Automatic Relevance Determination
with Ensembles of Bayesian MLPs

Yu Fu

Submitted for the Degree of Doctor of Philosophy

Supervisor: Dr Tony Browne
Co-supervisor: Dr Matthew Casey

UNIVERSITY OF
SURREY

Department of Computing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, UK

May 2008

© Yu Fu 2008
Abstract

The problem of controlling model complexity and data complexity are fundamental issues in neural network learning. Some researchers have used Bayesian-learning on neural networks to control the model complexity. The Bayesian-based technique, *Automatic Relevance Determination* (ARD), can effectively control the complexity of data, and automatically determine the relevance of input features by controlling the distribution of corresponding groups of weights in a network. However, we found that the relevance determination made by a single ARD model is not stable and accurate. Neural network ensemble techniques were used in our research to improve the accuracy of feature relevance determination. The accuracy of the ensemble feature relevance determination was evaluated using two synthetic datasets in which the relevance of each individual input feature was pre-determined. The results showed that ensemble feature relevance determination can effectively separate relevant features, redundancies and irrelevant features from each other, and provide useful suggestions of the boundaries between these relevance levels. Thus, the features selected, based on the ensemble feature relevance determination, benefits not only non-linear models such as neural networks, but also linear models such as the linear regression model, by enabling them to classify the samples in several real-world datasets more accurately than by using all the available input features, extracted principal components and independent components from the datasets. We also found that an ensemble of ARD models is good at selecting group relevance features, but not at ranking the relevance for each individual input feature, because the relevance rank determination of an input feature can be affected by any redundancies in the data which are highly correlated with it.

**Keywords**: pattern recognition, feature selection, neural network ensembles, classification, automatic relevance determination
Acknowledgements

I would like to express my gratitude to my family, my supervisors, my friends and my fellow students. Without their support, encouragement and accompany, I would never achieved the completion of this thesis.

I am grateful to my principal supervisor, Dr. Tony Browne. Tony was always patiently listening to my problems and giving me generous advice and unfailing support. He taught me how to think critically and to justify my ideas. He also showed me how to solve research problems from different perspectives and to be persistent in accomplishing any goal.

I also want to thank my second supervisor, Dr. Matthew Casey for his detailed explanations and useful suggestions. He was always teaching me how to do things in a right way and correcting me directly when I did something improperly.

I am indebted to my friends Fan, Li, Yongyan, Jiong, Helen, Yonglan and Zhe for their great accompanies which helped me to settle in life in Guildford easily, especially in the early stage of my study here. To fellow students Craig, Jonathan and Thanos, thank you all for the stimulating conversations, discussions and laughs. I would also like to thank all my other friends who have cared me and helped me through these years; they made my life joyful and meaningful.

I appreciated the professional and friendly support from Sophie, Lydia, Gary, Noelle and other staff. Ms. Christine West has provided great support for my thesis writing, teaching me how to expressing myself clearly and improving my writing skills. I also want to convey my appreciation to the Department of Computing for the generous financial support which made it possible for me to study and live here.

Finally I offer my heartfelt gratitude to my parents, Enguo Fu and Yongzhen Feng for their unconditional support and encouragement to pursue my interests, for their selfless blessings and for the lifelong benefits of education which they bestowed on
me; to my sister, Li Fu for sharing her experiences with me, for her comforting when I was sad and her cheering when I was down, and also for her timely reminders and consistent support.
Contents

ABSTRACT ................................................................................................................................................. I
ACKNOWLEDGEMENTS ......................................................................................................................... II
CONTENTS ................................................................................................................................................ IV
CHAPTER 1 .................................................................................................................................................. 1
INTRODUCTION .......................................................................................................................................... 1
  1.1 CONTROL MODEL AND DATA COMPLEXITY .............................................................................. 2
  1.2 AUTOMATIC RELEVANCE DETERMINATION (ARD) .............................................................. 4
  1.3 CONTRIBUTIONS .......................................................................................................................... 8
  1.4 OUTLINE OF THE THESIS ........................................................................................................... 9
CHAPTER 2 .................................................................................................................................................. 11
UNCERTAINTY IN SINGLE ARD MODEL .............................................................................................. 11
  2.1 WHY A SINGLE ARD MODEL DOES NOT WORK WELL ........................................................... 13
  2.2 DEMONSTRATING THE INSTABILITY ....................................................................................... 14
  2.3 SUMMARY AND DISCUSSION ..................................................................................................... 19
CHAPTER 3 .................................................................................................................................................. 21
ENSEMBLES FEATURE RELEVANCE DETERMINATIONS ................................................................. 21
  3.1 CREATING ENSEMBLES OF ARD MODELS ............................................................................ 22
  3.2 COMBINING NETWORKS IN AN ENSEMBLE ............................................................................. 27
  3.3 SUMMARY AND DISCUSSION ..................................................................................................... 36
CHAPTER 4 .................................................................................................................................................. 38
INTERFERING FACTORS FOR AUTOMATIC RELEVANCE DETERMINATION ............................. 38
  4.1 ISSUES AND HYPOTHESES ....................................................................................................... 38
  4.2 EVALUATE HYPOTHESIS ........................................................................................................... 42
  4.3 SUMMARY AND DISCUSSION ..................................................................................................... 43
CHAPTER 5 .................................................................................................................................................. 45
LINEAR REGRESSION FOR FEATURE RELEVANCE DETERMINATION ......................................... 45
  5.1 FEATURE RELEVANCE DETERMINED BY THE LINEAR REGRESSION COEFFICIENTS ........ 46
  5.2 COMPARISON WITH THE ARD METHOD .................................................................................... 48
  5.3 CLASSIFICATION WITH SELECTED FEATURES ..................................................................... 49
  5.4 SUMMARY AND DISCUSSION ..................................................................................................... 52
CHAPTER 6 .................................................................................................................................................. 54
ENSEMBLE MEMBER SELECTION ......................................................................................................... 54
  6.1 DIVERSITY MEASURES ............................................................................................................... 56
    6.1.1 Disagreement Measure ........................................................................................................ 56
    6.1.2 Double-fault Measure ........................................................................................................ 57
  6.2 ENSEMBLE CONSTRUCTION ....................................................................................................... 58
  6.3 EXPERIMENTS AND RESULTS ................................................................................................. 59
  6.4 SUMMARY AND DISCUSSION ..................................................................................................... 61
CHAPTER 7 .................................................................................................................................................. 63
CASE STUDY: SEX SEPARATION OF TSETSE FLY PUPAE USING AN ENSEMBLE OF ARD CLASSIFIERS ............................................................................................................................................ 63
Chapter 1

Introduction

Artificial Neural Networks are mathematical models which are inspired by the way biological nervous systems process information. Neural networks, like people, learn by examples. Through learning processes, neural networks are configured for solving some humanlike problems such as generalisation and parallel processing of noisy inputs. Neural network models were developed from the early McCulloch-Pitts neuron model (1943) to more recent models, including Boltzmann Machines (Hinton & Sejnowski, 1983), Hopfield Networks (Hopfield, 1982), Competitive Learning Models (Rumelhart & Zipser, 1986), Multilayer Perceptrons (MLPs) (Rumelhart, Hinton & Williams, 1986) and adaptive resonance theory models (Grossberg, 1987). The quality of the solution produced by a neural network can be influenced by the complexity of a network model and the complexity of the data observed.

The research in this thesis has refined one of the Bayesian-based neural network models in order to control the model and data complexity. In a complex neural network model, there are many parameters that need to be adjusted during learning. However, the limited availability of training data may not provide enough information to configure all these parameters properly. Thus, the model might only learn the characteristics of some training samples, rather than the problem which needed to be solved. Normally, there are some irrelevant and redundant features in a high-dimensional dataset. These features can confuse a network when it is trying to correctly extract patterns and detect trends from the observed data. An Automatic Relevance Determination (ARD) model is a Bayesian-based neural network model (Neal, 1996). It has been argued that it not only can control model complexity by using Bayes’ theorem (Thodberg, 1996), but also it can determine the relevance of input features by controlling the distributions of weight vectors. This thesis mainly focuses on studying characteristics of the ARD technique and producing reliable
Chapter 1: Introduction

feature relevance determinations by combining an ensemble of ARD models. We have demonstrated the utility of this approach in two real world problem domains. First, based on the feature relevance determination produced by an ARD-based model, we located areas of the thalamus that are important in predicting schizophrenia. The N-acetylaspartate (NAA) concentrations within these areas are markedly lower than controls, which supports previous research findings of structural and functional alterations in these areas in schizophrenia (Browne, Jakary, Vinogradov, Fu & Deicken, 2006). Second, we provide a case study that demonstrates that the accuracy of sex separation for tsetse fly pupae can be effectively improved by using an ensemble of ARD models to select relevant features. Correctly separating tsetse fly pupae by sex can ensure only sterile males are released, which can reduce the risk of transmitting disease and help to control the population of tsetse flies.

1.1 Control Model and Data Complexity

For supervised learning, the parameters in a neural network are learned based on a set of training samples whose target values are known. Usually, these parameters are adjusted during training procedures to minimise errors between network outputs and targets. Normally, a more complex model may fit the data set better. However, if one blindly minimises the error function of a network which is trained on limited data, and does not control the network complexity, the network will inevitably be overfitted by learning spurious details and noise in the data. Occam's razor, as a principle, expounds that one should not increase, beyond what is necessary, the number of entities required to explain anything (Tipping, 2004). To prevent overfitting, several approaches have been developed. These approaches include early stopping (Prechelt, 1996), weight decay (Krogh & Hertz, 1995) and noise injection (Holmstrom & Koistinen, 1992). Early stopping uses a validation dataset to monitor the model's performance and stop training when the validation error rate “starts to go up”. The term validation dataset refers to a dataset which is not used in the training process and where the actual value of a target variable is known. The requirement of a validation dataset worsens situations where there is a shortage of training data. Moreover, in practice, it is hard to tell exactly when the validation error starts to go up, because it may go up and down numerous times during training. Weight decay uses a penalty
term in the error function to penalise large weights. Because different types of weights in the network usually require different decay constants, adjusting all these decay constants to produce good generalisation always needs huge amounts of computation. Noise injection deliberately adds random small amounts of noise to the inputs during training. Generally, the more training samples, the better it is. It seems to be a convenient way to improve training and prevent the network from approximating some specific data points too closely. However, how much noise should be added is difficult to decide. Too much noise will produce garbage, while too little noise might be no different than simply presenting the samples in the training dataset. Thus, for these techniques, controlling model complexity is rather crude and often computationally expensive. In addition, there is a lack of tools to perform analysis for the confidence of results produced by the models determined by these techniques.

Bayesian neural networks (Neal, 1996) take a different view on the learning of weights. Instead of seeking the most suitable set of weights, the Bayesian approach considers a probability distribution function over weight space, which represents the relative degree of belief in different values for the weight vector. Each neural network consists of a specification of the network structure, such as the number of hidden units, the type of activation functions, and the number of adaptable parameters (and hyperparameters). Once the data is observed, the belief in a network can be justified by using Bayes’ theorem (Thodberg, 1996) in the form:

\[
p(M_i | D) = \frac{p(D | M_i) p(M_i)}{p(D)}
\]  

(1.1)

where, \( p(M_i) \) represents the prior probability of the network \( M_i \). Normally we do not prefer one model over another, so each network has the same prior probability. Because the data prior probability \( p(D) \) is independent of the network, the network posterior probability depends on the factor \( p(D | M_i) \), which is called the evidence for the network \( M_i \). Within the evidence framework, Occam’s razor is automatically and quantitatively embodied in the learning to penalise over-flexible and over-complex models (MacKay, 1992a, 1992b; Thodberg, 1996). Thus, Neal (1996)
argued that there is no statistical need to limit the complexity of the neural network architecture, such as number of hidden units and layers, when using a well-designed Bayesian approach.

Meanwhile, the complexity of data is another critical issue which can affect the quality of network mapping from inputs to output. Machine learning is generally applied to limited stored data to learn the solution for some particular task. With technological developments, more and more measurable attributes can be included as inputs to neural networks. Initially, these attributes are thought to improve prediction performance to a certain degree. However, some inputs may have little or no relevance. With a finite training dataset, some irrelevant inputs will, by chance, be more closely associated with the targets than those truly relevant inputs. This might cause learning algorithms to require a longer time to process the information and can even affect the prediction accuracy on new data (Guyon and Elisseeff, 2003). To avoid discarding useful information and over-emphasizing the irrelevant inputs, we need a technique which can automatically take inputs into account based on their degree of importance. Following this philosophy, MacKay (1994) and Neal (1996) developed a technique, Automatic Relevance Determination (ARD), to reduce the cost of being misled by some irrelevant features and the cost of not using the useful ones. The ARD technique determines the degree of the contribution of each input, by controlling the distribution of weights associated with that input.

1.2 Automatic Relevance Determination (ARD)

The Bayesian approach to neural network modelling starts with some prior weight distribution which reflects our prior knowledge about the form of the network mapping we expected to find. Several possible schemes for prior distributions were discussed in Buntine and Weigend (1991). To simplify the analysis, both MacKay (1994) and Neal (1996) used the Gaussian distribution with zero mean as the weights' prior distribution for the ARD model. In an ARD model, each input feature is associated with a hyperparameter $\alpha$ (as it controls the distribution of other parameters, so it is called a hyperparameter). A hyperparameter represents the inverse variance of weights fanning out from an input. Figure 1.1 illustrates that the weight
vector $W_i$ includes all the weights on the connections out of an input feature $I_i$ as an example. These weights have an independent Gaussian prior with zero mean:

$$p(W_i) = \frac{1}{Z_w(\alpha_i)} \exp\left(-\frac{\alpha_i}{2} \| W_i \|^2\right)$$  \hspace{1cm} (1.2)

where, hyperparameter $\alpha_i$ represents the inverse variance of the distribution of the weight vector $W_i$ and $Z_w(\alpha_i)$ is the normalization constant which ensures that the $\int p(W_i) dW_i = 1$.

In this hierarchical structure, the inverse variances of the prior distribution of weights are the hyperparameters in the model which can be justified during learning. Therefore, the allowed degree of the smoothness of input-output mapping is indicated by the data. The values of these hyperparameters are determined by maximizing the evidence for hyperparameters, which is found by integrating model evidence over all possible weights:

$$p(D | \alpha, \beta) = \int p(D | w, \alpha, \beta) p(w | \alpha, \beta) dw$$  \hspace{1cm} (1.4)

where, $\beta$ is the inverse of the variance of noise distribution for the output. For a regression output, it is a Gaussian distribution; while for a classification output, it is a Bernoulli distribution (Ayyub & McCuen, 1997). The Bernoulli distribution is a
single-parameter discrete probability distribution. It takes value 1 with probability $p$ and value 0 with probability $q = 1 - p$. Thus, its probability mass function $f$ is:

$$f(k; p) = \begin{cases} p & \text{if } k = 1 \\ 1 - p & \text{if } k = 0 \\ 0 & \text{otherwise} \end{cases}$$ (1.5)

where, $k$ is a binary variable and the probability for $k = 1$ is $p$ and $k = 0$ is $1 - p$.

As in MacKay's (1992c) evidence framework, the evidence for hyperparameters is maximised with respect to $\alpha$ in order to determine the value of the hyperparameters. These hyperparameters $\alpha$ can be re-estimated in the form:

$$\alpha_{*}^{*} = \frac{\gamma_{*}}{||W||^{2}}$$ (1.6)

where, $\gamma_{*}$ is the number of weights which are ‘well-determined’ by the data (Penny & Roberts, 1999).

