m ← 0;

else if TSSE<SSE*err_ratio

lr ← *lr*lr_inc;*

lambda ← *lambda** *lambda_inc;*

end

3.3.3 Parameter initialization

The weights connecting the input layer with the hidden layer are randomly initialised between -1 and 1. The weights connecting the hidden layer and the output layer are

randomly initialised between -0.1 and 0.1. The bias vectors of every layer are unit vectors.

Previous studies by MIRG have determined some optimal initial parameter value ranges for DECH ICU. Table 3-4 listed are initial values of the parameters used in this thesis.

Parameter	Initial Value
lr	0.00001
lr_inc	1.001
lr_dec	0.995
lamda	0.00001, after 500 epochs: 0.0001
lamda_inc	1
lamda_dec	1
w_0	0.001
momentum	0.2
error_ratio	1.0001

Table 3-4 Initial parameter values (MLP)

3.4 Implementing the RBF network system

Unlike MLP networks, RBF networks can be trained in a variety of ways and their characteristics can be quite different. Both K-mean clustering way and regularisation OLS way are implemented in our system to set up a RBF network. However, since the ROLS method needs less training effort and is more stable, it is used more often as a high performance method in our study while K-mean provides an optimal approach and a comparative reference. In the following sections, the methodologies to implement RBF networks via different approaches are discussed in detail.

3.4.1 A conventional K-mean approach

The conventional K-mean approach to calculate the centres/widths of RBF is an efficient way provided the centre number K is known. This is a typical data clustering technology based on the similarities between data and includes the following steps in training phase [5]:

1) Initialize the centres/widths by randomly selecting K data points;
2) For the whole training dataset, calculate the distance between every data point and every centre;
3) Assign every data point to the cluster whose centre is the nearest to the data point.
4) Recalculate the centres/widths based on the present members of every cluster.
5) Calculate the difference between the present centres and the previous centres, noted as error;
6) Repeat step 2-5 until the centres/widths do not change, i.e., error is smaller than a threshold. These centres/widths are the final centres/widths.
7) Calculate the Radial Basis Functions for each centre, i.e., the output of the hidden layer.
8) Calculate the weight between the hidden layer and the output layer using a pseudo-inverse matrix.

After the centres/widths of the hidden layer and the weights between the hidden layer and the output layer are determined, an epoch of training is finished. Now, new data can be tested in the following steps:

9) Calculate the values of Radial Basis Functions for each centre over the test dataset, i.e. the outputs of the hidden layer.
10) Calculate the output of the output layer by a linear production of the weights and the output of the hidden layer. This is the output of the whole network, i.e., the prediction value of the test data.

3.4.2 Regularized Orthogonal Least Squares (ROLS) approach

Regularized Orthogonal Least Squares method is an advanced RBF learning algorithm. In this thesis, we compute the centres in a forward selection way. With this method, centres are chosen one by one from a candidate set and new centres are selected in a forward way in the effort to reduce the regularized error. Following are the steps of this algorithm [62]:

1) Assign all training data points as centre candidates. Initialize the full design matrix **F** with all the initial candidates. Initialize the design matrix **H** and $\tilde{\mathbf{H}}$ as empty;
2) Select a f_i from the full design matrix **F**, initialize the $\tilde{f}_i$ and $\tilde{h}_i$;
3) Compute all $\tilde{f}_i$ by equation 2-30;
4) Selected f_i which minimise the equation 2-33 and append the $\tilde{f}_i$ to $\tilde{\mathbf{H}}$;
5) Compute the error using cross-validation;

6) If the error exceeds a preset threshold, go to 2) to continue to select centre, otherwise, go to 7);

7) Assign all the columns in **H** to centres.

8) Compute the weights.

9) Compute the outputs of hidden layer over the test dataset.

10) Compute the output of the network over the test dataset.

3.4.3 ROLS approach with a K-mean centre initialization method

As illustrated in Section 2.2.4, the efficiency of ROLS method for computing the RBF centres is affected by the centre candidate initializations because the later centres are selected from this initial candidate set. If the initial candidates are not suitable to represent the whole data space, the final clustering result can be poor. A typical method for initialising the centre candidates is to assign the whole training dataset as the candidates (Section 3.4.2). However, in real application, the whole training dataset is not necessarily good enough to represent the data space. Therefore, in this study, an attempt is made to compute more representative centre candidates by using a K-mean method. Following are the procedure of this K-mean+ ROLS method:

1) Randomly select K data points of the training data set;

2) For the whole training dataset, calculate the distance between every data point and every centre (candidate);

3) Assign every data point to the cluster whose centre (candidate) is the nearest to the data point.

4) Recalculate the centres (candidates) / widths based on the present members of every cluster.

5) Calculate the difference between the present centres (candidates) and the previous centres (candidates), noted as error;

6) Repeat step 2-5 until the centres (candidate) / widths don't change, i.e., error is smaller than a stopping threshold. These centres (candidates) are the initial RBF centre candidates.

7) Assign all the centres acquired at step 6 as centre candidates. Initialize the full design matrix F with all the initial candidates. Initialize the design matrix H and $\hat{H}$ as empty;

8) Compute the RBFs using a ROLS method (See step 2-10 in Section 3.4.2)

Chapter 4 Results and discussion

Based on the previous discussion, a mortality prediction system assembling the MLP and RBF methods has been implemented. The MLP networks include models with and without weight-elimination and the RBF networks include models with and without regularisation. PCA is also implemented in this system to facilitate data analysis. Some other classification tools such as a constant predictor and Bayesian classifier are also implemented in this system as a performance reference. The system is tested on a real medical database originating from the DECH ICU database (Section 3.1): the whole post-operative cases database. All data sets were pre-processed in the way described in Chapter 5.1. This chapter outlines the experimental results of mortality prediction using this system. The chapter consists of the following three parts:

- Experimental results and discussion using MLP models;
- Experimental results and discussion using RBF models;
- Performance evaluation of different models.

For a medical prediction system, the quality we are most concerned with is its prediction efficiency in terms of the correct classification rate, sensitivity and specificity. Therefore, instead of the commonly used best error stopping criterion, medical prediction models are usually trained with other criteria with the aim of attaining the best classification rate and sensitivity/specificity. In our experiments, the best CCR criterion was chosen with some alteration, so that only models with sensitivity above a certain threshold (in this thesis 50%) are considered. This way, we can avoid unsuccessful training with a high CCR but

a very low sensitivity. The log-sensitivity index was also used in the experiments as the stopping criterion to select the most suitable model. There may be some slight discrepancies between these two stopping criteria. The log-sensitivity index emphasizes sensitivity more than the CCR criterion does.

In the following experiments, an epoch of MLP training is defined as one process consisting of: feeding the training data to the network; adjusting the weights of the network; calculating the output measures (variable value, ASE, CCR, sensitivity, specificity) of every layer; and feeding back the errors obtained at the output layer within the network. Correspondingly, an epoch of RBF training is one process consisting of: selecting a new centre; calculating the weights between the hidden layer and the output layer; and calculating the measures of the network (variable value, ASE, CCR, sensitivity, specificity).

4.1 Results of the MLP network method

The MLP networks implemented in this system were of two kinds: with weight-elimination and without weight-elimination. All networks were of the three layer structure with 51 input units, 6 hidden layer units and 1 output unit. Parameters were initialized as described in Section 3.3.3. Best CCR and best Log-sensitivity index stopping criteria were used to train these networks.

4.1.1 Results of the 3-layer MLP model without weight-elimination (MLP1)

Experiments were executed on three different datasets (Groups A, B, and C) using an MLP network without weight-elimination. The training of the network began with a group of initial parameters (Table 3-4) which were later adjusted to try different parameters of the network. Table 4-1 lists the final network parameter values corresponding to the best results of the network.

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Parameter	Group A	Group B	Group C	Group A	Group B	Group C
Lr	0.0025447	0.0024403	0.0024403	0.002857	0.0025447	0.0024403
lr_inc	1.004	1.004	1.004	1.004	1.004	1.004
lr_dec	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032
Lamda	0.0001	0.0001	0.0001	0.0001002	0.0001	0.0001
lamda_inc	1.001	1.001	1.001	1.001	1.001	1.001
lamda_dec	0.999	0.999	0.999	0.999	0.999	0.999
w_0	0.01	0.01	0.01	0.01	0.01	0.01
Momentum	0.99	0.99	0.99	0.99	0.99	0.99
error_ratio	1.0226	1.0151	1.0114	1.0451	1.0226	1.0114

Table 4-1 The final parameter values of the best MLP model (without weight-elimination)

The performance of the network was measured in terms of CCR, sensitivity and specificity, and is listed in Table 4-2.