This re-estimation can be repeated several times, if desired. After several re-estimations, some hyperparameters are optimized to be infinity. An infinite hyperparameter specifies a small standard deviation for the corresponding group of weights, and therefore, these weights are restricted toward their mean of zero. With all small values of weights, that input will have little effect on the output; i.e. it is considered as irrelevant. By contrast, if an input is associated with a small hyperparameter, the group of weights on the connections from that input will have a large standard deviation, and therefore they are allowed to have large values. Thus, with those ‘big’ weights, that input is likely to have a significant effect on the output; i.e. it is considered as a relevant input. Since hyperparameters can control the magnitudes of groups of weights, in order to determine the degree of contribution for each input to output, and therefore, they can be used to indicate the relevance of inputs (MacKay, 1992a; Neal, 1996; Lampinen & Vehtari, 2001).
The evidence framework can be applied to many different models for classification or regression problems, as long as these models can be interpreted in statistical terms (Neal, 1996). Here, we focus on using this framework on feedforward neural networks, Multi-layered Perceptrons (MLPs), to solve classification problems and implement the evidence framework by using the Netlab software package (Nabney, 2002). The implementation procedure can be summarised in following steps:

1. Initialise the parameters and hyperparameters in the network. At this stage the network weights do not need to be initialised from the prior distribution defined by the hyperparameters. If there is little prior knowledge about the weights, the hyperparameters will start with small values. The initial weights, if drawn from those priors, will tend to be large. Because we use a local non-linear optimiser, which is likely to stop at a point with large weights, the final solution is unlikely to be satisfactory.

2. Train the network with a standard non-linear optimization algorithm, such as conjugate gradients, scaled conjugate gradients and the quasi-Newton algorithm (Bishop, 1995), to minimise the overall error function for an MLP network:

\[ E = S(w) = \alpha E_w + \beta E_D \]

\[ \alpha E_w = \frac{\alpha}{2} \sum_{i=1}^{W} w_i^2 \]

\[ \beta E_D = \left\{ \begin{array}{ll}
\frac{\beta}{2} \sum_{n=1}^{N} \left[ y(x_n; w) - t_n \right]^2 & \text{(regression)} \\
- \sum_{n=1}^{N} \left[ t_n \ln y(x_n; w) + (1 - t_n) \ln(1 - y(x_n; w)) \right] & \text{(classification)}
\end{array} \right. \] (1.7)

where, the \( \alpha E_w \) is a weight error term and \( W \) is the number of weights; the term \( \beta E_D \) is the likelihood error function of a target value \( t \) given an input vector \( x \) and the term \( N \) is the number of input vectors.

3. When the network training has reached a local minimum, Gaussian approximation can be used for integrating over the weights in a model with given values of hyperparameters, in order to compute the evidence for the
hyperparameters \( p(D \mid \alpha) = \int p(D \mid w)p(w \mid \alpha)dw. \) These hyperparameters will be re-estimated several times with the formula (1.6 above) to values that maximise the evidence for the hyperparameters.

4. Repeat step 2 with the newly determined hyperparameters and step 3 with the newly determined weights of the network, until convergence is reached.

The ARD technique uses these optimised hyperparameters to control the distribution of weights in order to prune and remove irrelevant input features.

### 1.3 Contributions

In this thesis, the use of *neural network ensembles* is proposed in order to improve the reliability of the feature relevance determination produced by the ARD technique, and its practical implementation method is described. This approach was discussed in a related paper by Fu & Browne (2007).

The improved feature relevance determination can tell the degree of relevance of each of the input features in a dataset. According to this information, people can decide which features should be measured and included for future data collection, in order to save time and cost on acquiring data and also to save space for storage. Meanwhile, with fewer input features, a model will need less time to find a solution for a problem. By reducing noise and irrelevant features in the dataset, prediction accuracy can also be improved.

By further study of the ARD technique, we found it suitable for selecting groups of relevance features but not for ranking the relevance of each individual input feature (based on characteristics of the ARD technique discovered during our research). This discovery can help other researchers to properly utilise the ARD technique for their research, which is discussed in a paper by Fu & Browne (2008).

In this thesis, we point out that different methods consider the relevance of features differently. The feature relevance determination made by using the ARD method can
not only benefit Bayesian MLPs but also can benefit other machine learning and statistical methods, leading to better performance.

1.4 Outline of the Thesis

Theoretically, the ARD technique can effectively explore the importance of each input by controlling how the target function depends on the data. In a practical implementation, we have to use some approximation methods to obtain the marginal likelihood (mainly involved in the Bayesian approach), because it is generally difficult to compute the marginal likelihood exactly. Although the evidence approximation tends to give better results in practice (MacKay, 1999), there are still some differences between the true and approximated values. These differences might cause some uncertainties for the parameters or hyperparameters in the model. In Chapter 2, we analyse the potential cause of these uncertainties in the ARD determinations and present the experiments we used to demonstrate these uncertainties.

We investigate successful techniques for reducing uncertainties within the members of an ensemble, in order to produce a better prediction. In Chapter 3, we describe the creation of an ensemble of the ARD models and use these models to produce a reliable feature relevance determination. An ensemble of feature relevance determinations were gradually combined to demonstrate the effectiveness of the ensemble in increasing the stability of the feature relevance determination. The accuracy of the ensemble feature relevance determination was evaluated by using a synthetic dataset in which the relevance of input features were pre-determined.

In Chapter 4, the characteristics of the ARD technique are further explored. An issue about the ARD technique addressed in other researchers' work is discussed in this chapter. We explained how the dataset used in their work was further modified to investigate the influential factors in the feature relevance ranks determined by using the ARD technique.
Chapter 1: Introduction

The relevance determination made by the ARD technique is based on the contribution of the features in the prediction. In Chapter 5, we compare the feature relevance determination made by the ARD technique and the linear regression method, to see whether or not different methods will consider the relevance of individual features to be the same. Meanwhile, relevant features selected by the ARD method, a non-linear method, are used on a linear regression model to test whether these features can also benefit a linear model.

The input features in a dataset might not all be relevant. Some features might have little or no relevance. Irrelevant features might affect the accuracy of the model prediction, so we use a feature selection method to ensure we only use the most relevant features to make a prediction. With a similar consideration, the created networks might not all be useful for a network ensemble to make a correct prediction. Thus, in Chapter 6 we discuss several groups of networks which were selected based on different criteria and tested whether the output prediction was improved by the network ensembles created with the selected groups of networks.

In Chapter 7, our ARD-based model is used on real world datasets to see how well it can cope with a real world task. Its capability on dimensionality reduction by selecting relevance features is compared with two commonly used dimensionality reduction techniques: the Principal Components Analysis and the Independent Component Analysis techniques. Meanwhile, its capability on classification is evaluated by the use of a linear regression model.

In Chapter 8, we summarise the research work included in this thesis and discuss possible future work based on current results.
Chapter 2

Uncertainty in Single ARD Model

In supervised learning, there are many factors affecting the success of algorithms on a given task. The quality of the data is one such factor. In a naïve theoretical view, having more input features should provide more information, in order to have more discrimination power. However, it is not the case in practice. If there are too many input features that are irrelevant to the target variable, not only will the model estimation process be severely complicated, but also the performance of the final model can be damaged. Even if all the input features are individually relevant to the target variable, they do not necessarily lead to good model performance, especially with limited training samples. This is because some of the available features might be redundant, which might cause the dimension of the feature space to be so large that it requires numerous samples to determine the relationship. This problem is commonly referred to as the *curse of dimensionality*, a term first coined by Richard Bellman (1961). The problems caused by inclusion of irrelevant and redundant features can be avoided by extracting new features containing the maximal information in the data or selecting only the relevant features. Hall and Holmes (2003) and Guyon and Elisseeff (2003) presented recent surveys of their research on *feature extraction* and *feature selection* for machine learning. Hall and Holmes compared six feature selection and feature extraction techniques and experimentally showed that refined (selected or extracted) features generally were beneficial for improving the prediction performance. However, there was no a single best approach suitable for all situations. The important thing seems to be choosing the right technique for a particular application. Guyon and Elisseeff summarised the results proposed in a special issue of a journal and concluded that, in general, sophisticated wrapper or embedded methods improve prediction performance compared to simple feature ranking methods like correlation. Meanwhile, for some domains, feature extraction can yield improved prediction performance and a more compact set of features. These diverse
approaches were motivated by various theoretical arguments but not a unifying theoretical framework, so they might only be suitable for some but not all data domains.

In feature extraction, the high dimensional dataset is transformed into a reduced set of features. These extracted features are expected to extract the maximal relevant information from the original input data, in order to perform the desired task using these reduced features instead of the full set of available inputs. Principal component analysis (PCA) is one of the commonly used feature extraction methods. However, it might not be well-fitted to supervised learning, because it selects projection direction by considering the variance in the data, but nothing from the target variable. Moreover, the relevance of features is not considered during determining the principal components, and therefore there is no information produced to guide the data collection process for future samples.

A good feature selection mechanism attempts to select only features which are highly relevant to the target, in order to remove most of the irrelevant and redundant features from the data. Some feature selection approaches are based on the correlation or the mutual information to identify the relevance of features (Qu, Hariri & Yousif, 2005; Battiti, 1994; Peng, Long & Ding, 2005). If the selected features are only required individually to have the largest correlation or mutual information with the target, some selected features might be redundant and some unselected features might be highly relevant when used with other features, but not useful when used on their own. Hence, some further steps are needed to reduce redundancy in the data and investigate the relevance of subsets of features. Automatic relevance determination (ARD) is a technique which can assess the relevance of each input feature during network training, as briefly introduced in section 1.1. Our research has shown that the relevant features can be effectively grouped together and separated from redundant and noisy features by an ensemble of ARD models but not simply by a single ARD model (Fu & Browne, 2007). In this chapter, we are going to discuss the reason why a single ARD model does not work well, and demonstrate the instability in the feature relevance determinations produced by single ARD models.
Chapter 2: Uncertainty in Single ARD Model

2.1 Why a Single ARD Model Does Not Work Well

The ideal Bayesian treatment for parameters whose values are unknown is to integrate them out of any prediction. MacKay (1992a) discussed an approach known as evidence approximation for the treatment of hyperparameters in a multilayer perceptron network. The evidence procedure is an iterative algorithm for determining optimal weights and hyperparameters (Nabney, 2002). During the implementation of the evidence procedure, the weights of a MLP network are firstly optimised by minimising the error function $S(w)$ (see formula (1.7)) to a local minimum. In the next step, the hyperparameters $\alpha$ are optimised by maximising the evidence for the hyperparameters $p(D | \alpha)$.

As a nonlinear model, the error function of a MLP neural network typically has multiple local minima. A parameter optimisation algorithm will detect a local minimum which is nearest to the initial weights. If optimisation starts at the point $S$ in Figure 2.1, for example, the local minimum $M_2$ will be found rather than the global minimum $M_1$.

![Figure 2.1. Illustration of local minima (adapted from Gurney (1997))](image)

The value of the detected local minimum will not have too much difference from the value of the global minimum, because the error function is typically slowly varying with the network weights when the optimisation reaches the local minimum. However, the distribution of the resulting weights might be different from the distribution of the weights at the global minimum. The hyperparameters $\alpha$ can be
interpreted as the variance of the distribution of the resulting weights, and be estimated based on the well-determined weights among the optimised weights. Thus, the subtle difference between the found local minimum and the global minimum might cause the hyperparameters to be estimated incorrectly, so that the feature relevance determination produced by a single model is not reliable.

Furthermore, under the evidence framework, the approach to find the values of the hyperparameters which maximise the posterior probability of the hyperparameters, is computationally equivalent to the type II maximum likelihood method of prior selection (Berger, 1985; Neal, 1996). This is an intuitively reasonable idea of choosing the hyperparameters which are most likely to give rise to the observed data. However, it can suffer from some deficiencies. Normally, a more complex model has lower evidence than a simpler model if they can both classify the data equally well. There is the possibility for some relevant features to be pruned out by taking their associated hyperparameters to infinity, in order to maximise the evidence for hyperparameters (Qi, Minka, Picard & Ghahramani, 2004). In other words, the relevance of some input features might be randomly overestimated by a model. Thus, even for the same input feature, different models might make different relevance determinations. Because of the overfitting of some hyperparameters, the optimised hyperparameters might incorrectly indicate the relevance for some features. To analyse the stability of feature relevance determined by an ARD model, two diabetes datasets (well-understood benchmark datasets) and one synthetic dataset (with predetermined feature relevance, so the accuracy of the prediction of feature relevance can be examined) were used.

### 2.2 Demonstrating the Instability

The two public domain diabetes datasets are the Pima Indian diabetes dataset (Merz & Murphy, 1996) and the African Americans in central Virginia (AAV) diabetes dataset (Harrell, 2002). The original Pima Indian diabetes dataset was donated by Vincent Sigillito and its data were collected by the US National Institute of Diabetes and Digestive and Kidney Disease (Merz & Murphy, 1996). This dataset contains 768
Chapter 2: Uncertainty in Single ARD Model

records of females of Pima Indian heritage who are at least 21 years old and each record has 8 input features and 1 target variable:

1) Number of times pregnant,
2) Plasma glucose concentration after 2 hours in an oral glucose tolerance test,
3) Diastolic blood pressure (mm Hg),
4) Triceps skin fold thickness (mm),
5) 2 hour serum insulin (mu U/ml),
6) Body mass index ((weight in kg)/(height in m)^2),
7) Diabetes pedigree function,
8) Age (years),
9) Class variable (0 or 1): value 1 is interpreted as tested positive for diabetes.

The diabetes diagnosis for the patients in this dataset is based on the World Health Organization criteria – if the 2 hour post-load plasma glucose was at least 200 mg/dl at any survey examination or if found during routine medical care.

The original AAV diabetes dataset contains 403 records. These records were obtained from interviews for a study to understand the prevalence of obesity, diabetes, and other cardiovascular risk factors in central Virginia for African Americans (Harrell, 2002). Each record has 18 input features and 1 target variable:

1) Subject id,
2) Total cholesterol,
3) Stabilized glucose,
4) High density lipoprotein,
5) Cholesterol / HDL ratio,
6) Glycosolated haemoglobin (the target value is based on the value of this variable),
7) Location: Buckingham, Louisa,
8) Age (years),
9) Gender: male, female,
10) Height (inches),
11) Weight (pounds),
12) Frame: small, medium, large,
13) First systolic blood pressure,
14) First diastolic blood pressure,
15) Second systolic blood pressure,
16) Second diastolic blood pressure,
17) Waist (inches),
18) Hip (inches),
19) Postprandial time when labs were drawn (minutes).

The diabetes diagnosis is based on the amount of glycosolated haemoglobin. If the glycosolated haemoglobin is larger than 7, the diagnosis of diabetes is usually taken as positive and the class variable for that record will be set to 1, otherwise it is taken as negative and the class variable will be set to 0.

For most applications, it is necessary firstly to transform the data into a new representation before using them to train a neural network (Bishop, 1995). Based on the discussion of data pre-processing in Bishop (1995), these two diabetes datasets have been pre-processed before they were presented to the networks. The data pre-processing steps for the Pima diabetes dataset were:

1. The records where the blood pressure is no bigger than zero and those where the number of pregnancies appears greater than 15 are removed.
2. The values of input features were scaled into the same range (from 0 to 1), in order to avoid misleading the network into considering irrelevant features as being more important because their values fall in a larger range.