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	91.16	91.50	92.86	89.12	89.46	92.86
Sensitivity	0.78	0.78	0.78	0.89	0.89	0.78
Specificity	0.92	0.92	0.93	0.89	0.90	0.93

Table 4-2 Performance of MLP models without weight-elimination

An example of a training procedure using the best CCR stopping criterion is illustrated in Figure 4-1. This figure shows that for a given set of network parameters, the training of the network needed about 470 epochs.

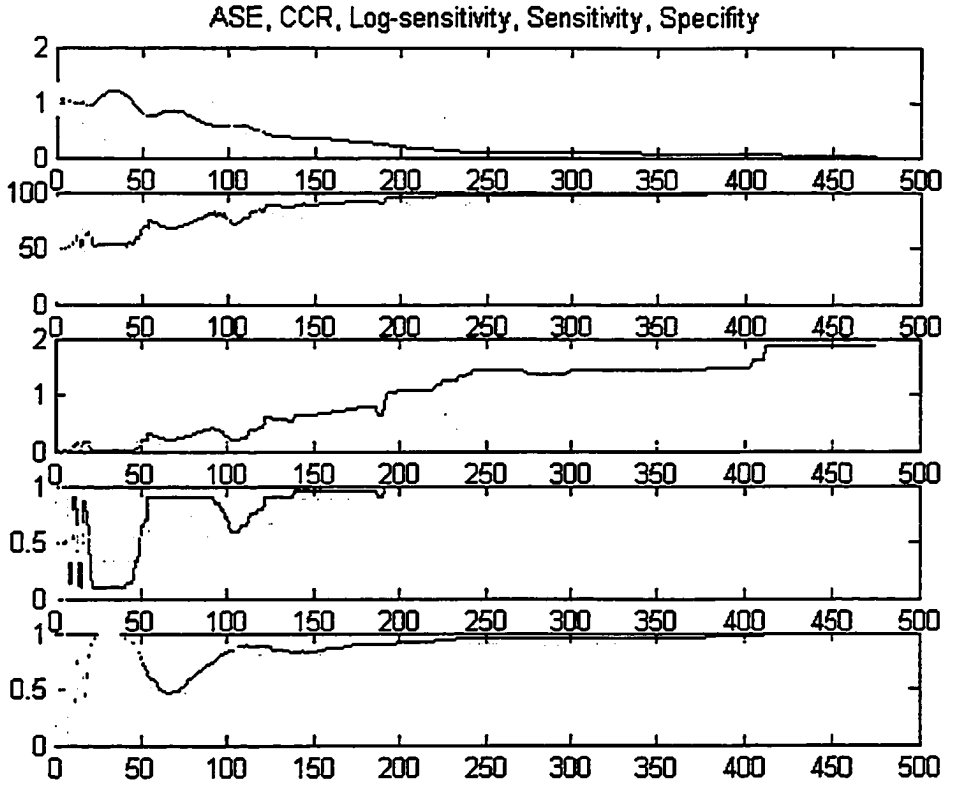

Figure 4-1 A typical training procedure of the MLP network without weight-elimination

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, and Specificity
* Training data: Group A
* Stopping criterion: best CCR

4.1.2 Results of 3-layer MLP model with weight-elimination (MLP2)

Experiments in this section were executed with the MLP model with weight-elimination. Similar to the training of the MLP model without weight-elimination, the model was trained with the same group of initial parameter values. Since there was a weight-

elimination term while adjusting the weights, the final parameter values are different from those obtained in the model without weight-elimination.

Table 4-3 and Table 4-4 list the final network parameter values and the performance measures of the network. Figure 4-2 shows the training procedure of the network.

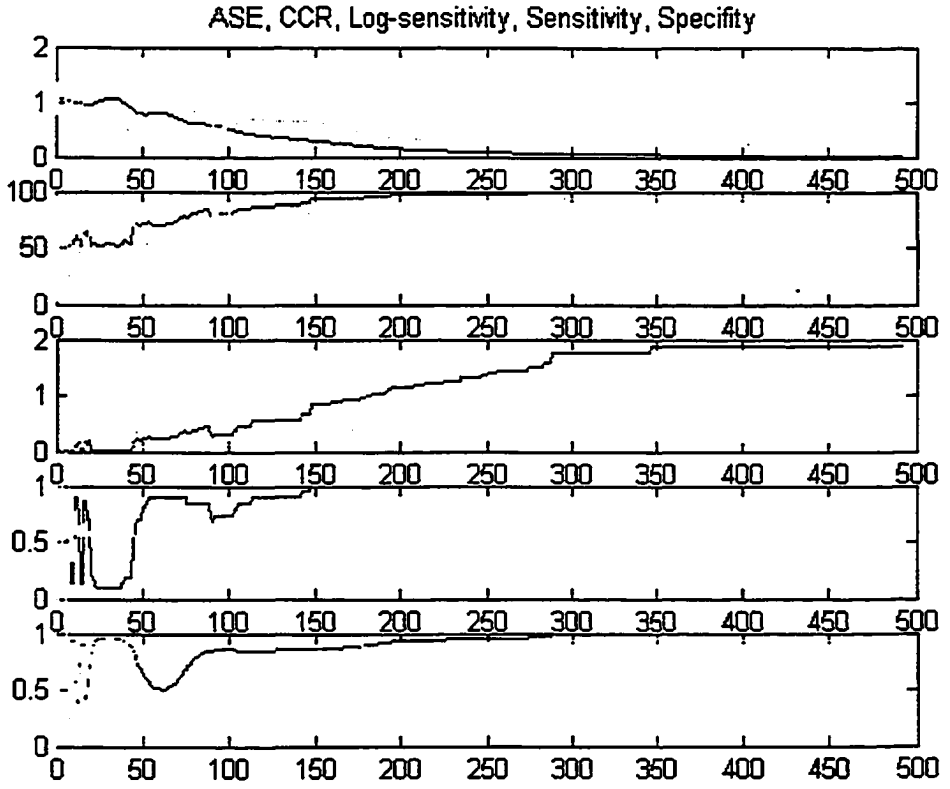

Figure 4-2 A typical training procedure of the MLP network with weight-elimination

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, and Specificity
* Training data: Group A
* Stopping criterion: best CCR

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Parameter	Group A	Group B	Group C	Group A	Group B	Group C
Lr	0.0030699	0.0029208	0.0030541	0.0030699	0.0088395	0.0030541
lr_inc	1.004	1.004	1.004	1.004	1.004	1.004
lr_dec	0.9032	0.9032	0.9032	0.9032	0.9032	0.9032
Lamda	0.0001002	0.0001	0.0001	0.0001002	0.0001	0.0001
lamda_inc	1.001	1.001	1.001	1.001	1.001	0.0001
lamda_dec	0.999	0.999	0.999	0.999	0.999	0.999
w_0	0.01	0.01	0.01	0.01	0.01	0.01
Momentum	0.99	0.99	0.99	0.99	0.99	0.99
error_ratio	1.0482	1.0482	1.0482	1.0482	1.0482	1.0482

Table 4-3 The final parameter values of the MLP model with weight-elimination

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	94.90	94.22	93.54	94.90	88.44	93.54
Sensitivity	0.78	0.78	0.78	0.78	0.89	0.78
Specificity	0.95	0.94	0.94	0.95	0.88	0.94

Table 4-4 Performance of MLP models with weight-elimination

4.1.3 Results of dimension reduction with the MLP network (with weight-elimination) (MLP3)

PCA was used to decrease the input dimensionality and determine optimal variables for the classifier through a linear transformation (Chapter 2). This method is commonly seen in RBF applications and is sometimes used in MLP applications. Using the PCA method, the original 51 input variables were transformed into a set of new combined variables. The proportion of every combined variable by principal component analysis is (%):

18.54 12.48 7.81 6.00 5.73 5.15 4.79 3.95 3.27 2.98 2.84 2.58 2.29 2.23 1.89 1.77 1.63 1.55 1.30 1.21 1.08 1.06 0.93 0.87 0.80 0.80 0.74 0.69 0.61 0.56 0.51 0.40 0.30 0.26 0.20 0.11 0.06 0.03 0.02 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

We can see the last few variables are almost of no importance, since they count for 0% in the new representation of the data. We tried different numbers of new variables to represent the data and found that 27 variables work the best. Thus, the first 27 new variables are then fed to the network as inputs to train the system. They account for 96.27 % information in total.

18.54 +12.48 + 7.81+ 6.00 + 5.73 + 5.15 + 4.79 + 3.95 + 3.27 + 2.98 + 2.84 + 2.58 + 2.29 + 2.23 + 1.89 + 1.77 + 1.63 + 1.55 + 1.30 + 1.21 + 1.08 + 1.06 + 0.93 + 0.87 + 0.80 + 0.80 + 0.74 = 96.27 (%)

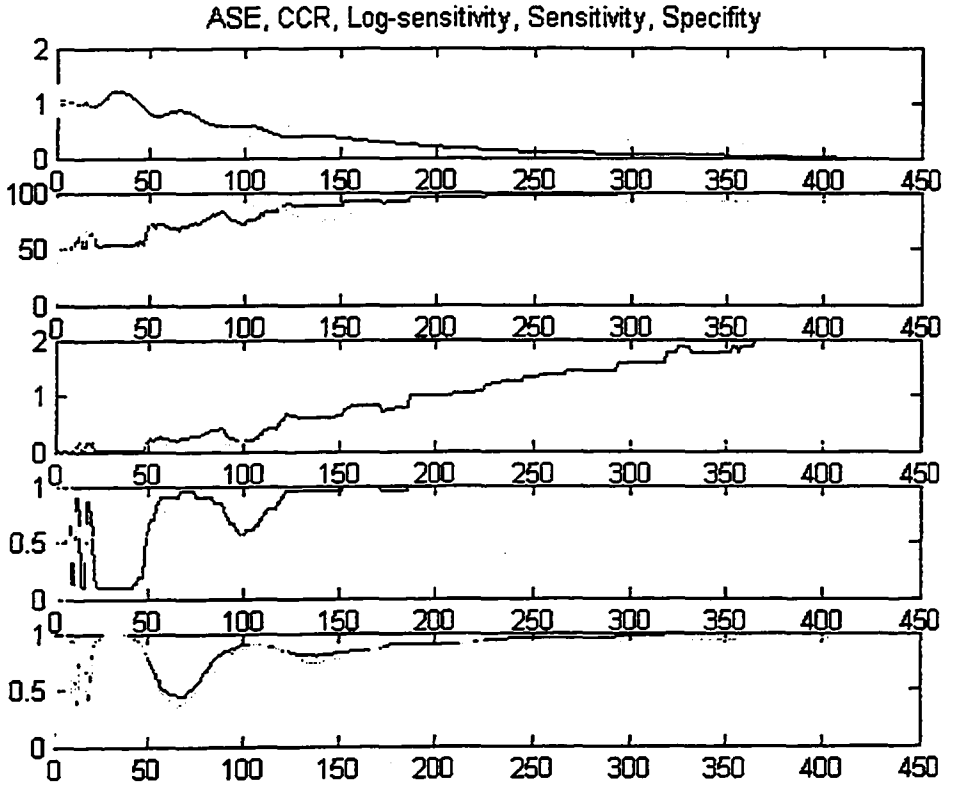

Figure 4-3 A typical procedure of the MLP network with weight-elimination +PCA

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, and Specificity
* Training data: Group A
* Stopping criterion: best CCR
* Dimension reduction (PCA): 51→27

Table 4-5 displays the experimental results of applying PCA to the MLP models (with weight-elimination). In our problem, the PCA approach does not significantly improve the performance of the MLP network.

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	93.88	97.62	93.54	93.88	93.20	91.83
Sensitivity	0.67	0.56	0.56	0.67	0.78	0.78
Specificity	0.95	0.98	0.95	0.95	0.93	0.92

Table 4-5 Performance of the MLP model (with weight-elimination) + PCA

Note:
* Dimension reduction (PCA): 51→27

4.2 Results of RBF network method

Based on the previous experiments and discussion, in this section we describe the use of the K-mean and the OLS techniques to implement RBF networks. More specifically, the approaches used in this study included a naïve K-mean method, a naïve OLS approach (with or without regularisation), and an advanced regularised OLS approach (with a K-mean method for centre candidates' initialization). A PCA approach was applied to the RBF network as well. Below we show the experimental results in detail.

4.2.1 Results of the RBF network using a K-mean method (RBF1 & RBF2)

Results using the K-mean method are not stable because this method is sensitive to the cluster initialization and the value of the cluster number K. In this section, we choose K=150 as an example to illustrate how this approach performs in our problem. Experimental results shown here are only typical examples.

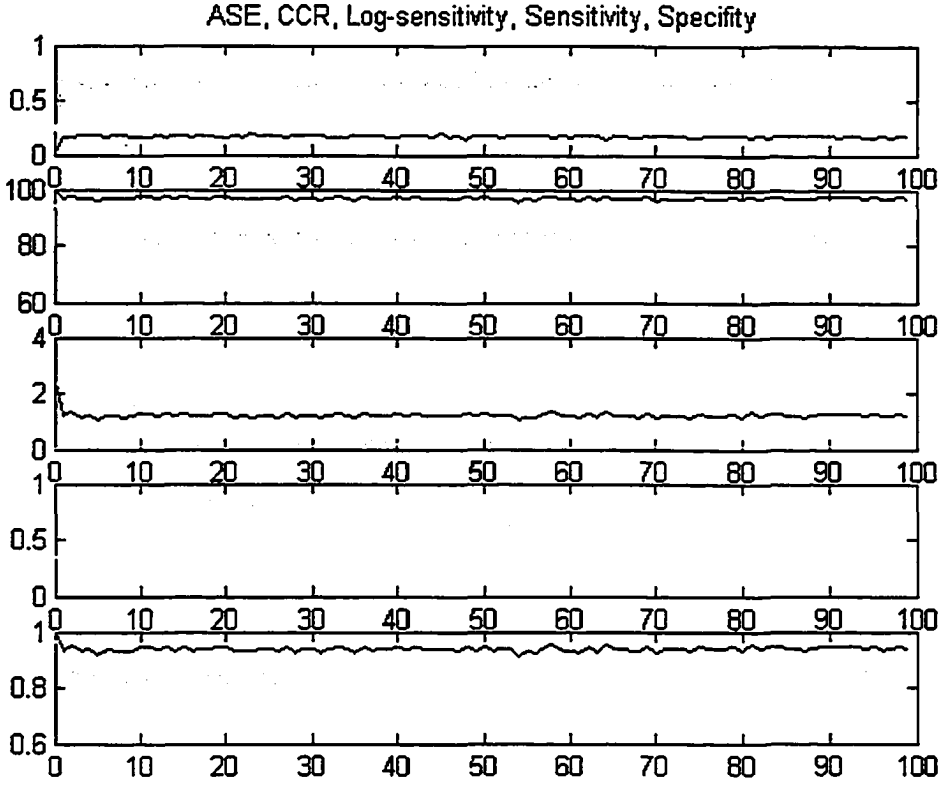

Figure 4-4 A typical training procedure of the RBF network with a K-mean method

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, Specificity
* Training data: Group A
* Stopping criterion: best CCR
*:k=150

Figure 4-4 and Table 4-6 (RBF1) show a typical training process and some typical performance results, respectively (k=150). Table 4-7 (RBF2) presents results with dimension reduction (PCA). Via PCA, the input variable number was reduced from 51 to 27. Similar to other models, this data analysis method improves the prediction efficiency a little for some datasets but not for all datasets.