The data pre-processing steps for the AAV diabetes dataset were:

1. Clean the dataset by removing incomplete records.
2. Remove the first feature, subject id, from the dataset, because it does not contain any information for that patient to help the model learn to accurately diagnose diabetes.
3. Remove the feature “Glycosolated haemoglobin” from the dataset. Removing this feature may influence the accuracy of classification, because the targets’
value are determined by this feature (if its value is bigger than 7, that sample is
taken as a positive diagnosis of diabetes). However, including it might
influence determining the relevance of the other features, and we are primarily
interested in knowing the relevance of the other features. Thus, this feature is
not included in the training and testing dataset, but instead provides the target
values for the networks.

4. Use 1-of-n method to recode features such as gender and frame.
5. Use body mass index (weight in kg/ (height in m)^2) to combine the two
features, height and the weight into a single feature.
6. To solve the problem of a shortage of diabetic samples in the dataset, each
input feature of diabetic samples is added in random noise that is uniformly
generated from the data range (-0.2, 0.2) to create another 56 diabetic samples.
7. Scale values of input features into the same range (0 to 1) to avoid misleading
the network by having inputs with different initial ranges.

After the pre-processing procedure to clean the data, there were 392 samples
remaining in the Pima diabetes dataset with 8 input features whose values were all
scaled from 0 to 1. Among the 392 samples, 129 samples were diagnosed with
diabetes and 263 samples were diagnosed as not having diabetes. Thus, the sample
distributions of the two classes are about 32.9% and 67.1%. For the AAV diabetes
dataset, there were 422 samples remaining with 16 input features. In these samples,
112 samples were diabetics, which include 56 original samples and 56 samples
created by adding noise, and 310 samples were not diabetics.

Each dataset was divided for training and testing. To have a network that learns both
classes equally well, the same numbers of samples from each class were randomly
selected to form a training dataset. For the Pima diabetes dataset, 116 diabetic
samples and 116 non-diabetic samples were randomly ordered in the training dataset.
There were 160 samples in the testing dataset, which included 13 diabetics and 147
non-diabetics. For the AAV diabetes dataset, 100 samples were randomly selected
from each class to form the training dataset, and then the rest of samples made up a
testing dataset which included 12 diabetics and 210 non-diabetics. Table 2.1 below
presents a summary of datasets partition.
The third dataset is a scaled sine function with random noise (SSFN). This synthetic dataset was created to evaluate the prediction of the feature relevance determination, because each feature's relevance degree was pre-determined. There were nine input features in this dataset and their relevancies were pre-determined into three levels: key features, noisy version of key features, and random noise. The first four features in the SSFN dataset are the key features for the target function, which is a sine function scaled into the interval [0,1] in the form:

\[ y = \frac{(\sin(2\pi(k_1 + k_2 + k_3 + k_4)) + 1)}{2} \]  

(2.1)

where, \(k_1, k_2, k_3, k_4\) represent the four key features. The following four features in the SSFN dataset are the noisy versions of these key features. They are created by adding Gaussian noise (scaled by 0.02, 0.04, 0.06 and 0.07 respectively) to the elements of the corresponding key features. Thus, their correlations with corresponding key features are in a descending order. These four input features were denoted as \(nk_1, nk_2, nk_3, nk_4\). The last input feature \(x_g\) is Gaussian noise with a mean of 0.5 and a variance of 0.25. There are 2000 samples in the training dataset with equal numbers of samples from the two classes, and 855 samples in the testing dataset (see Table 2.1). The class variable was set to 1 if a sample's target function was bigger than 0.5, otherwise the class variable was set to 0.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total</th>
<th>Features</th>
<th>Training</th>
<th>Testing</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pima</td>
<td>392</td>
<td>8</td>
<td>232</td>
<td>160</td>
<td>2</td>
</tr>
<tr>
<td>AAV</td>
<td>422</td>
<td>16</td>
<td>200</td>
<td>222</td>
<td>2</td>
</tr>
<tr>
<td>SSFN</td>
<td>2855</td>
<td>9</td>
<td>2000</td>
<td>855</td>
<td>2</td>
</tr>
</tbody>
</table>

To investigate the stability of the feature relevance determinations produced by single ARD models, the ARD technique was applied to 10 MLP networks. Each of these networks had the same network topology, but was initialised with different random
real numbered weights uniformly generated from -0.001 to 0.001. The procedure of optimising weights and hyperparameters was iterated three times for each network. During an iteration, weights and hyperparameters were optimised. Training was carried out for 700 epochs (experimentally determined). All of these were implemented by using the Netlab software package (Nabney, 2002) running under the Matlab software platform. Figure 2.2 illustrates the relevance rank of the first features in three datasets predicted by those ten ARD models. Ideally, these ten ARD models should produce the same feature relevance rank for an input feature. However, they ranked the relevance of the same feature differently. For example, some model predicted the first feature in the SSFN dataset as the most relevant feature; however, some model predicted it as the least relevant feature, even less relevant than the noise feature, although it is one of the key features in the dataset. The same thing happened on the features in the other two datasets: one model might predict a feature as the most relevant feature in the dataset; however, another model might consider it as the least relevant feature. Thus, the feature relevance determinations obtained by using single ARD models are not stable.

![Figure 2.2. The feature relevance ranks for the first features in the three datasets predicted by ten ARD models.](image)

### 2.3 Summary and Discussion

In the ‘evidence framework’ the hyperparameters are optimised at a local optimum in weights to estimate the variance of the distribution from which these weights come.
Although it attempts to only consider the 'well-determined' weights in the estimation, the difference between the global optimum and the local optimum weights may cause the estimated variance of the distribution of weights away from the real value. Meanwhile, during seeking the maximum of the evidence in the evidence framework, some hyperparameters associated with relevant input features are possibly pruned by being set roughly to zero. In other words, some hyperparameters may be overfitted because of the underfitting of some other hyperparameters in order to simplify the model and maximise the evidence. These factors might cause hyperparameters not to be correctly determined, so that the feature relevance determined based on these hyperparameters is not accurate. In this chapter, the instability of the feature relevance determination has been demonstrated by using two diabetes datasets and one synthetic dataset. In the next chapter, we are going to use neural network ensemble techniques to reduce the uncertainties in the hyperparameters, in order to produce stable and reliable feature relevance determination.
Chapter 3

Ensembles Feature Relevance Determinations

In the last chapter, we demonstrated that different ARD models predicted the relevance of a set of input features differently. This phenomenon is similar to a set of different neural networks producing different predictions on a set of testing samples. The reason for expecting networks to make errors on predictions for testing samples is that generally neural networks estimate a target function based on limited training data. Normally, the training dataset is not perfectly representative and the function is not so simple that it can be perfectly generalized by interpolating with limited training data. This inevitably causes the estimated function not to be identical to the target function (Sharkey, 1999). Thus, different neural networks might estimate the target function differently, leading to different error patterns on the testing data.

The idea of neural network ensembles is to utilise this error diversity against the errors made by individual neural networks in order to improve overall performance (Perrone & Cooper, 1993). The error of a neural network can be decomposed into terms of bias and variance (Geman, Bienenstock & Doursat, 1992). The bias measures the degree of the difference between the estimated function and the target function, and in terms of neural network ensembles, it measures the difference between ensemble output and the target function. The variance is the extent of the varying or scattering for the estimates, and in the terms of neural network ensembles, it measures the degree of the disagreements between ensemble members. To have low prediction error, both bias and variance are required to be low. However, there has to be a trade-off between bias and variance for network training, because attempts to decrease the bias are likely to increase the variance and vice versa. Usually the ensemble combination affects the variance in an ensemble of networks, causing it to be reduced, whereas it will not alter the bias. Thus, ideal ensemble members are expected to have low bias and high variance. Apart from good ensemble members, a suitable combination method is also required to produce a good ensemble prediction.
3.1 Creating Ensembles of ARD Models

The main motivation of neural network ensembles is to improve model performance by combining an ensemble of networks. However, combining a set of identical networks does not produce any advantage. The goal then is to create diverse ensemble members which produce different error distributions (Sharkey, 1999). Two ensemble creation strategies are usually used to achieve error diversity: altering the training data and altering the parameters of the networks. The idea of altering the training data is to have an ensemble of neural networks learn different things about the same task, in order to make errors in different sub-areas of the data space. This is implemented by having each member of the ensemble trained on slightly different subsets of training data. Some methods split the training data based on training patterns, such as cross-validation (Krogh & Vedelsby, 1995) and bagging (Breiman, 1996) and some methods split the training data based on training features, such as hill climbing (Cunningham & Carney, 2000) and genetic ensemble feature selection (Opitz, 1999). In addition, the diverse ensemble members can also be created by varying the parameters of the networks, such as initial weights, number of hidden units and learning algorithms. With different parameters, networks may model the training data differently, leading to incorrect predictions on different data spaces within the whole data space.

In this work, the latter strategy was used to create an ensemble of ARD classifiers (an ensemble of MLP networks applied with the ARD technique for classification problems). The members of the ensemble were initialised with random weights uniformly generated from three data ranges (-0.01, 0.01), (-0.001, 0.001) and (-0.0001, 0.0001) to ensure they are small and in the different data spaces. Initialising networks with random weights is helpful for avoiding problems due to symmetries in the network, and small weights can have activation functions that do not fall into the saturation regions (which can lead to a flat error surface). However, small initial weights might cause slow training (Bishop, 1995). The commonly used optimization algorithms, Quasi-Newton (Davidon, 1991), Conjugate Gradient (Hestenes & Stiefel, 1952) and Scaled Conjugate Gradient (Moller, 1993) were used to optimise the weights in the classifiers, adding another layer of diversity between ensemble members. Another parameter varied in order to create different ensemble members.
was the number of hidden units. The number of hidden units in a feedforward neural network, such as an MLP network, can significantly influence the performance of the network (Fujita, 1998). To determine a suitable varying range of hidden units without training too many networks, the ensemble members were constructed with sequential odd numbers of hidden units between 1 and 19. For each ensemble member, the evidence procedure of optimising weights and hyperparameters was iterated three times. During each repetition, the networks were trained for 700 epochs and the hyperparameters were re-estimated three times to have the network weights and hyperparameters well trained. The practical steps of ensemble creation are summarised in Table 3.1.

**Table 3.1**

**Creating an Ensemble of ARD Networks**

1. for initial weights range = {(-0.01, 0.01), (-0.001, 0.001), (-0.0001, 0.0001)}
2. for hidden units = 1:2:19
3. for algorithms = {Quasi-Newton, Scaled Conjugate Gradient, and Conjugate Gradient}
4. initialise a network
   for iteration = 1:3
5. optimise weights
   re-estimate hyperparameters
6. end
7. end
8. end
9. end

Under this framework, 90 classifiers were created. For each number of hidden units, 9 classifiers were constructed, varying in their weight distributions and the training algorithms used. The general performance of the classifiers with the same number of hidden units can be represented by an average of their predictions. Three datasets, the Pima diabetes dataset, the AAV diabetes dataset and the SSFN synthetic dataset respectively, were learned by this ensemble of ARD classifiers. To evaluate whether the ensemble members with the number of hidden units varied up to 19, could generalise these datasets well, the general classification accuracies of the classifiers having the same number of hidden units were calculated for each dataset. These
averaged classification performances are plotted in Figure 3.1 against the number of hidden units. Along with these general classification accuracies, the best and the worst performance amongst their corresponding classifiers are also plotted in Figure 3.1. The graphs in this figure show that there are no tendencies towards generalising datasets better if the classifiers are constructed with a larger number of hidden units. Thus, the number of hidden units does not need to be extended to create more ensemble members which potentially model the datasets better. Exploring models with more than 19 hidden units became computationally very costly and the ensemble performance has no improvement for these three datasets, so we configured the ensemble members with no more than 19 hidden units.

(a) The Pima diabetes dataset
Each ARD classifier in the ensemble independently produced its own determination of feature relevance. Because each ARD model started with different initial weights, the optimised ARD results (the values of hyperparameters) fell in different data ranges. The biggest ARD value in one model, for instance, could be smaller than the smallest ARD value in another model. Thus, it is nonsense to compare the absolute magnitudes of ARD results across models to see whether the relevance of a feature
predicted by an ARD model is the same as the prediction made by another model. For the feature relevance determinations made by different models to be comparable, the relevance of features was ranked in each model, based on the ARD results for that model. Thus, ensemble ARD ranking positions were combined, rather than absolute ARD values. The feature relevance ranks of the first feature in the Pima diabetes dataset, for example, produced by the 90 ARD models are plotted in Figure 3.2 (a) grouped by the number of hidden units. The feature relevance ranks of the first features in another two datasets are respectively plotted in Figure 3.2 (b) and (c).

According to the statements in Table 3.1, each group is composed of 9 classifiers which were constructed with the same number of hidden units and different initial weights and algorithms. Among the 9 classifiers, the first three were trained with the Quasi-Newton algorithm; the second three were trained with the Scaled Conjugate Gradient algorithm and the last three were trained with the Conjugate Gradient algorithm. Therefore, feature relevance ranks plotted in Figure 3.2, can not only demonstrate that an ARD model initialised with different initial weights might predict feature relevance rank differently, but also show that the ARD models created under this framework of varying training algorithms and initial weights distributions can diversify the feature relevance determinations. The variation of relevance ranks for an input feature produced by the ensemble members covers all feature ranges. In other words, an input feature might be predicted as the most relevant feature by one model and be predicted as the least relevant feature by another model.

(a) The feature relevance ranks of the first feature in the Pima diabetes dataset, grouped by the number of hidden units.
3.2 Combining Networks in an Ensemble

In the last section, we discussed how to create the diverse set of ensemble members. However, only having diverse predictions might not be enough to produce a good ensemble prediction. In this section, we are going to discuss how to combine these diverse predictions produced by the ensemble members, because effectively utilizing the diversity in these predictions can benefit the ensembles in increasing the
prediction accuracy. The commonly used ensemble combination methods generally can be classified into two categories: linear combination methods and non-linear combination methods.

Both the *simple averaging* and the *weighted averaging* (Perrone & Cooper, 1993; Merz & Pazzani, 1997) are linear combination methods. The simple averaging method treats all ensemble members equally and averages their outputs to produce the ensemble prediction:

\[
\bar{f} = \frac{1}{M} \sum_{i=1}^{M} f_i
\]  

(3.1)

where \( M \) is the number of ensemble members, \( f_i \) is the output of the \( i \)th member in the ensemble, and \( \bar{f} \) is the ensemble prediction. If the simple averaging method is referred to as combining a set of predictions by a uniform weighting, the weighted averaging method can be referred to as combining a set of predictions by a non-uniform weighting and the combination weights are normally determined on the basis of the accuracies of predictions. The ensemble prediction produced by using the weighted averaging combination method then will be the sum of these weighted outputs:

\[
f_{\text{sum}} = \sum_{i=1}^{M} w_i f_i
\]  

(3.2)

where \( w_i \) is the non-negative real-valued combination weight for the output of the \( i \)th ensemble member. The sum of these weights is constrained to be unity, \( \sum_{i=1}^{M} w_i = 1 \).