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	88.10	87.41	91.16	88.10	87.41	91.16
Sensitivity	0.89	0.56	0.78	0.89	0.56	0.78
Specificity	0.88	0.88	0.92	0.88	0.88	0.92

Table 4-6 The results of the RBF network with a K-mean method

* K=150

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	89.46	89.12	89.80	89.46	89.12	89.80
Sensitivity	0.89	0.56	0.78	0.89	0.56	0.78
Specificity	0.89	0.90	0.90	0.89	0.90	0.90

Table 4-7 The results of the RBF model with a K-mean+PCA method

* K=150
* Data space dimension is reduced from 51 to 27 through a PCA process

4.2.2 Results of the RBF network using an OLS method with a naïve centre initialization (RBF3, RBF4 & RBF5)

Compared to the K-mean method, the OLS method is an advanced way of computing the centres for the RBF network because it uses a rapid algorithm to determine the centres automatically. In this section and the next, we show the experimental results for RBF networks using the OLS method. Experiments in this section used a naïve means to set up the centre candidate set, which considers the entire database as the centre candidates.

Figure 4-5 shows the training quality of the RBF network using the OLS method with a naïve centre initialization (RBF3). The generalization/prediction ability (test phase) of such a network is not ideal, although its approximation ability (training phase) is good. A regularized OLS method (RBF4) has a better generalization ability, as shown in Figure 4-6. A PCA process is also applied to this method. The performance of RBF networks using these three methods is shown in Table 4-8, Table 4-9 and Table 4-10.

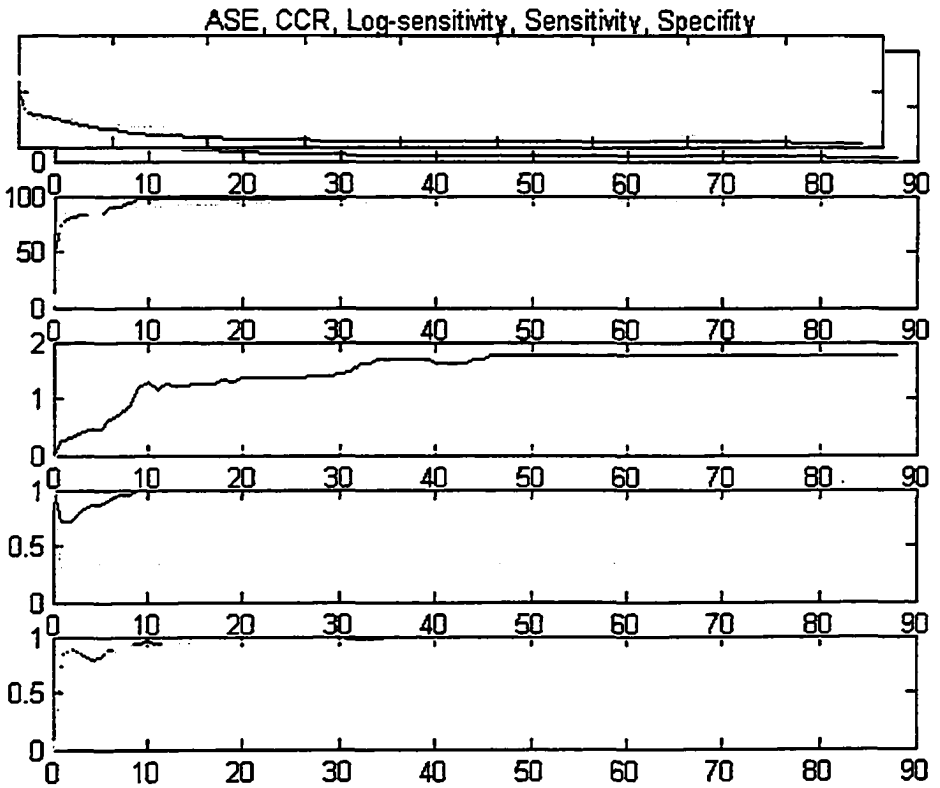

Figure 4-5 A typical training procedure of the RBF network using the naïve OLS method

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, Specificity
* Training data: Group A
* Stopping criterion: best CCR

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	90.48	91.16	92.86	90.48	91.16	92.86
Sensitivity	0.89	0.89	0.78	0.89	0.89	0.78
Specificity	0.91	0.91	0.93	0.91	0.91	0.93

Table 4-8 The performance of the RBF model using naïve OLS method

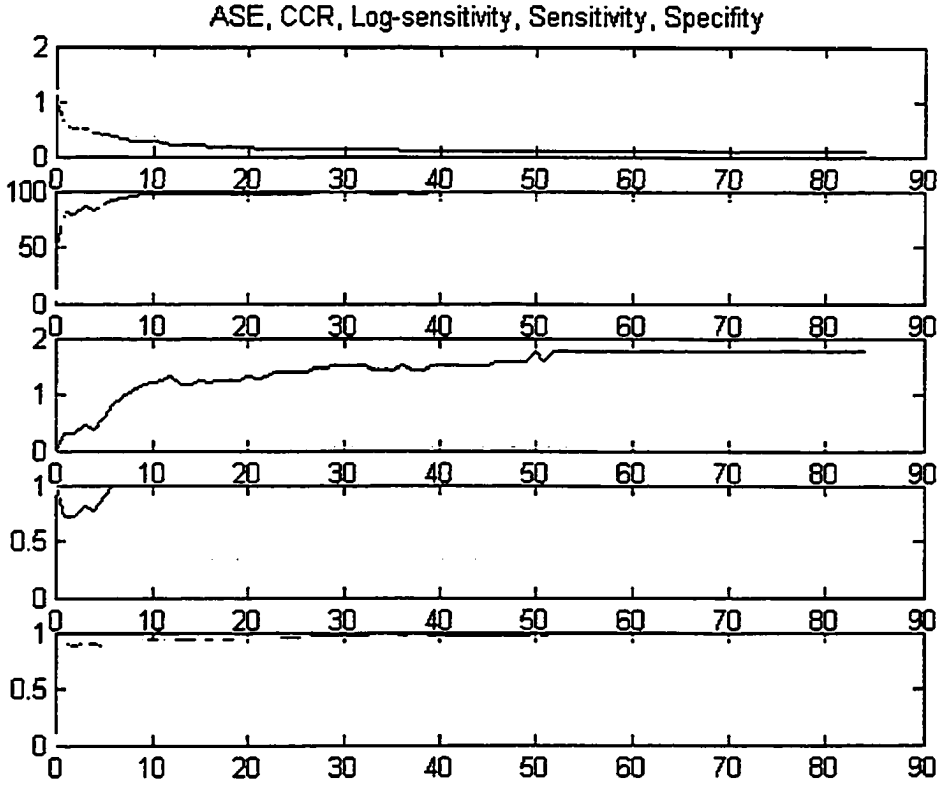

Figure 4-6 A typical training procedure of the RBF network using ROLS method with a naïve centre initialization

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, and Specificity
* Training data: Group A
* Stopping criterion: The best Log-sensitivity index

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	91.84	91.50	94.90	91.16	91.50	94.90
Sensitivity	0.78	0.89	0.78	0.89	0.89	0.78
Specificity	0.92	0.92	0.95	0.91	0.92	0.95

Table 4-9 The performance of the RBF network using ROLS method with a naïve centre initialization

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	93.20	94.56	95.24	93.20	94.56	95.24
Sensitivity	0.78	0.89	0.67	0.78	0.89	0.67
Specificity	0.94	0.95	0.96	0.94	0.95	0.96

Table 4-10 The performance of the RBF network using ROLS method with a naïve centre initialization +PCA

4.2.3 Results of the RBF network with centre initialization using K-mean method (RBF6)

All experiments on RBF models in the previous section used all the data points as initial centre candidates. This is a common method, but is less effective when the original database is not well formed. The centre candidates determined in this way may result in a poor ability to detect low occurrence patterns, although it may still exhibit a high correct classification rate. To avoid this weakness, we apply the K-mean method to determine the initial centre candidates (Chapter 3). All data points in the database are clustered first to compute representatives (centres of the clusters) for each cluster. These representatives are then used as the candidates from which the final centres are selected. The experimental results of this method (Figure 4-7 and Table 4-11) show that through the K-mean method for centre initialization, with proper K value, the performance of the network may improve somewhat. Similar to the situation using the K-mean to calculate the centres, different K values have a slight effect on the network's performance. Table 4-11 lists the experimental results with k=50 and k=150; k=50 has an optimal value. To show the difference between the performance of this method and the method using K-mean clustering only to calculate the RBFs (RBF 1), the experimental results with k=150 is also list here.

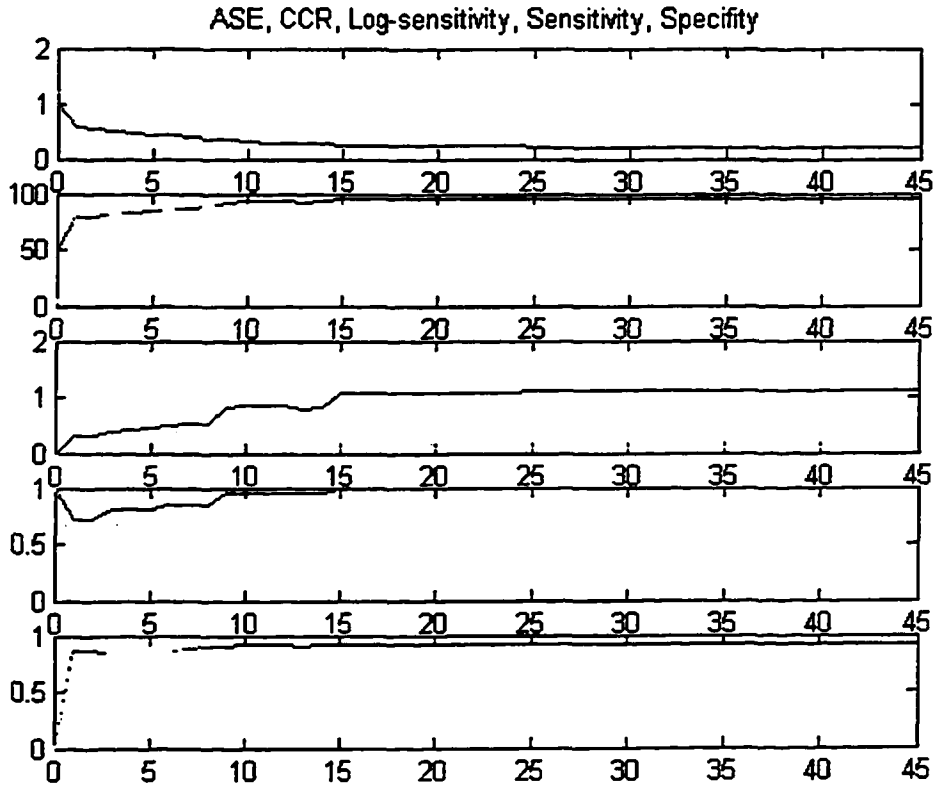

Figure 4-7 A typical training procedure of the RBF network using ROLS method with a K-mean centre initialization

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, Specificity
* Training data: Group A
* Stopping criterion: The best Log-sensitivity index
* K=50 in K-mean method for centre initialization

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	91.50	90.48	94.56	91.16	88.78	94.22
Sensitivity	0.56	0.56	0.78	0.89	0.89	0.78
Specificity	0.93	0.92	0.95	0.91	0.89	0.95

Note:
* K=150 in K-mean method for centre initialization

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	93.20	91.16	95.24	90.14	91.16	95.24
Sensitivity	0.56	0.89	0.78	0.89	0.89	0.78
Specificity	0.94	0.91	0.96	0.90	0.91	0.96

Note:
* K=50 in K-mean method for centre initialization

Table 4-11 The performance of the RBF network using ROLS method with a K-mean centre initialization

4.2.4 Results of dimension reduction using PCA (RBF7)

As we did with the MLP networks, a dimension reduction technique was applied on the RBF network as well. The RBF network referenced in this section is the one using the ROLS method with a K-mean centre initialization. The input variable number was reduced from 51 to 27. Our experiments show that the PCA method can efficiently reduce the training time (due to the reduction of the data space dimension) and increase the sensitivity (Figure 4-8 and Table 4-12).

	Best CCR with sensitivity>=0.5			Best Log-Sensitivity Index		
Performance measure	Group A	Group B	Group C	Group A	Group B	Group C
CCR	95.24	97.28	96.60	94.56	96.26	96.60
Sensitivity	0.67	0.78	0.78	0.89	0.89	0.78
Specificity	0.96	0.98	0.97	0.94	0.96	0.97

Table 4-12 The performance of the RBF network using ROLS method with a K-mean centre initialization + PCA.

Note:
* K=150 in K-mean method for centre initialization
* Data space dimension is reduced form 51 to 27 through a PCA process

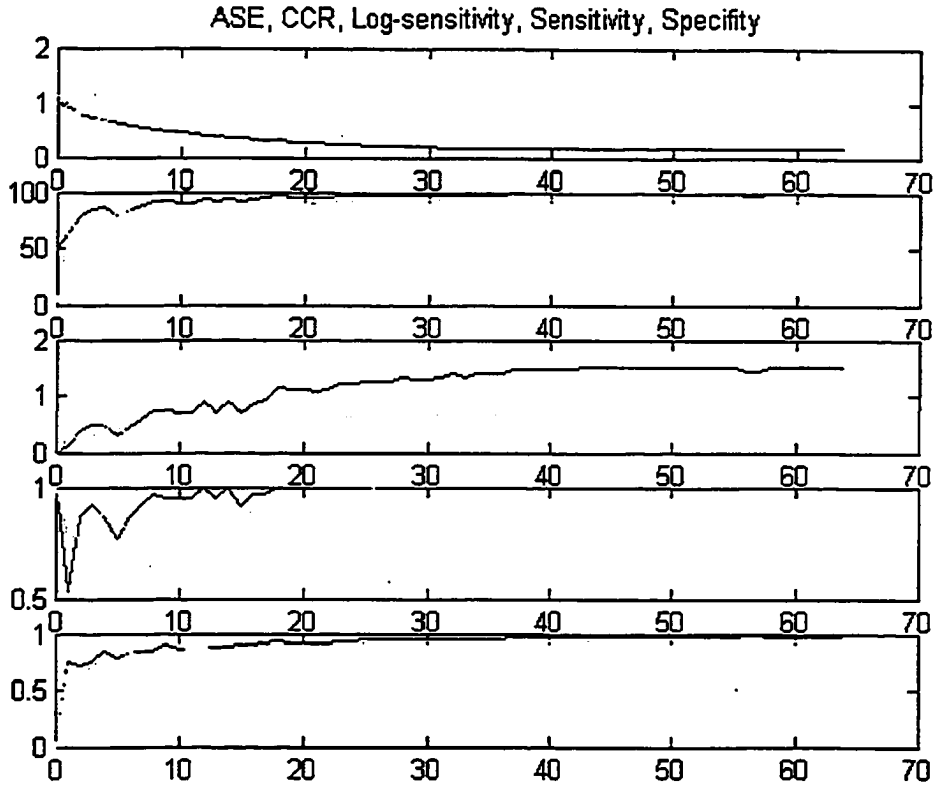

Figure 4-8 A typical training procedure of the RBF network using ROLS method with a K-mean centre initialization + PCA

Note:
* Performance measure vs. epoch. The measures are (from top to bottom): ASE, CCR, Log-sensitivity, Sensitivity, and Specificity
* Training data: Group A
* Stopping criterion: The best Log-sensitivity index
* K=150 in K-mean method for centre initialization
* Data space dimension is reduced form 51 to 27 through a PCA process

4.3 The experimental results with the non-operative database

To test performances of the designed models on non-homogeneous data, experiments were executed with the non-operative database as well (See Group D in chapter 3). All models described in the previous sections were tested in this section. Parameters are

chosen in the same way of the experiments with Post-op and will not be repeated in this section. Here we only list the results of the PCA process particularly.

Through a PCA process, the data of Group D is analysed and new combined variables are obtained in the order of importance which is shown below:

The proportion of every new variable by principal component analysis is (%):

19.03 12.55 7.64 6.34 5.83 5.33 3.81 3.33 3.15 2.89 2.44 2.34 2.18 1.97 1.77 1.68 1.60 1.51 1.47 1.27 1.24 1.20 1.11 1.01 0.89 0.80 0.76 0.69 0.63 0.55 0.53 0.48 0.46 0.36 0.35 0.25 0.23 0.20 0.08 0.04 0.03 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

In this thesis, the first 20 combined variables are selected which account for 88.13% of the information.