Besides these linear combination methods, some non-linear combination methods are also commonly used for ensemble combination, such as *majority voting* (Suen et al., 1992; Lam & Suen, 1997) and *order statistics* (Tumer & Ghosh, 1995). The majority voting method is widely used for solving classification problems. It takes the classification decision agreed by more than half of the ensemble members as its
ensemble decision. The order statistics combination method was introduced by Tumer and Ghosh (1995) as an alternative to linear combination. For a given input \( x \), they ordered the outputs of each of the \( M \) classifiers for a particular class, \( f_1(x) \leq f_2(x) \ldots \leq f_M(x) \). Here, the output \( f_i(x) \) represents the posterior probability of a class for a given \( x \). The minimum, medium and maximum combiners are defined as follows:

\[
\begin{align*}
    f_{\text{min}}(x) &= f_1(x) \\
    f_{\text{med}}(x) &= \begin{cases} 
    f_{M+1}(x) & \text{if } M \text{ is odd} \\
    \frac{1}{2} \left( f_{\frac{M}{2}} + f_{\frac{M+1}{2}} \right) & \text{if } M \text{ is even}
    \end{cases} \\
    f_{\text{max}}(x) &= f_M(x)
\end{align*}
\] (3.3)

The classification decision made by the maximum combiner is the class with the highest posterior probability. It is the same as choosing a classifier with the highest confidence in its decision. The minimum combiner follows a similar logic, but focuses on the class to which the given \( x \) least likely belongs, rather than on the class to which the given \( x \) most likely belongs. The medium combiner considers the most popular class determined, which produces the similar classification decision as the one made by using the majority voting method.

Kittler and Alkoot (2003) theoretically studied the sum (averaging) and majority voting combination methods. They found that when the estimated errors of a class have a Gaussian posterior distribution, the sum always outperforms the majority voting method. However, for heavy tailed distributions, majority voting may outperform the sum, especially when the margin between the posterior probabilities of two classes is small and/or the number of the ensemble members is small. They used synthetic data to confirm their theoretical hypothesis, and used real data to support their general findings. Tumer and Ghosh (1999) provided a mathematical framework to underline the reasons for improvements on classification obtained from combining networks in ensembles and quantified the gains achieved. Their experimental results obtained using a number of public domain datasets presented evidence that the order
statistics combiners improve upon the performance of individual classifiers, and the simple averaging method generally performed better than the order statistics combiners.

In this work, we chose linear combination methods to produce the ensemble predictions, because they are easy to implement and generally provide good performance. The overall topology of the ensemble combination form used in this work is illustrated in **Figure 3.3**. This diagram demonstrates that an ensemble of ARD models is trained on the same training data to independently produce their predictions for a particular task. Each trained ensemble member produces two types of outputs: the ARD results for feature relevance determinations and the classification predictions on the unseen data. The ensemble determinations of feature relevance can be obtained by combining the ARD results over all ensemble members. The ensemble classification predictions can be obtained by combining the classification outputs from all ensemble members.

![Figure 3.3. The illustration of the topology for the ensemble combination](image)

In section 3.1, we created an ensemble of ARD models and trained them on three different datasets. Each of the ensemble members independently ranked the relevance
of features for these three datasets. In section 3.1 we also demonstrated the diversity of the relevance ranks made by ensemble members, by using the first features in three datasets as examples. In this section, we will combine the relevance ranks for each dataset to obtain the ensemble feature relevance ranks for the features in these three datasets. To demonstrate the effect of combining an ensemble of networks on reducing the variance of ARD results between models to produce stable predictions, we gradually added randomly selected ensemble members in the combination. Each time 5 members were selected until all ensemble members were included. Firstly, the simple averaging combination method was used. The combined relevance ranks for the features in the Pima diabetes dataset, the AAV diabetes dataset and the SSFN dataset, are respectively plotted in Figure 3.4 (a), (b) and (c). The graphs in this figure show that with the number of combined ensemble members increasing, the uncertainties of feature relevance determinations were effectively decreased, so that the ensemble relevance ranks became stable towards some particular values. Recalling that the relevance of features in the SSFN dataset was pre-determined into three levels: the first four features are the key features in the dataset, so they are on the first relevance level; the next four features are the noisy version of key features and on the second relevance level; the last feature is random noise in the dataset and it is on the third relevance level. Figure 3.4 (c) shows that combining the ensembles of feature relevance predictions could efficiently and effectively group features together, based on their relevance levels. Once the number of combined members exceeded 20, these feature groups were clearly distinguished from each other and on the order of the pre-determined feature relevance levels. Although each feature in the SSFN dataset was correctly determined into its relevance group, the features with the same relevance level were not predicted to the same relevance rank by using the simple averaging combination method.
Chapter 3: Ensembles Feature Relevance Determinations

(a) Feature relevance ranks for the features in the Pima diabetes dataset provided by a single ARD model and by combining ensembles of ARD models using the simple averaging combination method.

(b) Feature relevance ranks for the features in the AAV diabetes dataset provided by a single ARD model and by combining ensembles of ARD models using the simple averaging combination method.
Chapter 3: Ensembles Feature Relevance Determinations

Secondly, the weighted averaging combination method was used in an attempt to improve the ensemble relevance prediction. In the weighted averaging combination method, the outputs of the ensemble members are weighted before the averaging. Within the ARD technique, the optimal hyperparameter values can reflect the relevance of their associated features and the evidence for hyperparameters defined in MacKay’s evidence framework is the likelihood function of the hyperparameters, which represents how well the hyperparameters fit the data. Thus, the combination weight $w_i$ for the $i^{th}$ ensemble member was defined in the form:

$$w_i = \frac{\text{evidence}_i}{\sum_{j=1}^{M} \text{evidence}_j} \quad (3.4)$$

where $M$ is the number of members in the ensemble and $\text{evidence}_i$ is the evidence value for the $i^{th}$ member in the ensemble. Therefore, the ensemble combination weights assigned to the ensemble members are positively correlated with the confidence of their predictions of feature relevance. Since the relevance of each feature in the SSFN dataset was pre-determined, it is easy to justify the accuracy of the relevance predictions for the features in this dataset. Thus, only the weighted
ensemble relevance ranks for the features in the SSFN dataset are plotted in Figure 3.5 to see whether the weighted averaging ensemble combination method can increase the accuracy of ensemble relevance prediction. The ensemble relevance ranks plotted in Figure 3.5 were also obtained by gradually combining 5 additional randomly selected ensemble members per step. By comparing the relevance ranks plotted in Figure 3.4 (c) and Figure 3.5, we can see the weighted averaging combination method seems more efficient than the simple averaging combination method on separating key features from their redundancies and noise. However, in Figure 3.5 the gap between the noisy version of key features and the random noise feature is narrower than the one in Figure 3.4 (c) and it is hard to distinguish these two levels. Meanwhile, both the simple averaging method and the weighted averaging method failed to assign the same relevance rank to the features which are on the same relevance level.

![Figure 3.5](image)

**Figure 3.5.** The ensemble feature relevance ranks predicted for the features in the SSFN dataset by averaging the relevance ranks, weighted with positively correlated weights.

To further test the effect of the evidence for hyperparameters on feature relevance determination, the combination weight set to an ensemble member was negatively correlated to its evidence value in the form:
where $M$ is the number of members in the ensemble, the term $M - 1$ is the normalisation constant which assures the sum of these negatively correlated weights to be unity, $\sum_{i=1}^{M} w_i^{\text{negative}} = 1$ and the term $w_i$ is defined in the equation (3.5). If the evidences have a high correlation with the accuracy of the feature relevance determinations, the weighted ensemble relevance determination made with these negatively correlated weights should be erroneous. However, the ensemble feature relevance rank predictions for the features in the SSFN dataset were not destroyed by using these negatively correlated weights. The features on the same relevance level were still properly grouped together, and gradually grouping results of these features (plotted in Figure 3.6), were similar to the results produced by using simple averaging as presented in Figure 3.4 (c). This method also failed to determine the same relevance rank to the features which are equally important to the output.

![Figure 3.6](image)

Figure 3.6. Average the relevance ranks with the negative correlated weights for the SSFN dataset.

The 8 top relevance features determined by combining ensemble of relevance predictions for the Pima, the AAV and the SSFN datasets are listed in Table 3.2 in the relevance order. These ensemble relevance determinations were respectively
predicted by using three different combination methods: simple averaging, weighted averaging with weights positively correlated with evidence, and weighted averaging with weights negatively correlated with evidence. Numbers in italics show the disagreements between the ensemble relevance determinations. These experimental results showed that weighted averaging based on the evidence value cannot enhance the accuracy of prediction of feature relevance. Thus, the relationship between the model evidence and the accuracy of hyperparameters is not strong as we expected.

### Table 3.2

<table>
<thead>
<tr>
<th>Features Relevance Order</th>
<th>5 most relevant features</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pinha</strong></td>
<td></td>
</tr>
<tr>
<td>Simple</td>
<td>6 8 2 4 5 1 3 7</td>
</tr>
<tr>
<td>Positive</td>
<td>6 8 2 4 1 5 3 7</td>
</tr>
<tr>
<td>Negative</td>
<td>6 8 2 4 1 3 5 7</td>
</tr>
<tr>
<td><strong>SSFN</strong></td>
<td></td>
</tr>
<tr>
<td>Simple</td>
<td>4 3 2 1 5 7 8 6</td>
</tr>
<tr>
<td>Positive</td>
<td>4 3 2 1 5 7 8 6</td>
</tr>
<tr>
<td>Negative</td>
<td>4 3 2 1 5 7 8 6</td>
</tr>
<tr>
<td><strong>APY</strong></td>
<td></td>
</tr>
<tr>
<td>Simple</td>
<td>2 13 5 1 16 15 8 3 12</td>
</tr>
<tr>
<td>Positive</td>
<td>2 13 5 1 16 15 8 3 12</td>
</tr>
<tr>
<td>Negative</td>
<td>2 13 5 1 15 8 3 12</td>
</tr>
</tbody>
</table>

The italics show the disagreements between ensemble relevance determinations. The simple, positive and negative respectively represent simple averaging, combination weights of members positively correlated with evidence, and combination weights are negatively correlated with evidence. In the SSFN dataset, features $f_1$ to $f_4$ are the key features; features $f_5$ to $f_8$ are the noisy version of key features and the feature $f_9$ is random noise in the dataset.

### 3.3 Summary and Discussion

In this chapter, we introduced a framework for creating an ensemble of diverse ARD models, which involved initialising models with different initial weights and different numbers of hidden units, and training models with different optimisation algorithms. We also demonstrated that the ARD models created under this framework can produce diverse feature relevance determinations. This diversity was used by ensemble combination methods to increase the reliability of the ensemble feature relevance determination. The reliability of the ensemble relevance determinations for the input features in the two public domain datasets were justified by using a synthetic
Chapter 3: Ensembles Feature Relevance Determinations

dataset, the SSFN dataset. The ensemble relevance determinations for the SSFN dataset grouped input features into several groups. The features in a group have the same pre-determined relevance level. These groups were in the correct order based on the relevance levels of their included features.

Theoretically, weighted averaging should outperform simple averaging, but in practice, simple averaging normally outperforms or gives equal performance to weighted averaging (Fumera and Roli, 2005). Under the evidence framework, the evidence for hyperparameters can potentially help to improve the ensemble relevance determination by emphasizing the proper predictions. This is because the evidence for hyperparameters $p(D | \alpha)$ is the likelihood function of the hyperparameters for given data. Thus, we expected a bigger model evidence representing its relative hyperparameters means it is more likely to give rise to the observed data. However, the experimental results demonstrated that features with different relevance levels can be more effectively separated by using simple averaging rather than using the weighted averaging combination method.

The experiments in this chapter showed that both simple averaging and weighted averaging ensemble combination methods can correctly group features which are on the same relevance level. However, the equally relevant features, such as the ones from the SSFN dataset, were not assigned the same relevance rank. In the next chapter, we investigate the factors which can affect the relevance orders of features within the relevance groups.
Chapter 4

Interfering Factors for Automatic Relevance Determination

The Bayesian based Automatic Relevance Determination (ARD) method can effectively determine the relevance of input features based on their hyperparameter values which control the scale of feature weights (MacKay, 1992c; Neal, 1996). However, the ARD method might not be a good feature relevance ranking method, as demonstrated by Wang, Jones & Partridge (2001) and Coen et al. (2005). This might be because the ARD method can only globally separate relevant features from their redundant features and irrelevant features, but cannot predict the same relevance rank to the features which are equally relevant, as we demonstrated (Fu & Browne, 2007). In this chapter, we are going to discuss the influence of correlations between features on feature relevance ranking obtained.

4.1 Issues and Hypotheses

In the last chapter, we demonstrated that the disagreement between single ARD models on feature relevance determinations can be reduced by neural network ensemble techniques. The ensemble relevance determination could effectively separate the relevant features, such as the key features in the SSFN dataset from the irrelevant features and the redundant features. However, for those relevant features, although they are identically important, they could not be predicted as belonging to the same relevance rank. Wang, Jones & Partridge (2001) also demonstrated the problem of using the ARD technique as a feature relevance ranking method, by using a synthetic dataset called LIC1. They compared the ARD technique with other methods, such as the weight product (Tchaban, Taylor & Griffin, 1998) and the decision tree (Alpaydin, 2004) and found that the ARD technique could not rank the
Chapter 4: Interfering Factors for Automatic Relevance Determination

To investigate the issues raised by their research, we re-created the LIC1 dataset, according to the details provided in their paper to repeat their experiments.

The LIC1 dataset is a well designed problem used by Partridge (1996) and Partridge & Yates (1996) for using a multi-version system of neural networks to predict an outcome. The LIC1 dataset is originally defined as:

\[
LIC1 = \begin{cases} 
1 & d((x_1, y_1), (x_2, y_2)) > \text{length} \\
0 & \text{otherwise}
\end{cases}
\]  

where \(d((x_1, y_1), (x_2, y_2))\) is the Euclidean distance between two points \((x_1, y_1)\) and \((x_2, y_2)\). The problem is whether the distance between the two points is greater than a given random value, \(\text{length}\). The LIC1 dataset was composed with 1500 samples which were generated at random with each input feature uniformly distributed on \([0, 1]\). Wang et al. (2001) modified the dataset by adding an extra input, \(m_6\), into the dataset. They set \(m_6\) to \((x_1 - x_2)\) and expected that the feature \(m_6\) was more relevant than either \(y_1\) or \(y_2\), and would relegate \(x_1\) and \(x_2\) to be redundant, as \((x_1 - x_2)\) essentially contained the information stored in both \(x_1\) and \(x_2\).

Therefore the features in the LIC1 dataset are in three different relevance levels: the feature \(\text{length}\) is a crucial feature to the decision, so it is on the first relevance level; the features \(m_6\), \(y_1\) and \(y_2\) are the essential features for calculating the distance between two points, so they are on the second level; whilst the redundant features, \(x_1\) and \(x_2\) belong to the third level. Their experimental results reported that the ARD method ranked the \(m_6\) feature as the 4th relevant feature after the features \(\text{length, y1, and y2}\). This is different from what they expected and the results obtained by other methods. On the basis of their modified LIC1 dataset, we used an ensemble of ARD models to rank the relevance for the feature \((x_1 - x_2)\), as they did, and also to predict the relevance rank for the feature \((y_1 - y_2)\). We assumed that the feature \((x_1 - x_2)\) and \((y_1 - y_2)\) would be predicted to the same relevance rank compared with the other features, because the features \(x_1, x_2, y_1\) and \(y_2\) were generated in the same way and were independent to each other. However, our experimental results,
presented in Figure 4.1 and Figure 4.2, showed that the ARD-based ensembles ranked their relevance differently. The feature \((x_1 - x_2)\) was ranked less relevant than the features \(y_1\) and \(y_2\), while the feature \((y_1 - y_2)\) was ranked more relevant than \(x_1\) and less relevant than \(x_2\).

One characteristic of the ensemble feature relevance determination revealed by these results is the tendency to group similar features together. The statistical measure correlation measures one type of similarities between input features and it is widely used in machine learning and statistics for feature relevance analysis (Yu & Liu, 2004; Qu, Hariri & Yousif, 2005). Thus, we supposed that the relevance ranks determined for the features \((x_1 - x_2)\) and \((y_1 - y_2)\) were influenced by their highly
correlated redundancies. The correlations between features \((x_1 - x_2), (y_1 - y_2)\) and their redundancies were calculated and listed in Table 4.1. The feature \((x_1 - x_2)\) was highly correlated with the features \(x_1\) and \(x_2\). Thus, the ranking prediction for the feature \((x_1 - x_2)\) might be influenced by these two features, so that it was different from the expected rank. To examine this hypothesis, the features \(x_1\) and \(x_2\) were removed from the dataset respectively. With the absence of either \(x_1\) or \(x_2\), the feature \((x_1 - x_2)\) was ranked more relevant than the features \(y_1\) and \(y_2\) (see Figure 4.3 and Figure 4.4). Comparing with the feature \((x_1 - x_2)\), the relevance rank predicted for the feature \((y_1 - y_2)\) was less affected. Thus, we deduce that the degree of the influence on the feature relevance ranking was dependent on the strength of the correlation between features, because the correlations between \((x_1 - x_2)\) and its corresponding redundancies were stronger than the correlations between \((y_1 - y_2)\) and its redundancies.

**Table 4.1**

<table>
<thead>
<tr>
<th>Feature</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(y_1)</th>
<th>(y_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1 - x_2)</td>
<td>0.7228</td>
<td>-0.7138</td>
<td>-0.0129</td>
<td>-0.0131</td>
</tr>
<tr>
<td>(y_1 - y_2)</td>
<td>0.0277</td>
<td>0.0278</td>
<td>0.6934</td>
<td>-0.6948</td>
</tr>
</tbody>
</table>

**Figure 4.3.** The feature relevance rank determination for the LICI dataset with the feature \((x_1 - x_2)\) and without the feature \(x_1\)
4.2 Evaluate Hypothesis

The SSFN dataset was used to evaluate our deduction that the correlations between features can affect the prediction of feature relevance ranks. The correlations between key features and their redundancies were predetermined in the SSFN dataset by adding different levels of noise to the key features to generate the redundancies (see data generation details in Chapter 2). In Table 4.2, it is shown that the correlations between key features and their redundancies were successively decreasing from the first key feature to the fourth key feature. An ensemble of ARD models (created as described in section 3.1) was used to determine the relevance of features in the SSFN dataset. The determined relevance ranks for the features from \( k_1 \) to \( k_4 \) were in an increasing order. This order is completely opposite to the order of the degrees of correlations between these four features and their redundancies (see the results listed in Table 4.2 and Figure 4.5). Thus, the determination of the feature relevance rank can be locally influenced by the redundancies in the dataset, and the degree of the influence is based on the strengths of their correlations. In other words, if a feature has a higher correlation with its redundant feature, the prediction of its relevance rank is more likely to be affected.
Chapter 4: Interfering Factors for Automatic Relevance Determination

### Table 4.2

**Correlations between key features and their noisy version features**

<table>
<thead>
<tr>
<th></th>
<th>$k_1$</th>
<th>$k_2$</th>
<th>$k_3$</th>
<th>$k_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$nk_1$</td>
<td>0.9975</td>
<td>0.0215</td>
<td>0.0152</td>
<td>-0.0014</td>
</tr>
<tr>
<td>$nk_2$</td>
<td>0.0198</td>
<td>0.9911</td>
<td>-0.0344</td>
<td>0.0066</td>
</tr>
<tr>
<td>$nk_3$</td>
<td>0.0172</td>
<td>-0.0324</td>
<td>0.9786</td>
<td>0.0125</td>
</tr>
<tr>
<td>$nk_4$</td>
<td>0.0043</td>
<td>0.0152</td>
<td>0.0087</td>
<td>0.9722</td>
</tr>
</tbody>
</table>

![Graph showing feature relevance rank for SSFN dataset.]

**Figure 4.5.** Feature relevance rank for SSFN dataset.

### 4.3 Summary and Discussion

The above experimental results demonstrated that an ensemble of ARD models can effectively separate features, and rank them based on the degree of their relevance, although the determination of the relevance ranks for the essential features can be influenced by their corresponding redundancies. The ensemble relevance determination seems to have a certain degree of tolerance to this influence. This is reflected in the changing of the relevance ranks for the feature $(x_1 - x_2)$. It was changed from lower than the ranks of the features $y_1$ and $y_2$ to higher than their ranks when either the feature $x_1$ or $x_2$ was removed from the dataset; i.e. the influence from redundant features was reduced. However, we still do not clearly know the degree of the tolerance to this influence, which should be the topic of future study.
Chapter 4: Interfering Factors for Automatic Relevance Determination

The ARD method might not be a good feature relevance ranking method, because of its failing on some cases. Nevertheless, the ARD method can be a good feature selection method, because it can correctly distinguish the relevant features from the irrelevant features in order to produce useful suggestions about which subsets of features should be selected for a particular task.

The feature relevance determination made by an ARD model is based on the contribution of each input feature used by a model for solving a problem. It is interesting to know whether those input features will make the same contribution to other different methods when attempting to solve the same problem. In the next chapter, we will introduce a linear regression method. This method will be compared with the ARD technique to investigate whether the linear method and the non-linear method could produce the same feature relevance determination, according to the contributions of input features to each method for finding a solution for a problem.
Chapter 5

Linear Regression for Feature Relevance Determination

Regression analysis is a statistical technique for modelling the relationship between variables. In a linear regression model a dependent variable $Y$ can be explained by a number of independent variables $X_1, X_2, \ldots, X_n$ in the form:

$$Y = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3 + \ldots + a_n X_n + \varepsilon$$

where the regression coefficients $a_0, a_1, \ldots, a_n$ are the parameters in the model and $\varepsilon$ is an error term, in a statistical view, which is a random variable accounting for the failure of the model to fit the data exactly (Montgomery, Peck & Vining, 2006). The dependent variable in the literature is also called the response variable, endogenous variable and criterion variable. The independent variables are also called regressors, predictors, explanatory variables and exogenous variables (Sengupta & Jammalamadaka, 2003). When a linear regression model involves more than one regressor variable, it is usually called a multiple linear regression model. The term linear here is employed to indicate the model is a linear function of the unknown parameters $a_0, a_1, \ldots, a_n$, but not the regressors $X_1, X_2, \ldots, X_n$. The parameter $a_0$ is the intercept of the regression plane. The parameter $a_i$ indicates the rate of the expected change in response variable with the regressor $X_i$, when the other regressors are held constant (Berry & Feldman, 1985; Allison, 1999).

The quantitative relationship between a group of regressors and a response variable established by a multiple linear regression method is useful for:

1) Understanding which regressors have the greatest effect (the ones with the greatest magnitude of regression coefficients will have the most effect);
2) Knowing the direction of the effect (i.e. with a positive coefficient when $x$ is increased, $y$ will be increased, and with a negative coefficient when $x$ is increased, $y$ will be decreased);

3) Making predictions of the future values of the response variable based on currently known values of the regressor variables.

### 5.1 Feature Relevance Determined by the Linear Regression Coefficients

A multiple linear regression model separates the effects of regressor variables on the response variable, so that the unique contribution of each variable can be examined and the magnitude of their contribution can be estimated. The regression coefficients for the multiple linear regression method are the slopes of regressions which represent the average amount that the response variable $Y$ changes when the corresponding regressor variable changes by 1 unit and the other regressor variables are held constant. Because these coefficients depend greatly on the units of measurement for the regressor variables and the response variable, it is hard to compare the various coefficients for the regressor variables measured on different scales. For example, in the equation:

$$y = 0.1x_1 + 10x_2$$  \hspace{1cm} (5.2)

the variable $x_1$ is on a scale from 1000 to 2000 and the variable $x_2$ is on a scale from 0 to 1. Even if the variable $x_1$ is set to its smallest possible value 1000 and the variable $x_2$ is set to its largest possible value 1, the term $0.1x_1 = 0.1 \times 1000 = 100$ still has bigger effect on the response variable $y$ than the term $10x_2 = 10 \times 1 = 10$. However, if all the variables are in the same scale, the coefficients can be compared to get some idea of which variables are more or less important.

The classification problem can be considered as a special case of regression problem, where the output space $Y$ is discrete, $Y = \{c_1, c_2, c_3 \ldots\}$, rather than continuous. Here, the $c_i$ refers to a possible class to which each input sample can be predicted. Thus, a linear regression model also can be used for a classification problem. For two
classification problems, the SSFN and the LIC1 datasets, their input features are all in the same data range, so the relevance of these features can be determined based on their regression coefficients. A linear regression model was built by using Matlab, which solves these unknown regression coefficients by minimizing the sum of the squares of the residuals (least-square fit). The sign before each coefficient indicates the direction of a regressor variable’s effect on the response variable, but does not relate to the contribution degree of that regressor variable. Therefore, the input features from the SSFN dataset and the LIC1 dataset were sorted respectively in a descending relevance order based on the absolute values of their corresponding coefficients. The sorted feature relevance for features in the SSFN dataset is presented in Figure 5.1 below. The information demonstrated in Figure 5.1 shows that the noise in the dataset was effectively distinguished from other features, and considered as the least relevant feature in the dataset. However, the highly correlated input features (the key features and their redundancies) were determined to have similar contributions to the output. For example, the feature \( nk_1 \) was considered as the first relevant feature and the feature \( k_1 \) was considered as the second relevant feature. In other words, the relevance ranks of key features and their corresponding redundancies were next to each other, regardless of the difference of the relevance degrees between relevant features and redundant features.

![Figure 5.1](image.png)

**Figure 5.1.** Input features in the SSFN dataset are sorted in a descending relevance order on the basis of the absolute values of their corresponding regression coefficients.

**Figure 5.2** shows that the input features in the LIC1 dataset and the absolute values of their corresponding regression coefficients. These features were presented in a
descending relevance order from left to right. In this feature relevance determination, the crucial feature length was highly emphasized by the linear regression model. The interesting thing is that although the features $x_1$ and $x_2$ are highly correlated with feature $(x_1 - x_2)$, they were not ranked next to each other like the relevance predictions made for the features in the SSFN dataset. Except for the feature length, the linear regression model concerned all the input features have almost the same contribution to the output. The magnitudes of their regression coefficients were similar to each other and all were smaller than 0.1 which is more than 10 times less than the magnitude of the coefficient for the feature length. Thus, these features have a similar and minimal effect on the output, regardless whether a feature is a redundancy of another feature or not.

![Figure 5.2](image.png)

**Figure 5.2.** Input features in the LIC1 dataset are sorted in a descending relevance order on the basis of the absolute values of their corresponding regression coefficients.

### 5.2 Comparison with the ARD Method

The feature relevance determined by both the ARD technique and the linear regression model are all obtained on the basis of the contributions of input features on the output. The ARD technique is a non-linear method and the multiple linear regression is a linear method. Thus, comparing the feature relevance determination produced by these two methods, we can know whether a feature makes the same contribution for a linear and a non-linear model when making a prediction. The feature relevance determination made by a linear regression model for two synthetic
datasets were compared with the relevance determination made by an ensemble of ARD models (presented in Chapter 4). With this comparison, we can see that these two methods follow different mechanisms to determine the contribution of input features on finding the solution for the same problem. The linear regression model considered the highly correlated features as having a similar contribution, regardless of whether they were redundant or not. On the other hand, an ensemble of ARD models took the redundancy into account, so the features and their redundancies were treated differently. The key features in the SSFN dataset, for example, are assigned higher relevance ranks by an ensemble of ARD models when compared to their redundancies.

5.3 Classification with Selected Features

So far, we have discussed and demonstrated the advantages of the feature relevance determination made by combining an ensemble of ARD models. However, we were still curious whether the classification accuracy can be increased by using these selected features. The ensemble feature relevance determination, made for the two synthetic datasets, LIC1 and SSFN, demonstrated that the ARD method can effectively separate the relevant features from their redundancies and noise. Those breaking points on the plotting suggested the potential boundaries of different degrees of feature relevance (see Figure 4.5). Thus, we supposed the features selected by using an ensemble of ARD models could also be helpful for the linear regression method to increase its classification accuracy (though, as described above, they select their own relevant features on the basis of different factors). Two public domain datasets, the Pima diabetes dataset and the AAV diabetes dataset, were respectively classified by using a linear regression model and an ensemble of ARD classifiers with sets of selected features. The ensemble feature relevance determination for the features in the Pima diabetes dataset is presented in Figure 5.3.
According to the breaking points on the plotting in Figure 5.3, the potential groups of relevant features for the Pima diabetes dataset are \{6\}, \{6,2,8\} and \{6,2,8,4,5,3,1\}. These groups of features were respectively fed into an ensemble of ARD classifiers and a linear regression model to predict whether a patient had diabetes or not. Their prediction accuracies are listed in Table 5.1. From these results we can see the best accuracy for an ensemble of ARD models is 77.32% obtained by using 3 selected features, which is better than the accuracy of 75.72% using all the input features. Meanwhile, the best classification accuracy for the linear regression model is 76.25% obtained by using 3 selected features as well, which is better than the accuracy of 75.63% obtained by using all the input features and it is slightly worse than the best result obtained by using an ensemble of ARD classifiers.

<table>
<thead>
<tr>
<th>selected features</th>
<th>ARD ensembles</th>
<th>linear regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>51.25 ± 0.00%</td>
<td>62.50%</td>
</tr>
<tr>
<td>6,2,8</td>
<td>77.32 ± 0.66%</td>
<td>76.25%</td>
</tr>
<tr>
<td>6,2,8,4,5,3,1</td>
<td>76.82 ± 0.86%</td>
<td>75.00%</td>
</tr>
<tr>
<td>all features</td>
<td>75.72 ± 0.76%</td>
<td>75.63%</td>
</tr>
</tbody>
</table>

The best performance for each method is highlighted.

The ensemble feature relevance determination for the AAV diabetes dataset is presented in Figure 5.4. According to the breaking points on the plotting, the
potential groups of relevant features for the AAV diabetes dataset are \( \{2\} \), \( \{2, 5, 13\} \), \( \{2, 5, 13, 1, 16, 8, 15, 3, 12, 14\} \), and \( \{2, 5, 13, 1, 16, 8, 15, 3, 12, 14, 4, 10, 6\} \).

The sub-datasets created with these groups of selected features were used to predict whether a patient had diabetes or not. The classification accuracies of the predictions made by an ensemble of ARD models and a linear regression model based on these sub-datasets are listed in Table 5.2. These results show that the best classification accuracy for the ARD classifiers is 90.45% with 13 selected features. This result is better than the percentage of 88.47% produced by using all the available input features. For a linear regression model, the best classification accuracy is 95.04% with 1 selected feature. This result is better than the classification accuracy of 86.94% by using all the input features and even better than the best result obtained by using an ensemble of ARD classifiers.

For both the Pima and the AAV diabetes datasets, the sub-datasets with selected features can produce better classification results than the datasets with a complete set of features, whether using an ensemble of ARD classifiers or a linear regression model. Thus, the features selected by using an ensemble of ARD models contain the important characteristics in a dataset for solving a problem, and also contain less noise than the original dataset. Therefore, both a non-linear model and a linear model classified samples more accurately by using the ARD-selected features rather than all the available features from the dataset. For the AAV dataset, the best performance
produced by a linear regression model is better than the best performance produced by an ensemble of ARD classifiers. This is probably because it is a purely linear problem. Thus, it would require all the hidden units of every network in the ensemble to be operating in the linear section at the exact centre of the hidden unit transfer function. Such a minimum may be very difficult to locate in the weight space.