All experimental results of different models with the non-operative dataset are shown below:

	CCR	Sensitivity	Specificity
MLP1	87.13	0.72	0.79
MLP2	87.62	0.84	0.88
MLP3	91.09	0.72	0.94
RBF1 [a, c]	79.70	0.88	0.78
RBF2 [a, c]	82.67	0.68	0.85
RBF3	80.20	0.72	0.81
RBF4	85.15	0.52	0.90
RBF5	90.10	0.56	0.95
RBF6 [a, b]	84.16	0.96	0.82
RBF7 [a, b]	91.09	0.52	0.97

Table 4-13 Summary of the experimental results with non-postoperative dataset

Note:
* Dataset: Group D
* Stopping criteria: the best CCR
* PCA reduces the variable dimensionality from 51 to 20

* a: This value may vary depending on the K value. Here is just one example.
* b: K=150 for the K-mean centre initialization
* c: K=150 for the K-mean clustering
* MLP1: MLP network without weight-elimination
* MLP2: MLP network with weight-elimination
* MLP3: MLP network with weight-elimination +PCA
* RBF1: RBF network using K-mean clustering method
* RBF2: RBF network using K-mean clustering method + PCA
* RBF3: RBF network using OLS clustering method with a naïve centre initialization
* RBF4: RBF network using regularized OLS clustering method with a naïve centre initialization
* RBF5: RBF network using regularized OLS clustering method with a naïve centre initialization + PCA
* RBF6: RBF network using regularized OLS clustering method with K-mean centre initialization
* RBF7: RBF network using regularized OLS clustering method with K-mean centre initialization + PCA

4.4 Discussion of the experimental results

For the sake of comparison, a brief summary of the previous discussion is shown in Table 4-13 and Table 4-14. For simplicity, only the experimental results on the dataset Group A is listed for the post-operative data here.

Experimental results show that all models implemented in this thesis can provide a helpful mortality prediction tool, yet each model has its own characteristics.

Relative to the other models, the MLP model with weight-elimination has good overall prediction ability. Compared with other models, it is more flexible in detecting low occurrence cases (non-survivors) and less sensitive to the data distribution in the dataset. The overall accuracy of this model is also comparatively good (the second highest). The weakness of this model is its iterative calculation algorithm, which usually means long training time.

The RBF model using a K-mean method has the lowest accuracy (though still acceptable) of all the models, but performs better than the RBF model using a naïve OLS method in

detecting low occurrence cases. Another weakness of this approach is that its performance is not stable because of its sensitivity to the data distribution and the cluster initialization. Furthermore, the cluster number is hard to determine, and its training time is long.

The RBF model using the ROLS method with a naïve centre initialization is the fastest model. However, its ability to detect positive cases is poor when the data set is badly formed. Of all the models discussed in this thesis, the RBF model using the ROLS method and a K-mean centre initialization has the best overall performance. This robust model has the highest prediction accuracy and sensitivity, and its training time is also good (longer than the one with a naïve ROLS method, yet shorter than the other models).

	CCR	Sensitivity	Specificity
MLP1	91.16	0.78	0.92
MLP2	94.90	0.78	0.95
MLP3	93.88	0.67	0.95
RBF1 [a,c]	88.10	0.89	0.88
RBF2 [a,c]	89.46	0.89	0.89
RBF3	90.48	0.89	0.91
RBF4	91.84	0.78	0.92
RBF5	93.20	0.78	0.94
RBF6 [a,b]	93.20	0.56	0.94
RBF7 [a]	95.24	0.67	0.96

Table 4-14 A brief summary of the performance of the models with the post-operative dataset

Note:
* Dataset: Group A
* Stopping criteria: the best CCR
* PCA reduces the variable dimensionality from 51 to 27
* a: This value may vary depending on the K value. Here is just one example.
* b: K=50 for the K-mean centre initialization
* c: K=150 for the K-mean clustering
* MLP1: MLP network without weight-elimination
* MLP2: MLP network with weight-elimination
* MLP3: MLP network with weight-elimination +PCA

* RBF1: RBF network using K-mean clustering method
* RBF2: RBF network using K-mean clustering method + PCA
* RBF3: RBF network using OLS clustering method with a naïve centre initialization
* RBF4: RBF network using regularized OLS clustering method with a naïve centre initialization
* RBF5: RBF network using regularized OLS clustering method with a naïve centre initialization + PCA
* RBF6: RBF network using regularized OLS clustering method with K-mean centre initialization
* RBF7: RBF network using regularized OLS clustering method with K-mean centre initialization + PCA

As a data analysis approach, PCA plays a meaningful role in this study. Experiments with the PCA approach show that it does not only reduce the computation complexity for all datasets and all models but also improves the performance of some models. As in this study, it performs well with the RBF models but does not make much difference with the MLP models. In some cases, it can also increase the CCR and the sensitivity. This may be because it filters noise out of the dataset when reducing the dimensionality. Experiments with RBF on other datasets (Group B, C) confirm this conclusion (Section 4.2).

Chapter 5 Conclusions

5.1 Concluding remarks

Because of the special characteristics of ICU, mortality prediction systems are required to be fast and of highly accurate, with especially high sensitivity. This thesis develops a medical decision system which is mainly comprised of multi-layer Artificial Neural Networks and Radial Basis Functions. A fast RBF model was implemented based on the study of radial basis functions, linear regression, statistical analysis and data clustering. Performance comparisons and a discussion of this method and the previously implemented ANN (MLP network) method are given. To facilitate this prediction system and provide a more flexible decision aid for medical staff, multiple criteria for model selection were designed so the medical staff could choose the most suitable model.

All models were tested on four different groups of training datasets and test datasets, which originally came from a medical database. Useful variables to be extracted from the original database were determined and normalized in consultation with the medical staff. Meanwhile, due to the fact that the medical data of an ICU is usually badly skewed, a balancing process was used to artificially make the negative value cases and the positive cases equal.

Several different models using different algorithms were studied and compared. They were the MLP model without weight-elimination, the MLP model with weight-elimination, the RBF model with a simple K-mean clustering method, the RBF model

with a simple OLS method, the RBF with a regularized OLS method, and the RBF with a regularized OLS method and a K-mean centre initialization strategy. The PCA technique was applied to both the MLP and RBF models as a means of data space dimension reduction and feature extraction.

Weight-elimination (in MLP models) and regularisation (in RBF models) methods were considered as suitable for the over-fitting problem of a classifier. Our experiments show they can increase the prediction accuracy. A PCA process is supposed to be and proved in our experiments to be a good method of dimension reduction and thus training time reduction. In addition, our experiments show that it is also able to improve the performances of RBF models.

Overall, in our study, all the models mentioned above were valid for mortality prediction. Among them, the RBF with a regularized OLS method and a K-mean centre initialization strategy along with PCA data analysis had the best performance. However, the RBF with an OLS method had a faster training speed. MLP models were in mid range in terms of performance, yet their training speeds were comparatively slow.

5.2 Contributions to knowledge

The ANN approach is commonly used in the medical field. The MIRG libratory has accumulated much experience in this respect while still trying other methods. This thesis aimed to develop a new method and to provide medical staff more support using a system

that replicates the real ICU environment. In this thesis, the following contributions were made to knowledge:

1 Development of a new morality prediction support system based on the RBF network approach; this approach has been rarely used in the medical field. Different training mechanisms of the RBF models were discussed, compared and implemented.

2 Design, implementation and discussion of an advanced centre initialization method of the regularised OLS approach for RBF models by integrating the K-mean clustering technique to improve the overall generalisation/prediction ability of the system, especially for low occurrence cases. According to my literature review, this thesis appears to be the first application of this method in the medical decision support field.

3 Confirmed the good performance of the MLP network with weight-elimination as a mortality prediction support method and the validity of the use of the Log-sensitivity index as a stopping criterion for model training with the data used in this thesis.

4 Compared the performance of different MLP models and RBF models based on experimental results using a medical database. Designed a system integrating the MLP and RBF methods to include the strengths of both methods and offer more reliable and flexible decision support to medical personnel. Composed programs to preprocess data, implement the new RBF networks, integrate the MLP and RBF networks in a system, and measure the performance of the models.