### Table 5.2

<table>
<thead>
<tr>
<th>selected features</th>
<th>ARD ensembles mean% ± std</th>
<th>multiple linear regression</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>all features</td>
<td>88.47 ± 0.68</td>
<td>86.94%</td>
</tr>
<tr>
<td>2,5,13,16,8,15,3,12,14</td>
<td>90.45 ± 0.51</td>
<td>86.49%</td>
</tr>
<tr>
<td>2,5,13,16,8,15,3,12,14,4,10,6</td>
<td>85.63 ± 0.81</td>
<td>87.39%</td>
</tr>
<tr>
<td>2,5,13,16,8,15,3,12,14,4,10,6</td>
<td>85.41 ± 0.61</td>
<td>90.09%</td>
</tr>
<tr>
<td>2</td>
<td>82.43 ± 0.00</td>
<td>95.04%</td>
</tr>
<tr>
<td>all features</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 5.4 Summary and Discussion

A linear regression method was introduced in this chapter and it was used to determine the relevance of features. A linear regression model determines the relevance of input features based on the contributions of individual features to the output. By comparing the feature relevance determination made by an ensemble of ARD models and a linear regression model, we found that different methods could choose different features as being relevant for their own solutions for a problem. The feature relevance determinations made for the two synthetic datasets showed that an ensemble of ARD models treated relevant features and their redundancies differently, whereas a linear regression model treated them similarly. Consequently, the ensemble relevance determination predicted the features containing extra noise to have lower relevance than the features without that noise.

The selected features obtained using an ensemble of ARD models were used to diagnose diabetes for the patients in the Pima diabetes dataset and the AAV diabetes
dataset. With selected features, both an ensemble of ARD classifiers and a linear regression model can diagnose diabetes more accurately than using all available input features. This means the features selected by an ensemble of ARD models contained the essential information for solving a problem, so not only can they benefit the ARD classifiers but also they can benefit the linear regression model to solve a problem more accurately.

Each feature included in a dataset is supposed to be useful for a model to make a correct prediction, but in practice not all of these features can be helpful, and some features even can worsen the prediction. In this chapter, we showed that properly selecting features can help a model to produce better performance. Extending this idea to an ensemble of neural networks, each member in an ensemble is created with the intention of increasing the accuracy of ensemble prediction. However, it is possible that not all are really helpful. Thus, properly selecting the ensemble members may also be beneficial in making better, i.e. more accurate, predictions. In next chapter, we are going to discuss selecting ensemble members using different selection techniques, to see whether selecting ensemble members can work as well as selecting features.
Chapter 6

Ensemble Member Selection

Neural network ensembles combine a set of independently trained neural networks to integrate the knowledge acquired by component networks and they are often found to give superior performance to the single-best trained network. The combination of an ensemble of classifiers has been widely studied to achieve high classification accuracy (Xu, Krzyzak & Suan, 1992; Tumer & Ghosh, 1996; Fumera & Roli, 2005; Kittler, Hatef, Duin & Matas, 1998; Sohn & Shin, 2007). Furthermore, a good set of networks is also important to create a good ensemble. It has been theoretically and empirically demonstrated that a good ensemble is the one where the individual networks were both accurate and comparatively diverse (Krogh & Vedelsby, 1995; Opitz & Shavlik, 1996; Tsymbal, Pechenizkiy & Cunningham, 2004). However, in some cases, combining more diverse classifiers is more important than combining those with better performance. Hashem, Schmeiser and Yih (1994) experimentally demonstrated that the effectiveness of ensembles did not have to depend on the accuracy of the ensemble members. They achieved better ensemble accuracy by combining poorly trained networks rather than by combining well trained networks. For example, the well-known ensemble learning algorithm, boosting (Schapire, 1990; Bauer & Kohavi, 1998), improves the ensemble generalisation ability by combining a set of “weak” learners. These weak learners are created by using the training samples that have been incorrectly predicted by the previous neural networks, and their incorrectly predicted samples play an important role in the training of later networks. Although, for a classification problem, these weak classifiers usually can only classify samples slightly more accurately than random guessing, their combination can produce highly accurate classification prediction. This example shows that an ensemble with members specially designed to make error on different data spaces can produce more accurate classification, because the individual low-accuracy of these classifiers is compensated for by the error diversity. However, a problem with the
boosting algorithm is that with a finite amount of training samples, unless the first network has very poor performance, there might not be enough samples to generate later training datasets.

Zhou, Wu & Tang (2002) used a large empirical study to analyse the relationship between the ensemble and its component neural networks, and revealed that it may be a better choice to ensemble many, instead of all, of the available neural networks. Meanwhile, the experiments in Aksela & Laaksonen (2006) also demonstrated that performance is always significantly increased by pruning the number of members in a classifier committee. Thus, selecting classifiers to reduce the number of shared failures among the ensemble members, could improve classification accuracy. Quantifying the diversity of classifiers, however, is difficult because there is no formal definition theory. Numerous error diversity measures have been proposed in the literature. These measures were widely studied and compared for ensemble accuracy improvement and for selecting classification ensemble members (Aksela & Laaksonen, 2006; Gal-Or, May & Spangler, 2005; Banfield, Hall, Bowyer & Kegelmeyer, 2005; Granitto, Verdes & Ceccatto, 2005). The disagreement diversity measure (Skalak, 1996) and double-fault diversity measure (Giacinto & Roli, 2001) are two of the most commonly used and the most easily to be implemented diversity measurements. Tsymbal, Pechenizkiy & Cunningham (2004) studied the correlation between diversity measures and improvement in ensemble accuracy. Their study showed that, on average, the disagreement measure was one of the measures with the greatest correlation in accuracy improvement, and the double-fault measure was one of the measures with the lowest correlation. However, Aksela and Laaksonen (2006) stated that sometimes the disagreement measure was associated with higher classification accuracy than the double-fault measure, and sometimes with lower classification accuracy than the double-fault measure. They concluded the reason for these experimental phenomena was that selection of ensemble classifiers should be dependent on the combination method's characteristics. These two diversity measure methods were used in our experiments to investigate whether the selected ensemble members could improve ensemble effectiveness, in order to increase ensemble classification accuracy.
6.1 Diversity Measures

Both the disagreement measure and the double-fault measure are pairwise measurement methods, as they are able to measure the diversity in the predictions of a pair of classifiers. They both consider that the correct classification answers are known for all the samples. For a two-class problem, if a sample is correctly classified by a classifier, it is marked as 1, and otherwise it is marked as 0. Table 6.1 presents the matrix of the relationship between classifier $i$ and classifier $j$, with $N$ observations.

The number of times both classifiers are correct is denoted as $N^{11}$; the number of times both classifiers are incorrect is denoted as $N^{00}$ and the number of times one classifier is correct and another is incorrect is respectively denoted as $N^{10}$ and $N^{01}$. Thus, the total number of observations is $N = N^{11} + N^{00} + N^{10} + N^{01}$.

<table>
<thead>
<tr>
<th>classifier $i$</th>
<th>correct (1)</th>
<th>wrong (0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>correct (1)</td>
<td>$N^{11}$</td>
<td>$N^{10}$</td>
</tr>
<tr>
<td>wrong (0)</td>
<td>$N^{01}$</td>
<td>$N^{00}$</td>
</tr>
</tbody>
</table>

The total number of samples is $N = N^{11} + N^{00} + N^{10} + N^{01}$. $N^{11}$ denotes the number of times both classifiers are correct; $N^{00}$ denotes the number of times both classifiers are incorrect, and $N^{10}$ and $N^{01}$ denote the number of times when just the first or second classifier is correct respectively.

6.1.1 Disagreement Measure

With the disagreement measure, the diversity of a pair of classifiers, $i$ and $j$, is represented by the percentage of samples where they make different predictions and one of them is correct. This can be described as:

$$dis_{i,j} = \frac{N^{10} + N^{01}}{N} \quad (6.1)$$
A larger value of $\text{dis}_{ij}$ reflects greater diversity. Since the total number of samples $N$ for all pairs of classifiers is the same, it will not affect the comparison of diversities between pairs of classifiers whether $N^{00} + N^{01}$ divides the total number of samples $N$ or not. To simplify the calculation we only calculate $N^{00} + N^{01}$ for each pair of classifiers. Thus, the simplified version of the disagreement measure is:

$$\text{dis}_{simp}_{i,j} = N^{00} + N^{01} \quad (6.2)$$

The mean diversity over all pairs of $L$ classifiers is:

$$\text{dis}_{simp}_{\text{aver}} = \frac{2}{L(L-1)} \sum_{l=1}^{L} \sum_{j=l+1}^{L} \text{dis}_{simp}_{i,j} \quad (6.3)$$

Meanwhile, the general degree of diversity of a classifier $i$ is defined as:

$$\text{dis}_{simp}_i = \frac{1}{L-1} \sum_{j=1}^{L} \text{dis}_{simp}_{i,j} \quad (6.4)$$

### 6.1.2 Double-fault Measure

The double-fault measure is another method that examines the rate of how often the classifiers were incorrect. For a pair of classifiers $i$ and $j$, their double-fault is the percentage of samples for which both classifiers make the wrong predictions:

$$\text{df}_{i,j} = \frac{N^{00}}{N} \quad (6.5)$$

Again, this definition can be simplified by omitting the term of dividing $N$, and the simplified version is:

$$\text{df}_{simp}_{i,j} = N^{00} \quad (6.6)$$
The mean of all pairs of $L$ classifiers is:

$$df_{sim \text{aver}} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} df_{sim_{i,j}}$$

(6.7)

and the general diversity degree of a classifier $i$ is defined as:

$$df_{sim_{i}} = \frac{1}{L-1} \sum_{j=i}^{L} df_{sim_{i,j}}$$

(6.8)

### 6.2 Ensemble Construction

Both the disagreement measure and the double-fault measure consider the targets of samples as being already known. However, in the real world situation, we want the model to produce predictions for unknown samples whose target values are not known. Sometimes people use a validation dataset (in which target values are known and these samples are not used in training process) to evaluate the model's performance. For the most real world problems, it is a luxury to have a validation dataset, because one of the general issues for machine learning is a shortage of samples to use.

One of the advantages of using Bayesian learning on neural networks (the framework on which the ARD model is based) is that there is no requirement for a validation dataset. To keep this advantage, in this research the error diversity was measured on the basis of the predictions of the training samples and the predictions of the testing samples. If it was measured based on the training samples, the relationship matrix (see Table 6.1 above) for a pair of classifiers was determined based on the target values of training samples. If it was measured based on the testing samples, the average of outputs from all available classifiers was treated as criteria to measure whether a classifier classified a sample correctly or not. Thus, the diversities of each pair of classifiers were measured four times, because two diversity measures were respectively used on the training dataset and the testing dataset. With each combination, the classifier ensemble was reconstructed with selected members whose
general diversity degrees were bigger than the mean of all pairs of classifiers; for example, considering the error diversity measured by using the disagreement measure based on the training dataset, a classifier \( i \) is selected if its general diversity is bigger than the average diversity over all classifiers, \( \text{dis}_\text{simp}_i > \text{dis}_\text{simp}_{\text{aver}} \).

### 6.3 Experiments and Results

In total, 90 networks were created following the steps explained in Chapter 3. The ensemble which consists of all 90 networks was denoted as \( \text{ensAll} \). The simple averaging combination method was used to combine the outputs of ensemble members in order to produce ensemble predictions. The general classification accuracies of the predictions made by an ensemble and its members are listed in Table 6.2. These results show that the ensemble classification prediction is more accurate than the average performance of the ensemble members. Especially for the SSFN dataset, the ensemble prediction generally can classify 97.05% of samples correctly. However, the individual ensemble members on average can only classify 80.56% of samples correctly.

<table>
<thead>
<tr>
<th>datasets</th>
<th>( \text{ensAll} ) ensemble members mean%± std</th>
<th>( \text{ensAll} ) ensemble mean%± std</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pima</td>
<td>72.97±3.09</td>
<td>76.88±1.47</td>
</tr>
<tr>
<td>AAV</td>
<td>83.08±3.38</td>
<td>89.19±0.84</td>
</tr>
<tr>
<td>LICI</td>
<td>94.54±5.63</td>
<td>98.15±0.10</td>
</tr>
<tr>
<td>SSFN</td>
<td>80.56±18.87</td>
<td>97.05±0.22</td>
</tr>
</tbody>
</table>

Furthermore, the ensembles were reconstructed with selected classifiers. Four datasets (two diabetes datasets and two synthetic datasets) were used to test the effectiveness of selecting ensemble members. When using the disagreement measure, if the classifiers were selected on the basis of the diversity measured on the \textit{training} dataset, the new constructed ensemble was denoted as \( \text{ens1} \), and if the classifiers were
selected based on the diversity measured on the testing dataset, the new constructed ensemble was denoted as \textit{ens2}. The simple averaging combination method was used to combine the outputs of these selected classifiers and the accuracies of combined predictions are listed in Table 6.3. These results show that generally the ensemble predictions were more accurate than the predictions produced by individual classifiers. The \textit{ens2} (measured on testing dataset) provided slightly more accurate predictions than the \textit{ens1} (measured on training dataset) provided.

\begin{table}[h!]
\centering
\caption{AVERAGE CLASSIFICATION ACCURACIES OF ENSEMBLES AND THEIR MEMBERS WHICH WERE SELECTED BY USING THE DISAGREEMENT DIVERSITY MEASURE.}
\begin{tabular}{|l|l|l|l|l|l|l|l|}
\hline
\textbf{datasets} & \textbf{\textit{ens1}} & & & \textbf{\textit{ens2}} & & & \\
 & \textbf{members} & \textbf{ensemble} & \textbf{No. selected} & \textbf{members} & \textbf{ensemble} & \textbf{No. selected} & \\
 & \textbf{mean\%± std} & \textbf{mean\%± std} & \textbf{nets} & \textbf{mean\%± std} & \textbf{mean\%± std} & \textbf{nets} & \\
\hline
\textit{Pima} & 74.03 ± 3.37 & 75.00 ± 1.08 & 25.4 ± 1.82 & 71.40 ± 3.30 & 76.38 ± 2.09 & 39.8 ± 2.95 & \\
\textit{AAV} & 85.07 ± 3.55 & 88.65 ± 1.29 & 14.4 ± 1.52 & 81.15 ± 3.95 & 88.02 ± 0.82 & 37.0 ± 2.65 & \\
\textit{LIC1} & 83.19 ± 3.06 & 85.96 ± 0.79 & 16.6 ± 0.89 & 89.89 ± 7.19 & 86.28 ± 1.15 & 16.8 ± 1.30 & \\
\textit{SSFN} & 51.34 ± 5.42 & 59.71 ± 8.06 & 25.0 ± 2.74 & 82.94 ± 17.23 & 77.66 ± 3.03 & 29.6 ± 2.88 & \\
\hline
\end{tabular}
\end{table}

The error diversities were also measured using the double-fault diversity measure. The double-fault diversity measure was respectively used on the training dataset and the testing dataset. The term \textit{ens3} denoted the ensemble whose members were selected based on the training dataset. The term \textit{ens4} denoted the ensemble whose members were selected based on the testing dataset. Their classification results are listed in Table 6.4. The classification results obtained in this set of experiments also demonstrated that in general the classifier ensembles outperformed the single classifiers. Except for the LIC1 dataset, the classifiers selected based on the testing dataset, generally provided more accurate prediction than the classifiers selected based on the training dataset.
### Table 6.4

<table>
<thead>
<tr>
<th>datasets</th>
<th>ens3</th>
<th>ens4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>members</td>
<td>ensemble</td>
</tr>
<tr>
<td></td>
<td>mean% ± std</td>
<td>mean% ± std</td>
</tr>
<tr>
<td>Pima</td>
<td>72.55 ± 2.83</td>
<td>75.50 ± 0.81</td>
</tr>
<tr>
<td>AAV</td>
<td>82.70 ± 3.23</td>
<td>88.74 ± 0.45</td>
</tr>
<tr>
<td>LIC1</td>
<td>97.11 ± 0.96</td>
<td>98.60 ± 0.20</td>
</tr>
<tr>
<td>SSFN</td>
<td>90.44 ± 8.98</td>
<td>97.05 ± 0.19</td>
</tr>
</tbody>
</table>

When comparing the ensembles ens3 and ens4 with ensembles ens1 and ens2, more classifiers were selected and the ensemble prediction were more accurate by using the double-fault measure than using the disagreement measure. However, the ensembles with all available classifiers generally outperformed the ensembles with selected classifiers. This might be because that the selected classifiers have high degree of error diversity compared with all classifiers but they are similar to each other.

### 6.4 Summary and Discussion

The results presented in this chapter show that the ensembles with classifiers selected using the double-fault diversity measure were more suitable for the simple averaging combination method than those selected using the disagreement diversity measure. Measuring the error diversities based on the training dataset and testing dataset did not lead to much difference in classification accuracy. However, on average, the ensembles with all classifiers outperformed the ensembles with selected members. This might be because the selected classifiers generalized the data similarly to each other (the standard deviations between the selected classifiers were dramatically reduced for some datasets), so that the diversity levels between selected classifiers were decreased. Furthermore, this might also be because a single measure of diversity may not be accurate enough to capture all the relevant diversity in the ensemble. This was concluded by Kuncheva (2004) after studying the relationship between the diversity and accuracy improvement for the ten most commonly used diversity measures. If this is the reason for the results we observed, combining several measures to select the ensemble members might lead to some accuracy improvement.
However, Kuncheva (2004) also spotted that there was a weak relationship between diversity measure and ensemble accuracy. Thus, defining and using error diversity to create classifier ensembles may be heading toward a dead end, or some powerful new ensemble building methodologies may be over the horizon.

Although in our experimental study, the superior classification accuracy was not achieved by combining selected ensemble members, we obtained better classification accuracy by using selected features. Thus, for the case study in the next chapter, we will describe how selecting relevant features for some real world datasets improved classification accuracy.
Case Study: Sex Separation of Tsetse Fly Pupae using an Ensemble of ARD classifiers

Sex separation of tsetse fly pupae is a real world task, which provides us with an opportunity to test the capability of our ARD-based ensemble techniques on a real world dataset. Tsetse flies are an enormous health risk in parts of Africa, because they can transmit a fatal disease called trypanosomiasis which is an infection of the central nervous system. There are two forms of trypanosomiasis: Rhodesian (also known as nagana) and Gambian (also known as sleeping sickness). Nagana is transmitted from animal to animal and mostly affects cattle, horses, and wild animals like antelope, although humans can also be infected. These diseases reduce the growth rate of livestock, milk productivity, and the strength of farm animals, generally leading to the eventually death of the infected animals. Annually, about three million cattle die of these diseases. These deaths cause a direct annual loss of about $1.5 billion (Dowell, Parker, Benedict & Robinson, 2005). Sleeping sickness only affects humans. The World Health Organization conservatively estimated that in 2001 there were about 300,000-500,000 cases of Human African Trypanosomiasis with 60 million people at risk in 37 countries covering approximately 40% of Africa (Aksoy & Rio, 2005). These diseases could be extinguished by eradicating the tsetse flies, because they are the only vector for nagana and sleeping sickness.

Tsetse flies are a type of fly which can move over very long distances, so they are hard to control. There are several techniques that have been used to control tsetse flies, such as spraying insecticides and land clearing. However, aerial application of insecticides can harm the environment and is usually uneconomical. Land clearing involves the complete removal of any brush or woody vegetation from an area to make the area inhospitable to the flies. This technique requires continuous clearing efforts and can also cause environmental problems. Sometimes a large land clearing
programme can cause tsetse flies to spread to more populated areas. However, by late 1996, Zanzibar's tsetse flies had been successfully eradicated by natural birth control techniques. Both male and female tsetse flies feed exclusively on blood, so they both can transmit disease. However, sterilized female tsetse flies survive in the field longer than sterilized males so they have a much greater potential to act as vectors. Thus, sterilized male tsetse flies were released over infested areas in Zanzibar Island, rather than female tsetse flies. The wild females mating with sterile males produced no offspring. The sex separation for the sterile tsetse flies used in Zanzibar was done by hand-sexing the chilled adults. For a large amount of sterile male tsetse flies to meet the requirements, this sex separation method involved intensive labour and it could only be done after emergence from the pupae. Thus, an automatic procedure to separate the sex of tsetse flies at the pupal stage which is a few days before emergence would be preferable. That is because it would leave more time to transport sterilized male pupae to release sites.

By using near-infrared (NIR) spectroscopy, Dowell et al. (2005) found that there were significant differences between the spectra for the pupae of males and females. Their differences appear to be maximised at 4-5 days before emergence. Therefore, these differences can be used to automatically separate male and female pupae. In previous chapters, we discussed the characteristics of the ensemble-base ARD technique on feature selection and demonstrated that the classification accuracy can be improved by using selected features. In this chapter, we are going to use an ensemble of ARD models to select the relevant features in the tsetse fly dataset, in order to reduce the dataset dimension and improve the accuracy of sex separation.

The capability of the ARD-based ensemble technique on dimensionality reduction was compared with another two techniques: the Principal Component Analysis (PCA) and the Independent Component Analysis (ICA) techniques. Both the PCA and the ICA are statistical linear transformation methods. PCA projects the observed data onto orthogonal directions of greatest variances in the data. ICA transforms the observed data into non-Gaussian components which are as independent of each other as possible. They both can be used for dimension reduction, and the number of dimensions that the data can be reduced to relies on the magnitudes of the eigenvalues.
of the covariance matrix of the data. The PCA and the ICA techniques will be introduced in section 7.1 and 7.2 respectively with more detail.

In addition, a linear regression model was used as a baseline technique on classification compared to an ensemble of ARD classifiers.

### 7.1 Principal Component Analysis (PCA)

PCA is a simple linear technique for data analysis. It can identify patterns in data and highlight their similarities and differences, by linearly transforming a number of (possible) correlated variables into a (smaller) number of uncorrelated variables called principal components (Jackson, 1991). If the dataset is not very strongly curvilinear, the PCA technique can reduce the dimensionality of a dataset without much loss of information. To make the PCA technique work properly, firstly the mean of the data has to be subtracted from each of the data variables, in order to produce a zero mean dataset. After this, a covariance matrix $S$ will be calculated to obtain the eigenvectors $U$ and eigenvalues $L$ for this matrix. All the eigenvectors of the covariance matrix are orthogonal with each other. Each column of $U$, $u_1, u_2, \ldots, u_d$, is an eigenvector which contains the direction of cosine values for a corresponding regression line. Each element of $L$ is an eigenvalue which is associated with an eigenvector and indicates the variance of the data along the eigenvector direction (Bullinaria, 1997). The sum of the eigenvalues is equal to the total variance in the data.

The PCA technique can reduce the data dimensionality by mapping a dataset consisting of data vectors, $x_n$ for $n = 1, 2, \ldots, N$ ($N$ is the number of samples) in the space $\mathcal{V} = \mathbb{R}^d$ ($d$ is the number of the dataset dimensions) into vectors $z_n$ in the space $\mathcal{U} = \mathbb{R}^m$ ($m$ is the number of dimensions which the data is reduced to), a subspace of $\mathcal{V}$. The $r^{th}$ element in the vector $z_t$ is in the form:

$$z_{ij} = \sum_{k=1}^{d} (x_{ik} - \bar{x}) \ast u_{ij} \quad (7.1)$$
which is the sum of difference between a variable and its mean, projected on a regression line.

7.1.1 Determining the Number of Components

One problem for using the PCA technique is to determine the data dimensionality that a dataset should be reduced to. If the number of dimensions is reduced too much, and the variances in the dataset are not heavily concentrated in the first few components, then too much information will be thrown away (Browne, 1998). There are several methods which are commonly used to determine the number of components which should be retained.

In the visual inspection method, all the eigenvalues are plotted to find the optimal cut-off point between them. Sometimes this method is defined as "the rubble at the bottom of a cliff" (Jackson, 1991). Here, the retained eigenvalues are the cliff and the deleted ones are the rubble. Although this method is quite popular, there still can be some problems with it. First, the plotted curve may have no break. In other words, the plot might be a fairly smooth curve. Second, the plot may have more than one break. In this case, customarily we choose the first break to determine the retained principal components. Third, the first few of the eigenvalues may be so wildly separated that it is difficult to plot all the eigenvalues without losing the details about the rubble which may be necessary to determine the break. To solve this problem, the logarithms of the eigenvalues can be plotted instead.

A significance test also can be used to judge how many principal components should be retained. Typically, the significance level is 5% (notated as $\eta = 0.05$) which means that the probability of incorrectly rejecting the null hypothesis when it is actually true is 5% (Wright, 2002). In this case, the null hypothesis is that the eigenvalues associated with the removed principal components are not significantly different from each other. This method normally tends to retain some principal components which only contain very little of the data variance.
A simple and quick way to determine how many principal components should be retained involves taking the average root, which only retains those principal components whose eigenvalues exceed the average over all eigenvalues $\frac{\text{Tr}(L)}{d}$ (where $\text{Tr}(L)$ is the sum of the eigenvalues and $d$ is the dimension of the original dataset). Sometimes this cut-off is too high, so Jackson (1991) recommended using 70% of the average of the eigenvalues as a threshold.

### 7.2 Independent Component Analysis

*Independent Component Analysis* (ICA) is a statistical technique that linearly transforms the multidimensional observed data into components that are statistically as independent from each other as possible. This can be described in the form:

$$s = Wx \tag{7.2}$$

where $s$ is the independent components; $W$ is a constant matrix (weight) for linear transformation and $x$ is the observed data. The ICA technique requires that its independent components are non-Gaussian (strictly, at most only one of the components is Gaussian), as was discussed in Comon (1994).

Since independence implies uncorrelatedness, the concept of the ICA method can be seen as an extension of the principal component analysis method which searches the maximum variance in the data and leads to linear orthogonal projections of the data. As with the PCA technique, some researchers used the ICA technique to reduce the dimensionality of their data (Lemon, Mercier, Mouchot & Hubert-Moy, 2001; Zhu, Varshney & Chen, 2007). Moreover, the ICA technique is widely used in blind source separation (Jung, Makeig, Humphries, Lee, Mckeown, Iragui & Sejnowski, 2000; Popivanov, Jivkova, Stomonyakov & Nicolova, 2005). Here, we are going to use the ICA technique for dimensionality reduction, which is implemented by using the FastICA matlab software pack (Hyvarinen & Oja, 2000). The FastICA algorithm is based on a fix-point algorithm (Hyvarinen, 1999) for finding a maximum of the nongaussianity of the projected components. In practice, before applying an ICA algorithm, the observed data usually is preprocessed by *centering* and *whitening* to
make the ICA estimation simpler and better conditioned (Hyvarinen & Oja, 2000). In centering, the observed data $x$ subtracts its mean vector $E\{x\}$, in order for its mean to be zero. After centering, the data will be linearly transformed into uncorrelated components in the whitening procedure, and the variances of these components are unity. The whitening can be done by using the eigenvalue decomposition of the covariance matrix $E\{xx^T\} = ED E^T$, where $E$ is the orthogonal matrix of eigenvectors of $E\{xx^T\}$ and the $D$ is the diagonal matrix of its eigenvalues (Hyvarinen & Oja, 2000). The dimensionality of the data can be reduced when we do the whitening by looking at the eigenvalues of the covariance matrix of the observed data. Those with small eigenvalues are discarded (as done in the principal component analysis).

### 7.3 Dataset of Tsetse Fly Pupae

Near-infrared spectroscopy is commonly used to measure the characteristics of biological materials, since all biological materials absorb near-infrared radiation based on their unique chemical compositions (Dowell, Parker, Benedict & Robinson, 2005). The male and female tsetse pupae were proposed to have sufficiently different chemical compositions. Thus, sex separation of tsetse fly pupae can be done by using near-infrared spectroscopy. Dowell, et al. (2005) found that the difference of absorbing spectra for the male and female pupae appeared to be significantly maximised at 4-5 days before emergence. To have more time for shipping, tsetse fly pupae were scanned at 5 days before emergence, based on the spectra in the range 950-1650nm with a gap of 5nm. Each pupa was scanned 10 times with different angles at each spectrum. In total, there were 1510 female samples and 1690 male samples in the dataset, with 141 input features. To have a model learn both classes equally well, 1050 samples were randomly selected from each class to form a training dataset. Excluding these training samples, the remained 1100 samples constituted a testing dataset.
7.4 Experiments and Results

Following the steps presented in section 3.1, a classification ensemble was created with 90 classifiers. For each particular number of hidden units, there were 9 classifiers in the ensemble, and their general performance was obtained by averaging their outputs. The classification accuracies of the general performances for classifiers with different numbers of hidden units are plotted in Figure 7.1 along with the best and the worst performance among their corresponding 9 classifiers. This graph shows that (except for the averaged prediction for the classifiers with one hidden unit), the averaged predictions outperformed the best singles in their groups. The general trend of the accuracies of these averaged predictions was rising when the number of hidden units was increased, although there were some fluctuations. There seemed to be no sign that the rising would stop, even when the number of hidden units reaches 19. Thus, this indicated that classifiers having more than 19 hidden units can probably produce better performance.

![Figure 7.1](image)

*Figure 7.1. Classification accuracy produced by classifiers with numbers of hidden units up to 19. For each number of hidden units, the classification accuracy of the ensemble prediction obtained by averaging outputs from 9 corresponding classifiers is listed along with the best and the worst components.*

Based on the framework described in section 3.1, the number of hidden units was expanded with the aim of including some better performing classifiers in the ensemble. These newly created classifiers had more than 19 (odd numbers) of hidden units.
units. They were initialised with weights randomly selected from three ranges: (-0.01, 0.01), (-0.001, 0.001) and (-0.0001, 0.0001) and trained by three different optimisation algorithms: Quasi-Newton, Scaled Conjugate Gradient and Conjugate Gradient. However, when the number of hidden units was increased to more than 30, the simulation was terminated because of a shortage of memory. Due to this computational limitation, classifiers were created with no more than 29 hidden units. Therefore, the ensemble of classifiers was expanded from 90 members to 135 members. The correct percentage of the ensemble prediction obtained by averaging outputs of all ensemble members then was changed from 91.20% to 91.42% (see Table 7.1). For the expanded ensemble, the average performances of the classifiers with the same number of hidden units are plotted in Figure 7.2. This plot illustrates that the average performances of classifiers with more than 17 hidden units become more stable and accurate than the average performances of classifiers with a fewer number of hidden units. This might be because classifiers with fewer than 17 hidden units are not complex enough to generalise correctly the data for the sex separation task. By removing these classifiers, the number of ensemble members was changed from 135 to 63. Meanwhile, the ensemble classification accuracy increased from 91.42% to 91.60% (see Table 7.1).