5 Validation of the PCA technique as a data dimension reduction method and noise filter which reduced the system complexity and improved the prediction ability of both MLP and RBF networks in this thesis.

6 Confirmed the utility of the cross-validation method as a means of balancing the general performance error of the RBF network.

5.3 Future work

Although this was a successful project, there is still some work to be done to improve the present system:

1 Develop more efficient algorithms of centre initialization in RBF networks. In general, the system in this thesis is fast and efficient. Nevertheless, new algorithms may still improve the accuracy while sacrificing little training speed.

2 Try other data analysis means, such as Independent Component Analysis (ICA), to see the effect on classification accuracy.

3 This system can be applied to predict other medical outcomes such as the duration of ventilation, the length of stay in ICU, and the diagnosis of disease and complications.

4 Based on the methodology outlined in this thesis, develop an online model for clinical ICU monitoring. A recurrent 3-layer network can be considered as a possible network structure.

5 Apply the mechanism to the neonatal ICU to see how it works for different databases.

Reference List

1. Abramson NS, Wald KS, Grenvik AN, Robinson D, and SnyderJ.Adverse. Adverse occurrences in intensive care units. JAMA 244, 1582-1584. 1980.

2. Abu-Hanna A and de Keizer, Nicolette. Integrating classification trees with local logistic regression in Intensive Care prognosis. 1-2. Artificial Intelligence in Medicine 29, 5-23.

3. Abu-Hanna A and Lucas PJ. Prognostic models in medicine. AI and statistical approaches. Method Inf.Med. 40, 1-5. 2001.

4. Aguila L and et al. Mortality risk factors in critical surgical patients. Rev Esp Anestesiol Reanim 47, 281-286. 2000.

5. Andrews, H. Introduction to Mathematical Techniques in Pattern Recognition. Wiley Inter science . 1972.

6. Bartlett, P. L. For valid generalization, the size of the weights is more important than the size of the network. Advances in Neural Information Processing Systems 9 , 134-140. 1997. Cambrideg, MA: The MIT Press.

7. Bezerianos, S.Papadimitriou, and D.Alexopoulos. Radial basis function neural networks for the characterization of heart rate variability dynamics. Artificial Intelligence in Medicine 15(3), 215-234. 1999.

8. Blanzieri E. and Giordana A. Mapping symbolic knowledge into locally receptive field networks. Proceedings of 4th Congress of the Italian Association for Artificial Intelligence.in M.Gori e G.Soda (Eds.) 992, 267-278. 1995. Springer Verlag, Berlin.

9. Bortolan G., Brohet C., and Fusaro S. Possibilities of using neural networks for ECG classification. Journal of Electrocardiology 29 Suppl, 10-16. 1996.

10. Burke HB, Goodman PH, Rosen DB, Wenson DE, WeinsteinJN, and et al. Artificial neural networks improve the accuracy of cancer survival prediction. Cancer 79(4), 857-862. 1997.

11. Chen S., Cowan CFN, and Grant PW. Orthogonal least squares learning algorithm for radial basis function networks. IEEE Transaction on Neural Networks 2, 302-309. 1991.

12. Chen T and Chen H. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. IEEE Transactions on Neural Networks 6(4), 911-917. 1995.

13. Chen YC and et al. Risk factors for ICU mortality in critically ill patients. J Formos Med Assoc 100(10), 656-661. 2001.

14. Clermont G, Angus DC, DiRusso SM, Griffin M, and Linde-Zwirble WT. Predicting hospital mortality for patients in the intensive care unit: a comparison of artificial neural networks with logistic regression models. Crit Care Med 29(2), 291-296. 2001.

15. Coiera E. Artificial Intelligence in Medicine Chapter 19. 1997. London, Oxford Univ. Press.

16. Coldhill D and Withington P. The effect of casemix adjustment on mortality as predicted by APACHE II. Intensive Care Med 22, 415-419. 1996.

17. Cross SS, Harrison RF, and Kennedy RL. Introduction to Neural Networks. Lancet 346(8982), 1075-1079. 2001.

18. Cuthbertson BH, McKeown A, Croal BL, , Mutch WJ, and , Hillis GS. Utility of B-type natriuretic peptide in predicting the level of peri- and postoperative cardiovascular support required after coronary artery bypass grafting. Crit Care Med 33(2), 437-442. 2005.

19. Dayhoff, Judith E, DeLeo, and James M. Artificial neural networks. Conference on Prognostic Factors and Staging in Cancer Management: Contributions of Artificial Neural Networks and Other Statistical Methods 91(S8), 1615-1635. 4-15-2001.

20. Dempster AP, Laird NM, and Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. Proceedings of the Royal Statistical Society B-39, 1-38. 1977.

21. Donowitz LG. High risk of nosocomial infection in the pediatric critical care patient. Crit Care Med 14, 26-28. 1986.

22. Dorothy F.Edwards, Holly Hollingsworth, Allyson R.Zazulia, and Michael N.Diringer. Artificial neural networks improve the prediction of mortality in intracerebral hemorrhage. Neurology 53(2), 351-357. 7-22-1999.

23. Edwards DF, Hollingsworth H., Zazulia AR, and Diringer MN. Artificial neural networks improve the prediction of mortality in intracerebral hemorrhage. Neurology 53, 351. 1999.

24. Ennett CM. Coronary surgery mortality prediction using artificial neural networks. M.A.Sc.thesis . 1999. School of Information Technology and Engineering (Electrical Engineering), University of Ottawa.

25. Ennett CM. Imputation of missing values by integrating artificial neural networks and case-based reasoning. Ph.D.thesis . 2003. Ottawa, ON., Dept of Systems and Computer Engineering, Carleton University.

26. Ennett CM, Frize M, and Charette E. Improvement and automation of artificial neural networks to estimate medical outcomes. Med Eng Phys 26(4), 321-328. 2004.

27. Fabian Jaimes, Jorge Farbiarz, Diego Alvarez , and los Martinez. Comparison between logistic regression and neural networks to predict death in patients with suspected sepsis in the emergency room. Critical Care 9, R150-R156. 2005.

28. Fahoum S. and Howitt I. Combined wavelet transformation and radial basis neural networks for classifying life-threatening cardiac arrhythmias. Medical and Biological Engineering and Computing 37(5), 566-573. 1999.

29. Ferland G. and Yeap T. Prediction of nonlinear dynamical. system output with multi-layer perceptron and radial basis function neural networks. Proc.IEEE-INNS International.Joint Conference on Neural Networks (IJCNN'99), 392-397. 1999.

30. Friedhelm Schwenker, Hans A.Kestler, and Gunther Palm. Three learning phases for radial-basis-function networks. Neural Networks 14, 439-458. 2001.

31. Frize M and Ennett CM. Improving the potential clinical significance of decision-support systems using artificial neural networks. Proc AMIA Symp , 1011. 2000.

32. Frize M, Solven FG, Stevenson M, Nickerson BG, Buskard T, and Taylor K. Computer-assisted decision support systems for patient management in an intensive care unit. Medinfo 8, 1009-1012. 1995.

33. Frize M., Ennett CM, Stevenson M., and Trigg HC. Clinical decision support systems for intensive care units: using artificial neural networks. Med Eng Phys 23(3), 217-225. 2001.

34. Frize M. and Frasson C. Decision-Support and Intelligent Tutoring Systems in Medical Education. Clinical and Investigative Medicine 23(4). 2000.

35. Fung KS, Chan FH, Lam FK, and Poon PW. A tracing evoked potential estimator. Medical and Biological Engineering and Computing 37(2), 218-227. 1999.

36. Girou E, Francois S, Novara A, Safar M, and Fagon J. Risk factors and outcome of nosocomial infections: Results of a matched case-control study of ICU patients. Am J Respir Crit Care Med 157, 1151-1158. 1998.

37. Glance LG, Osler T, and Shinozaki T. Intensive care unit prognostic scoring systems to predict death: a cost-effectiveness analysis. Crit Care Med 26(11), 1842-1849. 1998.

38. Hanson C. and Marshall B. Artificial intelligence applications in the intensive care unit. Crit Care Med 29(2), 427-435. 2001.

39. Haque MA, Hasan MK, and Tazawa H. Investigation of the nonlinearity in the heart rate dynamics. Medical Engineering and Physics 23(2), 111-115. 2001.