<table>
<thead>
<tr>
<th>Number of ensemble members</th>
<th>Classification accuracy of ensemble members mean% ± std</th>
<th>Ensemble classification accuracy mean% ± std</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>87.46 ± 2.93</td>
<td>91.20 ± 0.38</td>
</tr>
<tr>
<td>135</td>
<td>87.66 ± 2.64</td>
<td>91.42 ± 0.31</td>
</tr>
<tr>
<td>63</td>
<td>88.07 ± 2.86</td>
<td>91.60 ± 0.23</td>
</tr>
</tbody>
</table>
Chapter 7: Case Study

Figure 7.2. Classification accuracies produced by classifiers with numbers of hidden units up to 29. The accuracy of the averaged output is listed with the single best and worst performances for each number of hidden units.

The feature relevance ranks determined by these 63 ARD classifiers were averaged to produce an ensemble feature relevance determination. The input features were sorted in the order of their relevance based on this ensemble determination and plotted in Figure 7.3. By visually inspecting this plot, we can see there are several disconnected 'break' points on the plot. Based on our previous experimental results, these points can be the potential indicators marking the boundaries of different relevance levels. Thus, these breaking points were used as reference for determining how many features should be selected. For this tsetse fly dataset, the breaking points were at the 7th, 13th, 53rd, 55th, 67th, 76th, 81st, 87th and 101st relevant features. These nine breaking points determined nine sets of input features. With each set of input features, an ensemble with 63 ARD classifiers ran five times to obtain the average ensemble performance, since it was time-consuming for each run. Table 7.2 contains a summary of the performances of these ensembles with their mean and standard deviation. These results show that the ensemble trained with 67 selected input features generally outperformed the ensembles trained with other sets of selected input features. On average, it classified 92.32% of pupae samples correctly. The standard deviations in the table are all quite small. This means the ensembles trained with each set of selected input features performed stably under each run.

In addition, the classification accuracies obtained by using a linear regression model with selected input features are also presented in Table 7.2. The best performance
(88.82%) was achieved by using 55 selected features, which is slightly better than the performance (87.27%) obtained by using all the input features.

By comparing the performances listed in Table 7.2, we can see that for each set of selected input features, the ARD classifier ensembles outperformed the linear regression model. With fewer than half number of available features, both the ARD-based classifier ensembles and the linear regression model performed no worse than using all input features. However, these two methods produced their best performance with two different sets of selected input features. This might be because, as a non-linear method and a linear method, they respectively found their solutions for correctly separating the gender of pupae by focusing on different characteristics in the dataset.

![Figure 7.3. Averaged feature relevance ranks obtained by combining the feature relevance ranks produced by 63 ARD classifiers](image-url)
As an alternative to selecting relevant features by using our ARD-based method, the dimensionality of data can also be reduced by using the features extracted using the PCA and the ICA techniques. For this tsetse fly dataset, the sum of the 6 biggest eigenvalues includes 99.9% of the values of all non-zero eigenvalues. To have the transformed components retain the most significant information in the data, the original dataset was transformed into 6 principal components using the PCA technique and 6 independent components using the ICA technique.

The extracted 6 principal components were used to train an ensemble of classifiers created following the scheme described in section 3.1, with the number of hidden units varied up to 19. The general performances of the classifiers with the same number of hidden units were obtained by averaging their outputs. Their classification accuracies are plotted in Figure 7.4 along with the best and the worst performances in their corresponding groups. This graph shows that for each number of hidden units, the averaged prediction generally outperforms the best single classifier in the group, and there is no tendency of classifiers with more than 19 hidden units to classify pupae samples more correctly. Although classifiers with fewer than 9 hidden units did not produce good classification results, removing these classifiers from the ensemble did not bring better classification accuracy. Thus, the outputs of all 90 classifiers were averaged to generate an ensemble classification prediction which...
classified 83.5% of pupae samples correctly. Meanwhile, a linear regression model could classify 80.27% of pupae samples correctly using these 6 principal components.

![Figure 7.4](image)

**Figure 7.4.** The average performance for the classifiers with the same number of hidden units and trained on 6 principal components obtained using the PCA technique. The accuracy of the averaged output is listed with the single best and worst performances for each number of hidden units.

Similar experiments were done for the ICA technique. An ensemble of classifiers was created with odd numbers of hidden units from 1 to 19. Among of these classifiers, the averaged performances of classifiers with the same number of hidden units are plotted in Figure 7.5 along with the best and worst performance of the classifiers in their corresponding groups. The plot in Figure 7.5 shows that although there are some fluctuations, there is no tendency for the classification to be more accurate by increasing the number of hidden units. Thus, the ensemble classification prediction based on the 6 independent components was made by averaging the outputs of all 90 classifiers. This ensemble prediction classified 45.4% of the pupae samples correctly (see Table 7.3). On the other hand, 54.73% of pupae samples were correctly classified by using a linear regression method with these 6 independent components (see Table 7.3). Recalling that the ICA technique extracts independent components from the observed data, the process of this extraction does not involve the target variable, and for an observed dataset with highly correlated features, some significant
information might be lost with this feature extraction technique. This might be a reason for the poor performance produced by using the ICA technique.

### Table 7.3

**Classification Accuracies with Principal Components and Independent Components**

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Ensemble Prediction mean% ± std</th>
<th>Linear Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 PCs</td>
<td>83.86 ± 0.42</td>
<td>80.27%</td>
</tr>
<tr>
<td>6 ICs</td>
<td>49.53 ± 2.35</td>
<td>54.73%</td>
</tr>
</tbody>
</table>

**Figure 7.5.** The average performance for the classifiers with the same number of hidden units and trained on 6 independent components obtained by using the ICA technique. The accuracy of the averaged output is listed with the single best and worst performances for each number of hidden units.

The above experimental results show that the selected features can produce more accurate classification predictions compared to the principal components and the independent components, and that generally an ensemble of ARD classifiers outperforms the linear regression model. The classification predictions made by using independent components only classified about 50% of tsetse pupae correctly, which is roughly as accurate as the results obtained by randomly guessing. Thus, for later experiments we only tried our ensemble-ARD based feature selection rather than the principal components and independent components techniques.
Apart from this tsetse fly dataset, we also experimented on other tsetse fly datasets. However, each ensemble of ARD classifiers only ran once rather than five times for these datasets, in consideration of the time-consumption nature of each run, and the stability of the ensemble classification predictions obtained after each run. One dataset was obtained by scanning pupae samples at approximate 3 days before emergence. Another dataset was obtained by scanning pupae samples at approximate 5 days before emergence. For these datasets, the pupae were scanned with a fixed angle but not with 10 different angles. An ensemble of ARD classifiers could classify 87.6% of pupae samples in the 3 days dataset correctly by using 22 selected input features compared to the classification accuracy of 85.2% obtained by using all the input features. Furthermore, 99.2% of pupae samples in the 5 days dataset were correctly classified by using 27 selected input features, and the correct percentage with all features was 96.9%. Thus, the pupae samples scanned at about 5 days before emergence can provide better classification prediction than the samples scanned at about 3 days before emergence.

To build a model which can be used to separate sex for future pupae samples, four 5 days tsetse fly datasets were collected on four different days with a fixed scan angle. The earlier collected three datasets were used to train the model, and the later collected 5 days dataset was used to test the performance of the model. However, only 77.39% of pupae samples were correctly classified using all the input features and 77.61% of the samples were correctly classified using 132 selected input features. This unexpectedly poor result suggests that there is some variability in the data collection process from the pupae which occurs on a day-to-day process. This variability needs to be addressed in further investigations to make the automated gender separation of pupae a reality.

### 7.5 Summary and Discussion

In this chapter, an ensemble of ARD models was used to select relevant features in order to reduce the dimensionality of data. In addition, a comparison was made using the PCA and the ICA techniques to project data into lower dimensional data spaces. These dimensionally reduced datasets then were applied to an ensemble of classifiers
and a linear regression model to separate tsetse fly pupae by their gender. Based on our experimental results, we can see the features selected using an ensemble of ARD models can produce more accurate classification predictions than the components obtained using the PCA and the ICA techniques. The best performances obtained by using an ensemble of ARD classifiers with selected features, principal components and independent components for a 5 days dataset are 92.32%, 83.86% and 49.53% respectively. Meanwhile, the classification capability of an ensemble of ARD classifiers is better than the linear regression model.

By using the ensemble feature selections on different tsetse fly datasets, we can see that the accuracy of sex separation can be affected by the quality of the data. The quality of the data can be influenced by the status of the pupae development and the methods used to scan pupae. The classification predictions made based on two 5 days datasets can classify the gender of pupae more accurately than the predictions made based on the 3 days dataset. This matches previous research findings that the significant differences between the spectra for the pupae of male and female appear to be maximised at about 5 days before emergence of the adults (Dowell et al., 2005). Moreover, the experimental results for the two 5 days datasets show that pupae scanned with a fixed angle can be classified more correctly than those scanned with different angles. However, the model trained with three 5 days datasets scanned with a fixed angle could not produce highly accurate predictions on a 5 days dataset collected at a later date. Thus, it is likely that there are some other factors involved in the data collection process that can affect the quality of the scanned pupae samples, and further investigation is required to address these factors.
Chapter 8

Conclusion and Future Work

The complexity of data can directly affect the efficiency and the effectiveness of classifiers. Efficient feature selection can effectively control the complexity of data by reducing the dimensionality of multivariate data and removing most irrelevant and redundant features from data. The feature selection technique studied in this thesis is based on the ARD technique which attempts to reveal the relevance of input features by seeking the right distribution of corresponding weights in the network. However, our experimental observations demonstrated that the feature relevance determined by a single ARD model was not stable and reliable. We have used a neural network ensemble technique to reduce the variances in an ensemble of feature relevance determinations. Our experimental results show that after averaging a number of ARD results the ensemble prediction of feature relevance started to be stable. However, the ensemble feature relevance determination could not determine the same relevance rank to the features which are equally relevant. This failure may be due to the influence of the correlations between features on the feature relevance determination. Even though an ensemble feature relevance determination can not rank the relevance for individual features correctly, it still can be a good feature selection method as it effectively separates relevant features from irrelevant and redundant features, and provides useful suggestions about which sets of features should be selected.

As a non-linear technique, the ARD technique was compared with a linear technique which determined the feature relevance also based on the contribution of a feature to the solution of a task. From this comparison we can see that the multiple linear regression method can detect noise in the dataset but cannot reduce redundancy from the dataset, whereas the ensemble-based ARD technique can. Moreover, the relevant features selected by using an ensemble of ARD models improved the classification accuracy for two diabetes datasets.
Chapter 8: Conclusion and Future Work

Following the same line of thought, of feature selection reducing the irrelevant and redundant features from the data, we experimented with selecting ensemble members in an attempt to improve the ensemble classification accuracy. However, the experimental results show that combining ensemble members selected by using double-fault and disagreement diversity measurements cannot improve the ensemble classification accuracy. This might be because a single diversity measurement method is not accurate enough to capture all the relevant diversities in the ensemble. It also might be because the outputs of all ensemble members can represent the distribution of the prediction space better than the outputs of the selected members. Although the classification accuracy was not improved with selected ensemble members, it was effectively improved with selected relevant features. Our ensemble-based ARD technique was applied on a real world task, separating tsetse fly pupae by their gender. The quality of the reduced-dimension data with features selected by an ensemble of ARD models was justified by the PCA and the ICA dimensionality reduction techniques.

Principal component analysis is a commonly used dimensionality reduction method, which forces the variances in the data onto several orthogonal components. The criteria about how many dimensions the original dataset should be reduced to are based on magnitudes of eigenvalues of the covariance matrix of the original data. As an extension concept of the PCA technique, the ICA technique also can be used as a dimensionality reduction method. The ICA technique transforms the original data into several non-Gaussian components which are statistically as independent to each other as possible. The capability of an ensemble of ARD networks was compared with the PCA and the ICA technique. The compared results show that the dimensionally reduced dataset (generated by using ARD ensembles) with 67 selected features can generally classify 92.32% of the samples correctly. The classification accuracy is much higher than the percentage of 83.5% obtained by using 6 principal components and the percentage of 45.4% obtained by using 6 independent components. Additionally, another three tsetse fly datasets were studied: two 5 days datasets and one 3 days dataset. The difference between the classification accuracies obtained by using 5 days and 3 days tsetse fly datasets supported the previous research findings that the significant differences between the spectra for the pupae of male and female appear to be maximized at about 5 days before emergence of the
adults (Dowell et al., 2005). This result shows that the quality of pupae samples of tsetse fly pupae is sensitive to the pupal developing stage. The difference between the classification accuracies obtained by using two 5 days datasets (one obtained by scanning samples with a fixed angle and another obtained by scanning samples with 10 different angles) show that the quality of pupae samples is also sensitive to the conditions of the sample scanning. Because of these two or any other factors, the ARD ensembles (trained with 3 old 5 days datasets and tested with a new 5 days dataset) only classified 77.39% of tsetse pupae correctly with all the input features and classified 77.61% of pupae correctly with 132 selected features.

8.1 Future Work

So far in this thesis we introduced not only some achievements but also presented some as yet unsolved problems which need to be studied in future work.

The relevant features in a dataset can be properly separated from redundancy and noise but their determined relevance ranks can be affected by the correlation with their redundancies, although there is some degree of tolerance of this influence. In our future work, we are going to study the tolerance degree of this influence and theoretically study this finding, because so far it is based only on experimental observations. Furthermore, the neural network ensemble combination methods are also worth studying in order to reduce this influence. While reducing the influence on ensemble feature relevance ranks, the boundary between relevant features and irrelevant features hopefully can be more clearly marked. Then we can know exactly which features should be selected. If not, a further technique needs to be found for that boundary determination. To date we have several suggestions and need to run experiments to detect which suggestion is more accurate.

Apart from feature selection, we also experimented with selecting ensemble members by using two diversity measure methods. Although in our experiments the selected members did not produce better performance compared with using all members, some researchers have produced benefits from selecting members and, as we know, neural network ensembles do benefit from ensemble diversities to provide better predictions.
Thus, in future work, a new method needs to be designed in order to create good ensemble diversity by selecting ensemble members.

Furthermore, another drawback of this technique is that it is very time consuming. Because the ARD technique can be applied on other types of models such as the support vector machine (SVM), we will try an ensemble of other types of ARD models, instead of MLPs, to save running time.

Despite the drawbacks noted above, we are confident that our work has successfully demonstrated that the features selected by using an ensemble of ARD models can effectively improve classification performance for some linear and non-linear methods.
References


References


References


Neural Networks: Tricks of the Trade, 1524, pp. 55-69. Berlin, Heidelberg, New 
York: Springer-Verlag.

Relevance Determination by Expectation Propagation. In the Proceedings of Twenty-
first International Conference on Machine Learning, July 4-8, Banff, Alberta, Canada.

for Features. IEEE Transactions on Knowledge and Data Engineering, 17(9), 1199-
1207.

representations by error propagation. In Rumelhart, D. E., McClelland, J. L., Parallel 
Distributed Processing Explorations in the Microstructure of Cognition (318-362). 
Cambridge, MA, USA: MIT Press.


197-227.

Scientific Publishing Co. Pte. Ltd.

Artificial Neural Nets: Ensemble and Modular Multi-Net Systems (1-30). Springer-
Verlag.

Skalak, D. B. (1996). The Sources of Increased Accuracy for Two Proposed Boosting 
Algorithms. Proceedings AAAI 96 Workshop on Integrating Multiple Learning 
Models for Improving and Scaling Machine Learning Algorithms 120-125. Portland, 
Oregon, USA.


Recognition of Unconstrained Handwritten Numerals. Proceedings of the IEEE, 89(7), 
1162-1180.


near infrared spectroscopy. IEEE Transactions on Neural Networks, 7, 56-72.