40. Haykin S. Neural Networks: a comprehensive foundation, second edition. 1999. Prentice Hall Inc.

41. Huang RB, Law LT, and Cheung YM. An Experimental Study: On Reducing RBF Input Dimension By ICA and PCA. Proceedings of 1st IEEE International Conference on Machine Learning and Cybernetics (ICMLC'2002) 4, 1941-1946. 2002.

42. Johnston ME, Langton KB, Haynes RB, and Mathieu A. Effects of Computer-based Clinical Decision Support Systems on Clinician Performance and Patient Outcome. Ann Intern Med 120, 135-142. 1994.

43. Kayaalp M, Cooper GF, and Clermont G. Predicting ICU mortality: a comparison of stationary and nonstationary temporal models. Proc AMIA Symp , 418-422. 2000.

44. Keene AR and Cullen DJ. Therapeutic Intervention Scoring System: update 1983. Crit Care Med 11(1), 1-3. 1983.

45. Kennedy CE. and Aoki N. Generating a mortality model from a pediatric ICU (PICU) database utilizing knowledge discovery. Proc AMIA Symp , 375-379. 2002.

46. Knaus WA, Draper EA, Wagner DP, and Zimmerman JE. APACHE II: a severity of disease classification system. Crit Care Med 13, 818-829. 1985.

47. Knaus WA, Wagner DP, Draper EA, Zimmerman JE, Bergner M, Bastos PG, Sirio CA, Murphy DJ, Lotring T, and Damiano. The APACHE III prognostic system:Risk prediction of hospital mortality for critically ill hospitalized adults. Chest 100, 1619-1636. 1991.

48. Knaus WA, Zimmerman JE, Wagner DP, Draper EA, and Lawrence DE. APACHE -acute physiology and chronic health evaluation: a physiologically based classification system. Crit Care Med 9, 591-597. 1981.

49. Le Gall JR, Lemeshow S, and Saulnier F. A new Simplified Acute Physiology Score (SAPS II) based on a European/North American multicenter study. JAMA 270, 2957-2963. 1993.

50. Lemeshow S and Le Gall JR. Modeling the severity of illness of ICU patients. A systems update. JAMA 272, 1049-1055. 1994.

51. Lemeshow S, Teres D, Klar J, Avrunin JS, Gehlbach SH, and Rapoport J. Mortality Probability Models (MPM II) based on an international cohort of intensive care unit patients. JAMA 270, 2478-2486. 1993.

52. Lerotic M., Jacobsen C., Gillow JB, Francis AJ, Wirick S., Vogt S., and Maser J. Cluster analysis in soft X-ray spectromicroscopy: Finding the patterns in complex specimens . Journal of Electron Spectroscopy and Related Phenomena , 1137-1143. 2005.

53. Lewenstein K. Radial basis function neural network approach for the diagnosis of coronary artery disease based on the standard electrocardiogram exercise test. Medical and Biological Engineering and Computing 39(3), 362-367. 2001.

54. Li YC, Chiu WT, and Jian WS. Neural network modeling for surgical decisions on traumatic brain injury patients. International Journal of Medical Informatics 57(1), 1-9. 2000.

55. Mangiameli P., West D., and Rampal R. Model selection for medical diagnosis decision support systems. Decision Support Systems 36(3), 247-259. 2004.

56. Marin K and et al. Inadequate antimikrobial treatment of infections: A risk factor for hospital mortality among critically ill patients. Chest 115, 462-474. 1999.

57. McGowan HCE, Stevenson M, and Frize M. A reporting guideline for medical applications of artificial neural networks. Proc CMBEC . 1996.

58. Morris AH. Decision support and safety of clinical environments. Qual Saf Health Care 11, 69-75. 2002.

59. Morris AH. Iatrogenic illness: a call for decision support tools to reduce unnecessary variation. Qual.Saf.Health Care 13, 80-81. 2004.

60. Nikiforidis GC and Sakellaropoulos GC. Expert system support using Bayesian belief networks in the prognosis of head-injured patients of the ICU. Med Inform (Lond). 23(1), 1-18. 1998.

61. Nimgaonkar A, Karnad DR, Sudarshan S, Ohno-Machado L, and Kohane I. Prediction of Mortality in an Indian Intensive Care Unit: Comparison between APACHE II and Artificial Neural Networks. Intensive Care Med 30(2), 248-253. 2004.

62. Orr M J L. Regularisation in the Selection of RBF Centres. Neural Computation 7(3), 606-623. 1995.

63. Papadimitriou S. and Bezerianos A. Nonlinear analysis of the performance and reliability of wavelet singularity detection based denoising for Doppler ultrasound fetal heart rate signals. International Journal of Medical Informatics 53(1), 43-60. 1999.

64. Park J. and Sandberg IW. Universal approximation using radial basis function networks. Neural Comput 3, 246-257. 1991.

65. Patel, P. A. and Grant, B. J. B. Application of mortality prediction systems to individual intensive care units. Intensive Care Med 25(9), 977-982. 9-15-1999.

66. Ramoni M., Sebastiani P., and Dybowski R. Robust Outcome Prediction for Intensive Care Patients. Method Inf.Med. 40, 39-45. 2001.

67. Richardson DK, Gray JE, McCormick MC, Workman-Daniels K, and Goldmann D. Score for Neonatal Acute Physiology (SNAP): validation of a new physiology-based severity of illness index. Pediatrics 91, 617-623. 1993.

68. Rosenberg A and Watts C. Patients readmitted to ICUs: a systematic review of risk factors and outcomes. Chest 118, 492-502. 2000.

69. Taylor K.B., Nickerson B.G., Frize M., Solven F.G., and Dunfield V. Use of Case-Based Reasoning to Assist Patient Management in an IntensiveCare Unit. Proc.of the joint conference of COMP and the CMBES , 248-249. 1993.

70. Tianping Chen and Hong Chen. Approximation capability to functions of several variables nonlinear functionals and operators by radial basis function neural networks. IEEE Transactions on Neural Networks, 6(4), 904-910. 1995.

71. Titsias M. and Likas A. A Probabilistic RBF Network for Classification. IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00) 4, 4238. 2000.

72. Trigg HCE. An investigation of the methods to enhance the performance of artificial neural networks used to estimate medical outcomes. Master's thesis . 1997. University of New Brunswick.

73. Trigg HCE, Stevenson M, and Frize M. Estimating ventilation requirements in critical care medicine. Proc World Congress on Biomedical Engineering and Physics . 1997.

74. Tumer K., Ramanujam N., Ghosh J., and Richards-Kortum R. Ensembles of radial basis function networks for spectroscopic detection of cervical precancer. IEEE Transactions On Biomedical Engineering 45(8), 953-961. 1998.

75. Valdes-Cristerna, R., Medina-Banuelos, V., and Yanez-Suarez, O. Coupling of radial-basis network and active contour model for multispectral brain MRI segmentation. Biomedical Engineering, IEEE Transactions on 51(3), 459-470. 2004.

76. Walker CR, Ennett C, and Frize M. Use of an artificial neural network to estimate probability of mortality in neonatal intensive care unit patients. Pediatr Res 49, 310A. 2001.

77. Walker CR, Ennett CM, and Frize M. Will artificial intelligence systems help counselling of parents in the neonatal intensive care unit (NICU)? Paediatr Child Health 6, 28. 2001.

78. Weigend AS, Rumelhart DE, and Huberman BA. Generalization by weight-elimination with application to forecasting. Advances in Neural Information Processing Systems 3 . 1991. San Mateo, CA, Morgan Kaufmann.

79. Wong DT, Crofts SL, and Gomez. Evaluation of predictive ability of APACHE II system and hospital outcome in Canadian intensive care unit patients. Crit Care Med 23, 1177-1183. 1995.

80. Wong LS and Young JD. A comparison of ICU mortality prediction using the APACHE II scoring system and artificial neural networks. Anaesthesia 54(11), 1048-1054. 1999.

81. Zhang JP, Bloedorn E, Rosen L, and Venese D. Learning rules from highly unbalanced data sets. Data Mining, 2004.ICDM 2004.Proceedings.Fourth IEEE International Conference on 1-4, 571-574. 20041.